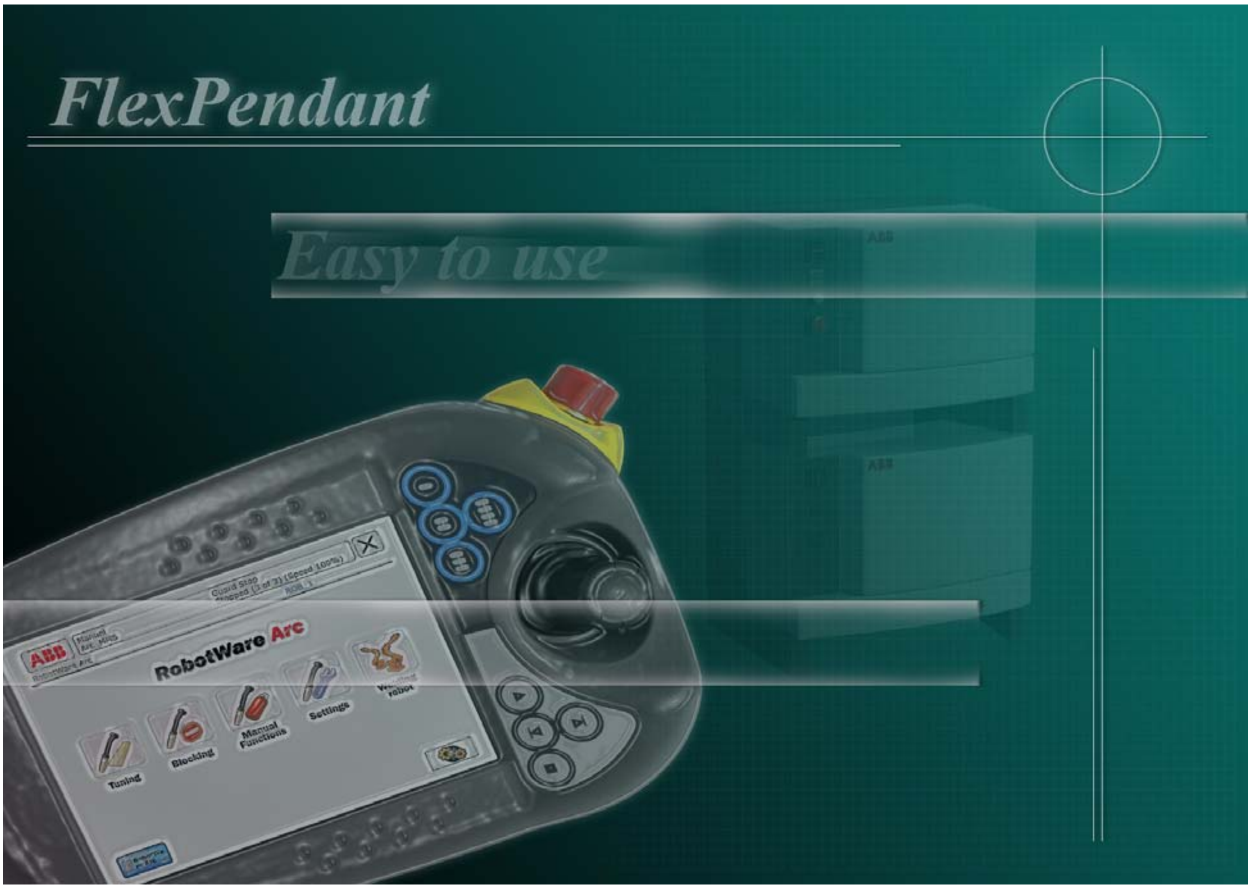
The heart of Robotics

Operating manual

IRC5 with FlexPendant

Controller software IRC5
M2004



Operating manual
IRC5 with FlexPendant

M2004

Document ID: 3HAC16590-1

Revision: K

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, and contents thereof must not be imparted to a third party nor be used for any unauthorized purpose. Contravention will be prosecuted.

Additional copies of this manual may be obtained from ABB at its then current charge.

ABB AB
Robotics Products
SE-721 68 Västerås
Sweden

Overview

About this manual

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

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

How to read the operating manual

The operating manual is structured in the following chapters.

Chapter	Title	Content
1	Safety	Safety instructions and warnings.
2	Welcome to FlexPendant	Descriptions of the FlexPendant and the controller.
3	Get started	Descriptions of connections and step-by-step instructions to the most common tasks
4	Navigating and handling the FlexPendant	Descriptions of the FlexPendant's user interface and basic procedures.
5	Jogging	Procedures for jogging.
6	Programming and testing	Procedures for programming and testing, including descriptions of some concepts for programming.
7	Running in production	Procedures for running in production.
8	Handling inputs and outputs, I/O	Procedures for handling I/O.
9	Handling the event log	Procedures for the event log.
10	Systems	Procedures for restart, backup, restore, and configuring systems.
11	Calibrating	Procedures for calibrating the robot system.
12	Changing FlexPendant settings	Procedures for changing the settings for the FlexPendant.
13	Descriptions of terms and concepts	Descriptions of terms and concepts used in robotics.

Prerequisites

The reader should:

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

Continues on next page

References

Reference	Document ID
Product manual - IRC5	3HAC021313-001
Operating manual - Getting started, IRC5 and RobotStudio	3HAC027097-001
Operating manual - RobotStudio	3HAC032104-001
Operating manual - Service Information System	3HAC025709-001
Operating manual - Trouble shooting	3HAC020738-001
Technical reference manual - System parameters	3HAC17076-1
Technical reference manual - RAPID overview	3HAC16580-1
Technical reference manual - RAPID Instructions, Functions and Data types	3HAC16581-1
Technical reference manual - RAPID kernel	3HAC16585-1
Application manual - Additional axes and stand alone controller	3HAC021395-001
Application manual - Engineering tools	3HAC020434-001
Application manual - Motion coordination and supervision	3HAC18154-1
Application manual - Motion functions and events	3HAC18152-1
Application manual - MultiMove	3HAC021272-001
Operating manual - Calibration Pendulum	3HAC16578-1

Revisions

Revision	Description
-	First issued. IRC5 M2004. Released with RobotWare 5.04.
A	Released with RobotWare 5.05.
B	Released with RobotWare 5.06. Organization of chapters restructured to task orientation.
C	Released with RobotWare 5.07.
D	Released with RobotWare 5.07.01.
E	Released with RobotWare 5.07.02.
F	Minor corrections.
G	Released with RobotWare 5.08.
H	Released with RobotWare 5.09. Description of displacements added.
J	Released with RobotWare 5.10. Some changes to the **Program Editor**, menus **Edit** and **Debug**. Some changes to Quickset menu, Mechanical unit.
K	Released with RobotWare 5.11. Minor corrections in section Restart procedures. Details describing the difference between PP to Main from the Production window and the Program editor is added to section Starting programs - Restart from the beginning. RobotStudio Online is integrated in RobotStudio.

Product documentation, M2004

General

The robot documentation is divided into a number of categories. This listing is based on the type of information contained within the documents, regardless of whether the products are standard or optional. This means that any given delivery of robot products **will not contain all** documents listed, only the ones pertaining to the equipment delivered.

However, all documents listed may be ordered from ABB. The documents listed are valid for M2004 robot systems.

Product manuals

All hardware, robots and controllers, will be delivered with a **Product manual** that contains:

- Safety information
- Installation and commissioning (descriptions of mechanical installation, electrical connections)
- Maintenance (descriptions of all required preventive maintenance procedures including intervals)
- Repair (descriptions of all recommended repair procedures including spare parts)
- Additional procedures, if any (calibration, decommissioning)
- Reference information (article numbers for documentation referred to in Product manual, procedures, lists of tools, safety standards)
- Part list
- Foldouts or exploded views
- Circuit diagrams

Technical reference manuals

The following manuals describe the robot software in general and contain relevant reference information:

- **RAPID Overview**: An overview of the RAPID programming language.
- **RAPID Instructions, Functions and Data types**: Description and syntax for all RAPID instructions, functions and data types.
- **System parameters**: Description of system parameters and configuration workflows.

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, CD with PC software)
- How to use the application
- Examples of how to use the application

Continues on next page

Continued

Operating manuals

This group of manuals is aimed at those having first hand operational contact with the robot, that is production cell operators, programmers and trouble shooters. The group of manuals includes:

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

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.

1.2. Applicable safety standards for IRC5

Health and safety standards

The robot complies fully with the health and safety standards specified in the EEC’s Machinery Directives.

The ABB robots controlled by the IRC5 conforms to the following standards:

Standard	Description
EN ISO 12100-1	Safety of machinery, terminology
EN ISO 12100-2	Safety of machinery, technical specifications
EN 954-1	Safety of machinery, safety related parts of control systems
EN ISO 10218-1:2006	Robots for industrial environments - Safety requirements - Part 1 Robot
EN ISO 60204-1:2005	Safety of machinery - Electrical equipment of machines - Part 1 General requirements
EN 61000-6-4 (option)	EMC, generic emission
EN 61000-6-2	EMC, generic immunity

Standard	Description
IEC 60204-1	Electrical equipment of industrial machines
IEC 60529	Degrees of protection provided by enclosures

Standard	Description
EN ISO 10218-1:2006	Robots for industrial environments - Safety requirements - Part 1 Robot
ISO 9787	Manipulating industrial robots, coordinate systems and motions

Standard	Description
ANSI/RIA 15.06/1999	Safety requirements for industrial robots and robot systems
ANSI/UL 1740 (option) ANSI/UL 1998 (option)	Safety standard for robots and robot equipment
CAN/CSA Z 434-03 (option)	Industrial robots and robot systems - General safety requirements

1.3 Safety terminology

1.3.1. Safety signals, general

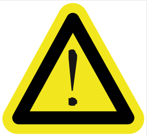
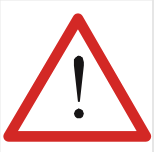
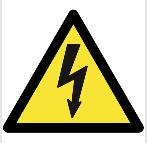
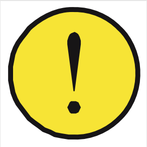
General

This section specifies all dangers that may arise from performing the work detailed in the manual. Each danger is detailed in its own section consisting 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.
- An instruction of how to eliminate the danger to facilitate performing the activity at hand.

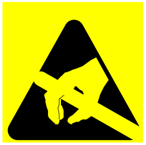
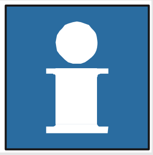
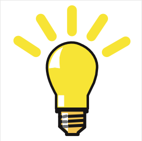
Danger levels

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

Symbol	Designation	Signification
 danger	DANGER	Warns that an accident *will* 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 etc.
 warning	WARNING	Warns that an accident *may* 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.
 Electrical shock	ELECTRICAL SHOCK	The electrocution or electrical shock symbol indicates electrical hazards which could result in severe personal injury or death.
 caution	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.

Continues on next page

Continued

Symbol	Designation	Signification
 Electrostatic discharge (ESD)	ELECTROSTATIC DISCHARGE (ESD)	The electrostatic discharge (ESD) symbol indicates electrostatic hazards which could result in severe damage to the product.
 Note	NOTE	Note symbols alert you to important facts and conditions.
 Tip	TIP	Tip symbols direct you to specific instructions, where to find additional information or how to perform a certain operation in an easier way.

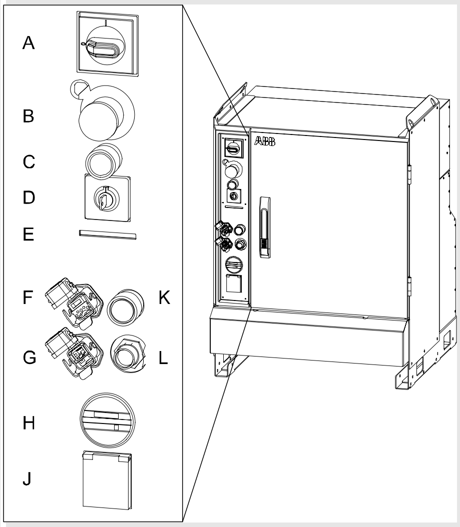
1.3.2. DANGER

1.3.2.1. DANGER - Make sure that the main power has been switched off!

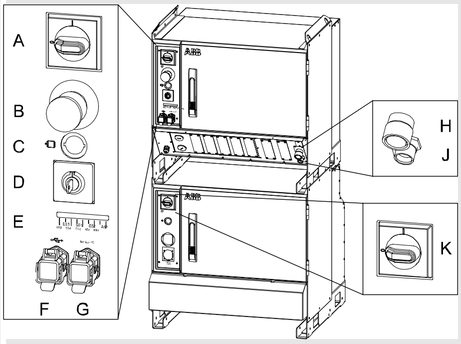
Description

Working with high voltage is potentially lethal. Persons subjected to high voltage may suffer cardiac arrest, burn injuries, or other severe injuries. To avoid these dangers, do not proceed working before eliminating the danger as detailed below.

Elimination, Single Cabinet Controller

	Action	Note/illustration
1.	Switch off the main switch on the controller cabinet.	 xx0600002782 A: Main switch

Elimination, Dual Cabinet Controller

	Action	Note/illustration
1.	Switch off the main switch on the Control Module.	 xx0600002783 A: Main switch, Control Module
2.	Switch off the main switch on the Drive Module.	K: Main switch, Drive Module (see illustration above)

1.3.2.2. DANGER - Moving robots are potentially lethal!

Description

Any moving robot is a potentially lethal machine.

When running the robot, it may perform unexpected and sometimes irrational movements. Moreover, all movements are performed with great force and may seriously injure any personnel and/or damage any piece of equipment located within the working range of the robot.

Elimination

	Action	Note
1.	Before attempting to run the robot, make sure all *emergency stop equipment* is correctly installed and connected.	Emergency stop equipment such as gates, tread mats, light curtains, etc.
2.	Usually the hold-to-run function is active only in manual full speed mode. To increase safety it is also possible to activate hold-to-run for manual reduced speed with a system parameter. The hold-to-run function is used in manual mode, not in automatic mode.	How to use the hold-to-run function in RobotWare 5.0 is detailed in section *How to use the hold-to-run function* in the *Operating manual - IRC5 with FlexPendant*.
3.	Make sure no personnel are present within the working range of the robot before pressing the start button.	

1.3.2.3. DANGER - Robot without axes' holding brakes are potentially lethal!

Description

Since the robot arm system is quite heavy, especially on larger robot models, it is dangerous if the holding brakes are disconnected, faulty, worn or in any way rendered non-operational.

For instance, a collapsing IRB 7600 arm system may kill or seriously injure a person standing beneath it.

Elimination

	Action	Info/illustration
1.	If you suspect that the holding brakes are non-operational, secure the robot arm system by some other means before working on it.	Weight specifications etc. may be found in the *Product manual* of each robot model.
2.	If you intentionally render the holding brakes non-operational by connecting an external voltage supply, the utmost care must be taken! **NEVER** stand inside the robot working area when disabling the holding brakes unless the arm system is supported by some other means!	How to correctly connect an external voltage supply is detailed in the *Product manual* of each robot model.

1.3.3. WARNING

1.3.3.1. WARNING - The unit is sensitive to ESD!

Description

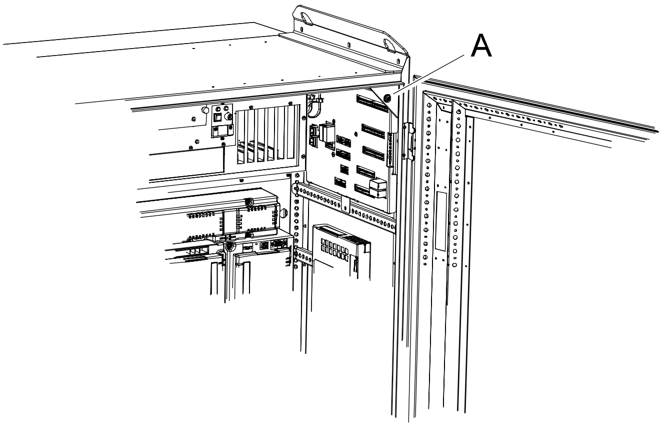
ESD (electrostatic discharge) is the transfer of electrical static charge between two bodies at different potentials, either through direct contact or through an induced electrical field. When handling parts or their containers, personnel not grounded may potentially transfer high static charges. This discharge may destroy sensitive electronics.

Elimination

	Action	Note
1.	Use a wrist strap	Wrist straps must be tested frequently to ensure that they are not damaged and are operating correctly.
2.	Use an ESD protective floor mat.	The mat must be grounded through a current-limiting resistor.
3.	Use a dissipative table mat.	The mat should provide a controlled discharge of static voltages and must be grounded.

Location of wrist strap button

The wrist strap button is located in the right corner as shown in the illustration below.



xx0500002171

A	Wrist strap button

1.3.4. What is an emergency stop?

Definition of emergency stop

An emergency stop is a state that overrides any other robot control, disconnects drive power from the robot motors, stops all moving parts, and disconnects power from any potentially dangerous functions controlled by the robot system.

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, i.e, resetting the emergency stop button and pressing the Motors On button, in order to return to normal operation.

The robot system can be configured so that the emergency stop results in either:

- An uncontrolled stop, immediately stopping the robot actions by disconnecting power from the motors.
- A controlled stop, stopping the robot actions with power available to the motors so that the robot path can be maintained. When completed, power is disconnected.

The default setting is uncontrolled stop. However, controlled stops are preferred since they minimize extra, unnecessary wear on the robot and the actions needed to return the robot system back to production. Please consult your plant or cell documentation to see how your robot system is configured.

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. How to perform normal program stops, see *Stopping programs on page 230*.

Classification of stops

The safety standards that regulates automation and robot equipment defines categories in which each type of stop applies:

If the stop is...	... then it is classified as...
uncontrolled	category 0 (zero)
controlled	category 1

Continues on next page

Emergency stop devices

In a robot system there are several emergency stop devices that can be operated in order to achieve an emergency stop. There are emergency stop buttons available on the FlexPendant and on the controller cabinet (on the Control Module on a Dual Cabinet Controller). 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.

1.3.5. What is a safety stop?

Definition of safety stops

A safety stop means that only the power to the robot motors is disconnected. 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 can be configured so that the stop results in either:

- An uncontrolled stop, immediately stopping the robot actions by disconnecting power from the motors.
- A controlled stop, stopping the robot actions with power available to the motors so that the robot path can be maintained. When completed, power is disconnected.

The default setting is controlled stop.

Controlled stops are preferred since they minimize extra, unnecessary wear on the robot and the actions needed to return the robot system back to production. Please consult your plant or cell documentation to see how your robot system is configured.

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 robot. How to perform normal program stops, see *Stopping programs on page 230*.

Classification of stops

The safety standards that regulates automation and robot equipment defines categories in which each type of stop applies:

If the stop is...	... then it is classified as...
uncontrolled	category 0 (zero)
controlled	category 1

Type of safety stops

Safety stops are activated through special signal inputs to the controller, see *Product manual - IRC5*. The inputs are intended for safety devices such as cell doors, light curtains, or light beams.

Safety stop:	Description:
Automatic mode stop (AS)	Disconnects drive power in automatic mode. In manual mode this input is inactive.
General stop (GS)	Disconnects drive power in all operating modes.
Superior stop (SS)	Disconnects drive power in all operating modes. Intended for external equipment.

NOTE!

Use normal program stop for all other type of stops.

1.3.6. What is safeguarding?

Definition

Safeguarding are safety measures consisting of the use of safeguards to protect persons from hazards which cannot reasonably be removed or sufficiently eliminated by design.

A safeguard prevents hazardous situations by stopping the robot in a controlled manner when a certain safeguarding mechanism such as a light curtain is activated. This is done by connecting the safeguard to any of the safety stop inputs at the robot controller.

The safety stops described in *What is a safety stop? on page 23*, should be used for safeguarding.

NOTE!

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

NOTE!

The safeguarding should not be used for normal program stops as this causes extra, unnecessary wear on the robot. How to perform normal program stops, see *Stopping programs on page 230*.

Safeguarded space

The safeguarded space is the space guarded by the guards. For example, a robot cell is safeguarded by the cell door and its interlocking device.

Interlocking devices

Each present guard has an interlocking device which, when activated stops the robot. The robot cell door has an interlock that stops the robot when the door is opened. The only way to resume operation is to close the door.

Safeguarding mechanisms

A safeguarding mechanism consists of a number of guards connected in series. When a guard is activated, the chain is broken and the machine operation is stopped regardless of the state of the guards in the rest of the chain.

NOTE!

Use normal program stop for all other type of stops.

1.3.7. Safe use of the FlexPendant

NOTE!

The enabling device is a push button located on the side of the FlexPendant which, when pressed halfway in, takes the system to MOTORS ON. When the enabling device is released or pushed all the way in, the robot is taken to the MOTORS OFF state.

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

- The enabling device must never be rendered inoperative 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.
- The programmer must always bring the FlexPendant with him/her, when entering the robot's working space. This is to prevent anyone else taking control of the robot without the programmer knowing.

Enabling device

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, robot operation is impossible.

Hold-to-run function

The hold-to-run function allows movement when a button connected to the function is actuated manually and immediately stops any movement when released. The hold-to-run function can only be used in manual mode.

How to operate the hold-to-run function is detailed in *Operating manual - IRC5 with FlexPendant*.

1.4 How to deal with an emergency

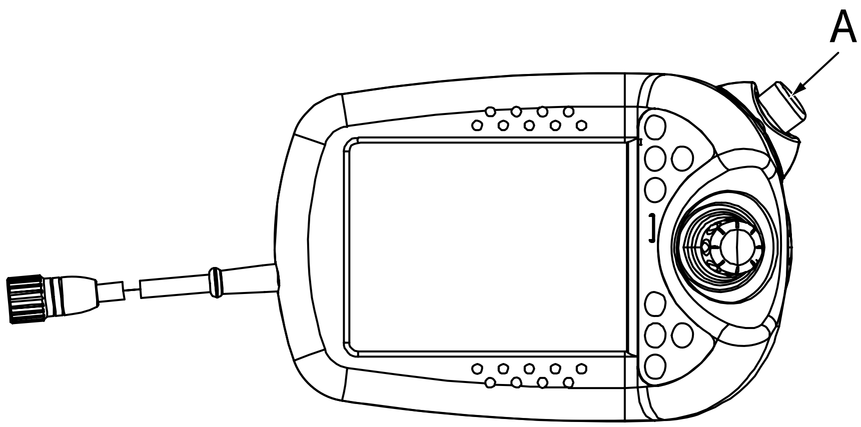
1.4.1. Stop the system

Overview

Press any of the emergency stop buttons immediately if:

- There are any personnel in the robot working area, while the robot is working.
- The robot causes harm to personnel or mechanical equipment.

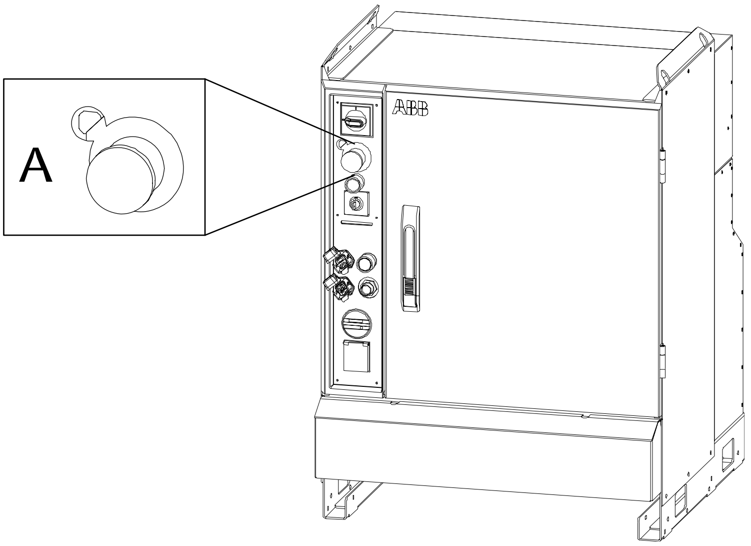
The FlexPendant emergency stop button



xx0300000449

A	Emergency stop button

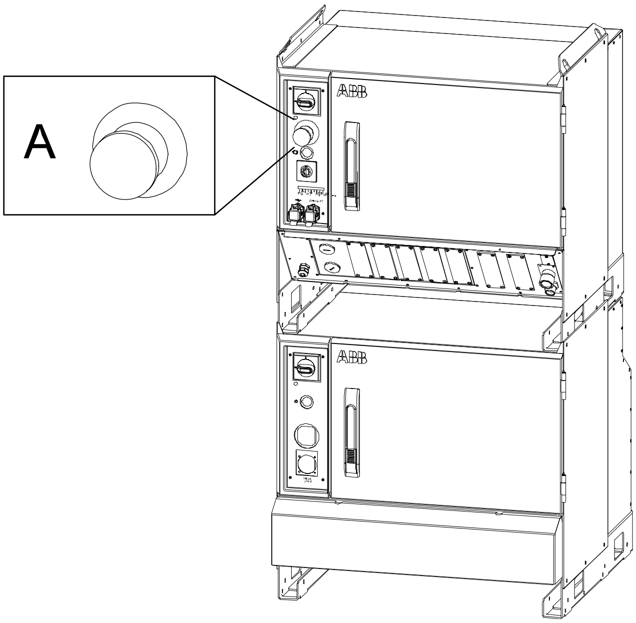
The controller emergency stop button



xx0600003423

A	Emergency stop button, Single Cabinet Controller

Continues on next page



xx0600003424

A	Emergency stop button, Dual Cabinet Controller

Other emergency stop devices

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

1.4.2. Release the robot holding brakes

Overview

The robot’s brakes may be manually released as long as power is available. As long as the controller's power switch is in its on position, power is available and applied even if the system is in emergency state.

Battery power

In case of a plant or cell power outage the brake system may be powered by a battery. How to connect the battery is different for each robot model. This is detailed in the *Product Manual* delivered with the robot.

Brake release buttons

Brake release buttons are placed differently depending on robot type, this is detailed in the *Product Manual*.

Always learn where the buttons are placed on robot models you work with.

Precautions

Before releasing the brakes verify:

- Which way will the arm go?
- How will an entangled object be affected?

A minor damage can easily become serious if the consequences are not considered.

⚠ **DANGER!**

Releasing the brakes is a hazardous action that may cause injury and damage property. It must be done with great care and only when absolutely necessary.

Releasing brakes

	Action
1.	If necessary, use an overhead crane, fork lift or similar to secure the robots arms.
2.	Make sure the robot is powered.
3.	Once more, make sure that damage to entangled objects is not extended when brakes are released.
4.	Press the appropriate brake release button to release the brake.

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.

1.4.4. Recover from emergency stops

Overview

Recovering from an emergency stop is a simple but important procedure. This procedure ensures that the robot 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

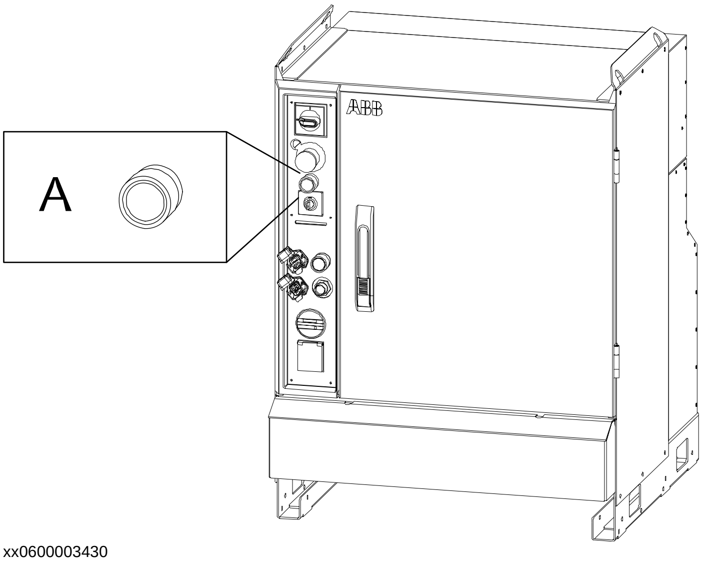
All automatic emergency stop devices also have some kind of latching feature that must be released. Please consult your plant or cell documentation to see how your robot system is configured.

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 to recover from the emergency stop condition.

The Motors On button

The Motors On button is located on the controller. On a Dual Controller the Motors On button is located on the Control Module. If your robot system uses another type of control cabinet, then the Motors On button may look different than the illustration below.



xx0600003430

A	Motors On button

1.4.5. Return to the programmed path

Overview

Turning off the power to the robot motors often results in the robot slipping from its programmed path. This may occur after an uncontrolled emergency or safety stop. The allowed slip distance is configured with system parameters. The distance can be different depending on operating mode.

If the robot is not within the configured allowed distance, you may choose to let the robot return to the programmed path or continue to the next programmed point in the path. Then the program execution continues automatically in programmed speed.

For more information see *Technical reference manual - System parameters*, section *Topic Controller - Type Path Return Region*.

1.5 Working in a safe manner

1.5.1. Overview

About the robot

A robot is heavy and extremely powerful regardless of its speed. A pause or longer stop in movement can be followed by a fast hazardous movement. Even if a pattern of movement is predicted, a change in operation can be triggered by an external signal resulting in an unexpected movement without warning.

Therefore, it is important that all safety regulations are followed when entering safeguarded space.

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.

1.5.2. For your own safety

General principles

A few simple principles should be followed in order to operate the robot system safely:

- Always operate the robot system in manual mode if personnel are inside safeguarded space.
- Always bring the FlexPendant along when you enter safeguarded space so that robot control is in your hands.
- Watch out for rotating or moving tools such as milling cutters and saws. Make sure those are stopped before you approach the robot.
- Watch out for hot surfaces both on work pieces as well as on the robot system. The robot's motors can become fairly hot if run for a long time.
- Watch out for grippers and objects gripped. If the gripper is opened the work piece could fall and cause injuries or damage equipment. The gripper can be very powerful and can also cause injuries if not operated in a safe manner.
- Watch out for hydraulic and pneumatic systems and live electric parts. Even with power off residual energy in such circuits can be very dangerous.

Disconnected FlexPendant

Always put away a disconnected FlexPendant safe from any robot cell or controller to avoid that a disconnected unit is used when trying to stop the robot in a hazardous situation.

CAUTION!

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

Custom FlexPendant connections

Any means of connecting the FlexPendant except by 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.

1.5.3. CAUTION - Handling of FlexPendant

General instructions

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 may cause breakage or failure.
- When not using the device, hang it on the wall bracket provided for storage so that it cannot fall to the ground by accident, or that nobody can trip over the cable.
- Never use sharp objects (e.g. screwdriver or pen) for operating the touch screen. This could damage the touch screen. Instead try a stylus, normally used for a PDA.
- Never clean the device with solvents, scouring agent, or scrubbing sponges. For cleaning the device, use a soft cloth and a bit of water or mild cleaning agent. See *Product manual - IRC5*, section *Cleaning the FlexPendant*.

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

CAUTION!

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

Waste disposal

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

Foreseeable misuse of enabling device

Foreseeable misuse means that it is not allowed to jam the enabling device in the enabling position. The foreseeable misuse of the enabling device must be restricted.

When releasing and then pressing the enabling device again, make sure to wait for the system to go to Motors Off state before pressing again. Otherwise you will receive an error message.

1.5.4. Safety tools

Safeguarding mechanisms

Your robot system can be equipped with a vast range of safeguards such as door interlocks, safety light curtains, safety mats, and others. The most common is the door interlock of the robot cell that temporarily stops the robot if you open it.

The controller has three separate safeguarding mechanisms, the *general mode safeguarded stop* (GS), the *automatic mode safeguarded stop* (AS) and the *superior safeguarded stop* (SS).

Safeguards connected to...	are...
the GS mechanism	always active regardless of the operating mode.
the AS mechanism	only active when the system is in automatic mode.
the SS mechanism	always active regardless of the operating mode.

Please consult your plant or cell documentation to see how your robot system is configured and where the safeguarding mechanisms are placed and how they work.

Safety supervision

The emergency stop and safeguarding mechanisms are supervised so that any failure is detected by the controller and the robot is stopped until the problem is solved.

Built-in safety stop functions

The controller continuously monitor hardware and software functionality. If any problems or errors are detected the robot is stopped until the problem has been solved.

If the failure is...	then...
simple and can easily be solved	a simple program stop is issued (SYSSTOP).
minor and can be solved	a SYSHALT is issued which results in a safety stop.
major, for instance concerns broken hardware	a SYSFAIL is issued which results in an emergency stop. The controller must be restarted in order to return to normal operation.

Restricting the robot's working range

The robot's working range can be restricted by means of mechanical stops or software functions, or by a combination of both.

Please consult your plant or cell documentation to see how your robot system is configured.

1.5.5. Safety in manual reduced speed and manual full speed mode

What is the manual mode?

The manual mode allows program creation, storage, and testing of robot paths and positions.

The manual mode is used when programming and when commissioning a robot system.

There are two manual modes:

- Manual reduced speed mode, usually called manual mode.
- Manual full speed mode (not available in all markets).

In manual mode, you need to press the enabling device to activate the robot’s motors.

What is the manual full speed mode?

In manual full speed mode the robot system can run in full speed. This mode is used when testing programs.

Operating speed

In manual reduced speed mode the robot can only be operated (moved) in reduced speed, 250 mm/s or slower. You should always operate in manual speed whenever working inside safeguarded space.

In manual full speed mode the robot moves in programmed speed. The manual full speed mode should only be used while all personnel are outside safeguarded space and only by specifically trained personnel extra aware of the implied risks.

Bypassed safeguard mechanisms

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

The enabling device

In **manual mode** the robot’s motors are activated by the enabling device on the FlexPendant. This way the robot can only move as long as the device is pressed.

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

The enabling device is designed so that you must press its push-button just half-way to activate the robot’s motors. Both in its all-out and full-in positions the robot will not move.

The hold-to-run function

The hold-to-run function allows stepping or running a program in manual full speed mode.

Note that jogging does not require the hold-to-run function, regardless of operating mode.

1.5.6. Safety in automatic mode

What is the automatic mode?

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

In automatic mode the enabling device is disconnected so that the robot can move without human intervention.

Active safeguard mechanisms

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

Coping with process disturbances

Process disturbances may not only affect a specific robot 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 robot cell. All remedial actions must be performed by personnel with good knowledge of the entire production line, not only the malfunctioning robot.

Process disturbance examples

A robot 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 robot in close proximity to the running conveyor.

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

2 Welcome to FlexPendant

2.1. About this chapter

Overview

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

A basic IRC5 robot system consists of a robot controller, the FlexPendant, RobotStudio, and one or several robots or other mechanical units. There may also be process equipment and additional software options.

This manual describes a basic IRC5 system without options. However, in a few places, the manual gives an overview of how options are used or applied. Most options are described in detail in their respective Application manual.

2.2. What is a FlexPendant?

Overview

The FlexPendant (occasionally called TPU or teach pendant unit) is a hand held operator unit used to perform many of the tasks involved when operating a robot system: running programs, jogging the manipulator, modifying robot programs etcetera.

NOTE!

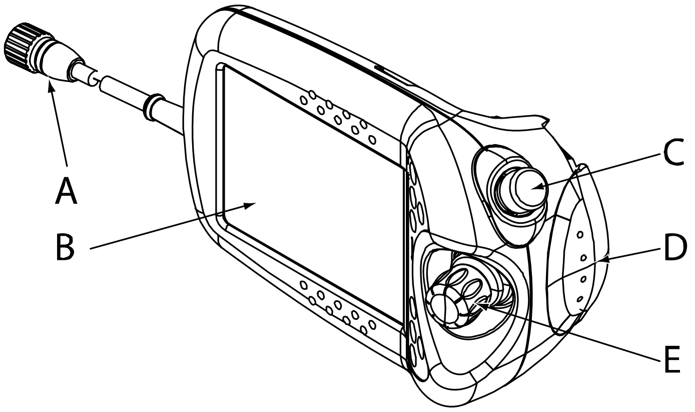
Note that a few specific functions, such as administer the user authority system (UAS), cannot be performed using the FlexPendant, but only by using RobotStudio. Use *Operating manual - RobotStudio* for details in these cases.

Complete computer and integral part of IRC5

The FlexPendant consists of both hardware and software and is a complete computer in itself. It is an integral part of IRC5, connected to the controller by an integrated cable and connector. The hot plug button option, however, makes it possible to disconnect the FlexPendant in automatic mode and continue running without it. See *Using the hot plug option on page 235* in *Operating manual - IRC5 with FlexPendant* for details.

Main parts

The FlexPendant is designed for continuous operation in harsh industrial environment. Its touch screen is easy to clean and resistant to water, oil and accidental welding splashes. These are the main parts of the FlexPendant.



en0300000586

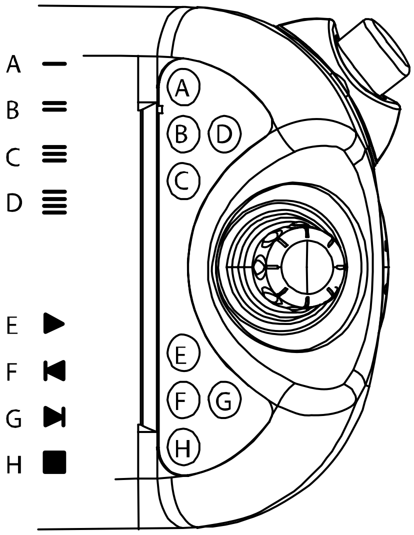
A	Connector
B	Touch screen
C	Emergency stop button
D	Enabling device
E	Joystick

Continues on next page

Continued

Hard buttons

There are eight dedicated hardware buttons on the FlexPendant, four of which the function is assigned by the end user.



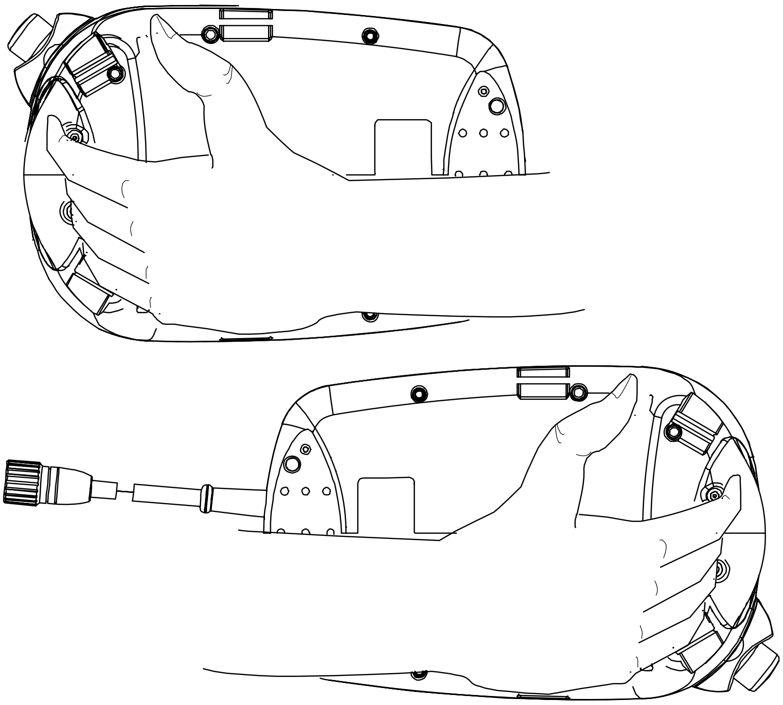
en0300000587

A - D	Programmable keys, 1 - 4. How to define their respective function is detailed in section *Changing programmable keys on page 314* in *Operating manual - IRC5 with FlexPendant*.
E	START button. Starts program execution.
F	Step BACKWARD button. Executes one instruction backward as button is pressed.
G	Step FORWARD button. Executes one instruction forward as button is pressed.
H	STOP button. Stops program execution.

Continues on next page

How to hold the FlexPendant

The FlexPendant is typically operated while being held in the hand. A right-handed person uses his left hand to support the device while the other hand performs operations on the touch screen. A left-hander, however, can easily rotate the display through 180 degrees and use his right hand to support the device. See *Adapting the FlexPendant for left-handed users on page 309* in *Operating manual - IRC5 with FlexPendant* for further information.



en0400000913

Operated in fourteen languages

As suggested by its name, the FlexPendant is designed with flexibility and adaptation to end-users' specific needs in mind. Currently, it can be operated in 14 different languages, including Asian character-based languages such as Chinese and Japanese.

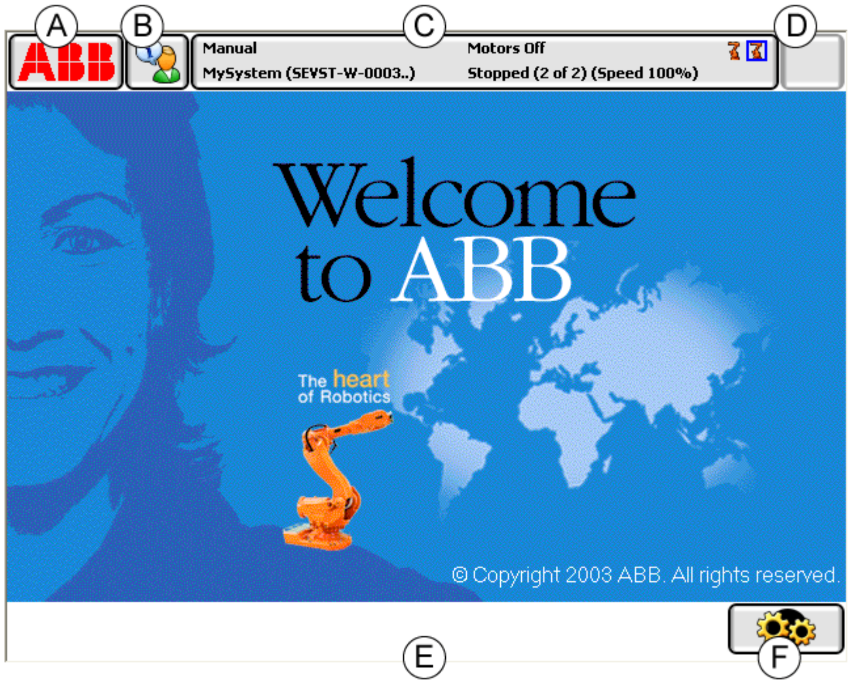
The individual FlexPendant supports up to three languages, selected before the installation of the system to the robot controller. Switching from one of the installed languages to another is easy. See *Changing language on page 313* in *Operating manual - IRC5 with FlexPendant* for information about how to do it.

Continues on next page

Continued

Touch screen elements

The illustration shows important elements of the FlexPendant touch screen.



Manual
MySystem (SEVST-W-0003..)

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

en0300000588

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

ABB menu

From the ABB menu the following items can be selected

- HotEdit
- Inputs and Outputs
- Jogging
- Production Window
- Program Editor
- Program Data
- Backup and Restore
- Calibration
- Control Panel
- Event Log
- FlexPendant Explorer
- System Info
- etc.

This is further detailed in section *The ABB menu on page 75*.

Continues on next page

Continued

Operator window

The operator window displays messages from robot programs. This usually happens when the program needs some kind of operator response in order to continue. This is described in section *Operator window on page 95*.

Status bar

The status bar displays important information about system status, such as operating mode, motors on/off, program state and so on. This is described in section *Status bar on page 96*.

Close button

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

Task bar

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

Quickset menu

The quickset menu provides settings for jogging and program execution. This is described in section *The Quickset menu on page 97*.

Handling and cleaning

- Handle with care. Do not drop, throw, or give the FlexPendant strong shock. It may cause breakage or failure.
- When not using the device, hang it on the wall bracket provided for storage so that it cannot fall to the ground by accident, or that nobody can trip over the cable.
- Never use sharp objects (e.g. screwdriver or pen) for operating the touch screen. This could damage the touch screen. Instead try a stylus, normally used for a PDA.
- Never clean the device with solvents, scouring agent, or scrubbing sponges. For cleaning the device, use a soft cloth and a bit of water or mild cleaning agent. See *Product manual - IRC5*, section *Cleaning the FlexPendant*.

2.3. What is an IRC5 controller?

The IRC5 controller

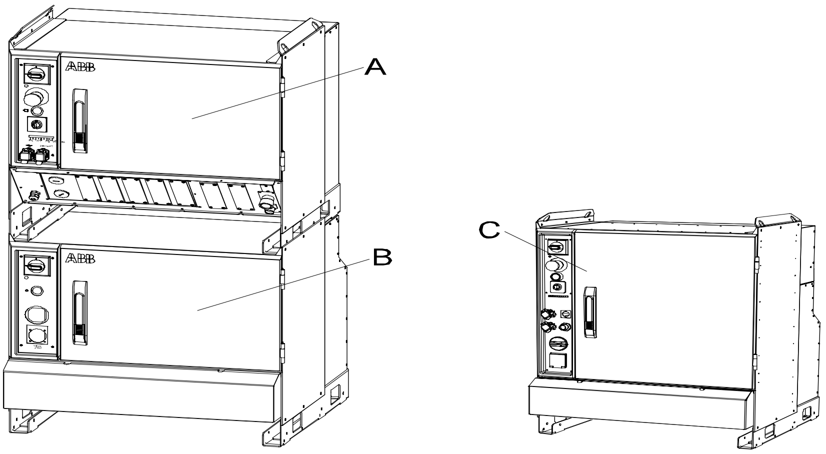
The IRC5 controller contains all functions needed to move and control the robot.

The standard IRC5 controller consists of a single cabinet. As an option, it can also be divided into two modules; the Control Module and the Drive Module. This is called a Dual Cabinet Controller.

The Control Module contains all the control electronics such as main computer, I/O boards, and flash memory.

The Drive Module contains all the power electronics supplying the robot motors. An IRC5 Drive Module may contain nine drive units and handle six internal axes plus two or additional axes depending on the robot model.

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



xx0500002046

A	Control Module, Dual Cabinet Controller
B	Drive Module, Dual Cabinet Controller
C	Single Cabinet Controller

Related information

Product manual - IRC5.

Application manual - MultiMove.

2.4. What is RobotStudio?

Overview

RobotStudio is a computer application for the offline creation, programming, and simulation of robot cells.

RobotStudio is available in full, customized, and minimal installation. The minimal installation is used for working in online mode on the controller, as a complement to the FlexPendant. The full (and customized) installation offers advanced programming and simulation tools.

RobotStudio online mode functionality

RobotStudio online mode is optimized for:

1. Creating, installing, and maintaining systems, using the System Builder.Text-based programing and editing, using the Program Editor.
2. File manager for the controller.
3. Administrating the User Authorization System.

2.5. When to use the FlexPendant and RobotStudio

Overview

For operating and managing the robot, you either use the FlexPendant or RobotStudio. The FlexPendant is optimized for handling robot motions and ordinary operation, and RobotStudio is optimized for configuration, programming and other tasks not related to the daily operation.

Start, restart and shut down the controller

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

Run and control robot programs

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

Communicate with the controller

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

Continues on next page

Program a robot

To...	Use...
Create or edit robot programs in a flexible way. This is suitable for complex programs with a lot of logic, I/O signals or action instructions.	**RobotStudio** to create the program's structure and most of the source code and the **FlexPendant** to store robot positions and make final adjustments to the program. When programming, RobotStudio provides the following advantages: • A text editor optimized for RAPID code, with auto-text and tool-tip information about instructions and parameters. • Program check with program error marking. • Close access to configuration and I/O editing.
Create or edit a robot program in a supportive way. This is suitable for programs that mostly consist of move instructions.	The **FlexPendant**. When programming, the FlexPendant provides the following advantages: • Instruction pick lists • Program check and debug while writing • Possibility to create robot positions while programming
Add or edit robot positions.	The **FlexPendant**.
Modify robot positions.	The **FlexPendant**.

Configure the robot's system parameters

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

Create, modify and install systems

To...	Use...
Create or modify a system.	**RobotStudio** together with **RobotWare** and a valid **RobotWare Key**.
Install a system on a controller.	**RobotStudio**
Install a system on a controller from a USB memory.	The **FlexPendant**.

Calibration

To...	Use...
Calibrate base frame etc.	The **FlexPendant**
Calibrate tools, work objects etc.	The **FlexPendant**

Continues on next page

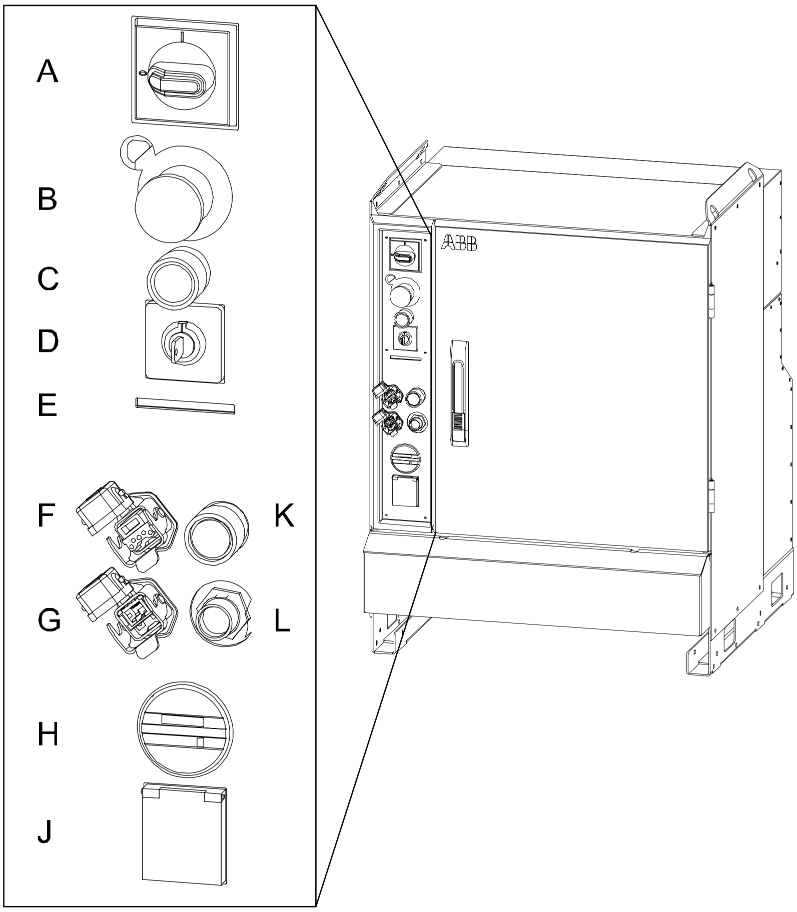
Related information

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

Recommended use...	for details, see manual...	Document number
FlexPendant	Operating manual - IRC5 with FlexPendant	3HAC16590-1
RobotStudio	Operating manual - RobotStudio	3HAC032104-001

2.6. Buttons and ports on the controller

Single Cabinet Controller buttons and ports

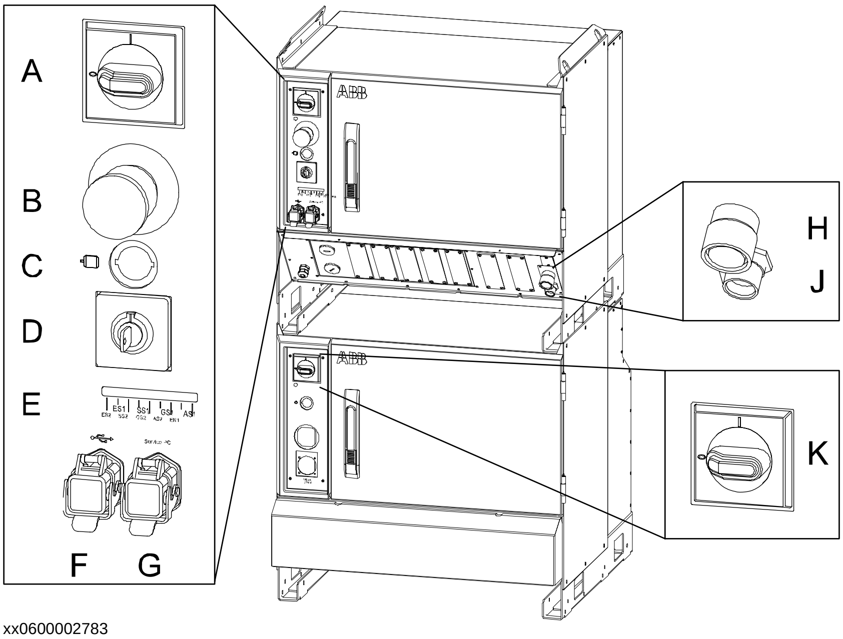


xx0600002782

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

Continues on next page

Dual Cabinet Controller buttons and ports



xx0600002783

A	Main switch, control module
B	Emergency stop
C	Motors on
D	Mode switch
E	Safety chain LEDs (option)
F	USB port (option)
G	Service port for PC (option)
H	Hot plug button (option)
J	FlexPendant connector
K	Main switch, drive module

Related information

Product manual - IRC5.

Operating manual - Trouble shooting.

3 Get started

3.1. About this chapter

Overview

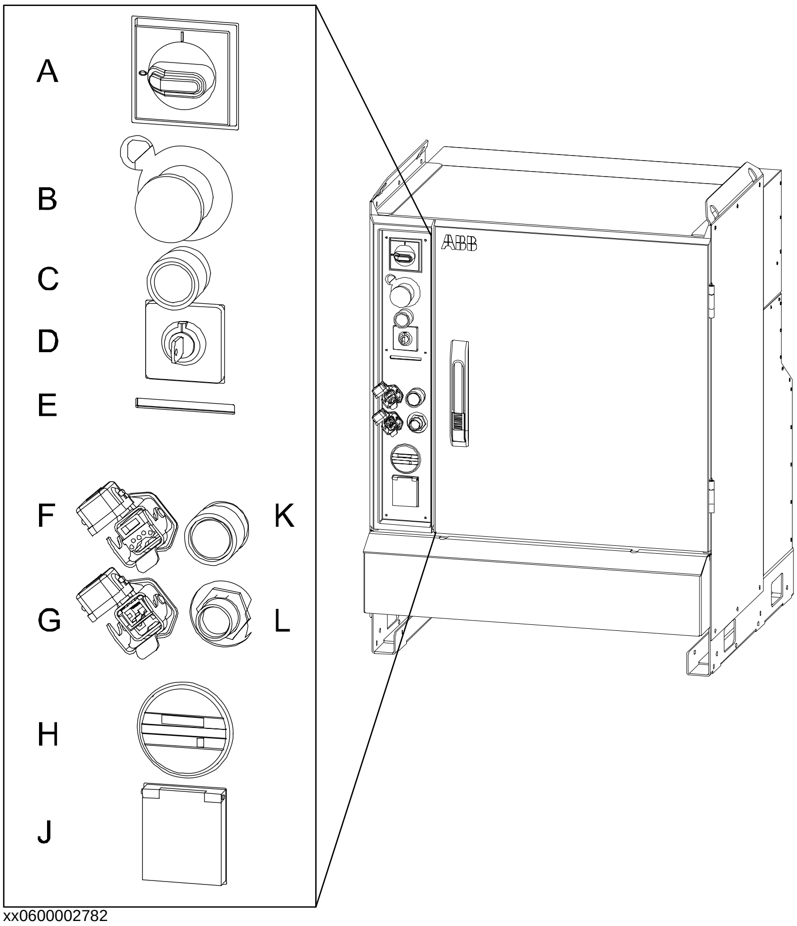
This chapter describes how to connect the FlexPendant to the controller and how to set up network connections. It also presents a number of often performed work tasks with the FlexPendant, described as action scenarios.

3.2 Connections

3.2.1. Connecting a FlexPendant

Location of FlexPendant connector

The FlexPendant connector is located as shown below.



xx0600002782

L	FlexPendant connector (A22.X1)

On a Dual Cabinet Controller, the FlexPendant connector is located on the front of the Control Module.

Connecting a FlexPendant

	Action	Info
1.	Locate the FlexPendant socket connector on the controller.	The controller must be in manual mode. If your system has the option Hot plug, then you can also disconnect in auto mode. See section *Using the hot plug option on page 235*.
2.	Plug in the FlexPendant cable connector.	
3.	Screw the connector lock ring firmly by turning it clockwise.	

3.2.2. Disconnecting a FlexPendant

Disconnecting a FlexPendant

This procedure details how to disconnect a FlexPendant

	Action
1.	Finish any ongoing activities that require the FlexPendant to be connected. (For instance path adjustments, calibration, program changes.)
2.	Shut down the system. If the system is not shut down when disconnecting the FlexPendant it will go to the emergency stop state.
3.	Unscrew the connector cable counter clockwise.
4.	Store the FlexPendant safely away from any robot system.

3.2.3. Set up the network connection

When do I need to setup the network connection?

You need to setup the controller's network connection when the controller is connected to a network for the first time or when the network addressing scheme changes.

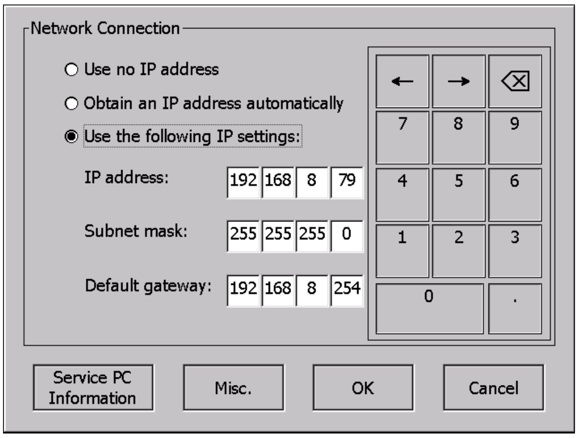
Preparations

If an IP address is to be obtained automatically, make sure there is a server running that supplies the network with IP addresses (a DHCP server). Otherwise you will not be able to access the controller via the controller network.

It is still possible to access the controller via the service PC connection.

Network connection dialog box

The illustration shows the network connection dialog box.



Network Connection

○ Use no IP address

○ Obtain an IP address automatically

● Use the following IP settings:

IP address: 192 168 8 79

Subnet mask: 255 255 255 0

Default gateway: 192 168 8 254

en0400000902

Set up the network connection

Regardless of how you choose to set up the network connections, the first steps are common:

	Action	Info
1.	Perform an X-start to start the Boot Application.	How to perform an X-start is detailed in section *Restart and select another system (X-start) on page 270*.
2.	In the Boot Application, tap **Settings**. The network connection dialog is displayed.	
3.	If you choose to use no IP address, then tap **Use no IP address**. Otherwise, proceed below!	In some cases it can be useful to disconnect the controller from the network, without disconnecting the network cable. Without IP address the controller cannot be accessed from other equipment on the same network.
4.	If you choose to obtain an IP address automatically, then tap **Obtain an IP address automatically**. Otherwise, proceed below!	

Continues on next page

Continued

	Action	Info
5.	If you choose to use a fixed IP address, tap **Use the following IP address**. Enter the IP address, subnet mask, and default gateway.	**NOTE!** Make sure a valid address is used so there are no conflicts in the network. A conflict may cause other controllers to malfunction
6.	Tap **OK** to save the new setting.	
7.	In the Boot Application, tap **Restart Controller** to restart the controller and use the new setting.	

3.3 Action scenarios

3.3.1. About action scenarios

Overview

This chapter presents brief procedures, detailing a number of typical actions a typical user may perform. It also includes references to detailed information about the same topics.

The brief information given, is intended to be used directly by experienced users, while the references may be more adequate for novices and for training purposes.

Related information

Note that there may be more information available than the one referred to in the procedures.

Information about:

- a specific menu is described in chapter *Navigating and handling FlexPendant on page 73*.
- a specific button on the FlexPendant is described in *What is a FlexPendant? on page 40*.
- a specific button is described in chapter *What is an IRC5 controller? on page 45* for tasks performed using the controls on the controller cabinet.
- how to perform a specific task is detailed in the tasks chapters, e.g. *Programming and testing on page 133* or *Running in production on page 227*.

Related information can also be found in other manuals:

- *Operating manual - RobotStudio*
- *Product manual - IRC5*
- *Operating manual - Trouble shooting*

3.3.2. System start up

Prerequisites before start up

This procedure details the main steps required to start the system when the power has been switched off.

All information is based on the assumption that *working system software has already been installed* on the robot controller, as the case would be at first start-up directly after delivery.

Note that there may be more information available than the one referred to in the procedure.

System start up

This procedure details all required steps to start the system for the first time. For everyday startup, step 4 is normally the only required step.

	Action	Info
1.	Install the robot equipment.	Mechanical installation and electrical connections between manipulator and controller is described in the *Product manual* of the robot and controller respectively.
2.	Make sure the safety circuits of the system are properly connected to the robot cell or have jumper connections installed (if required).	How to connect the safety circuits is detailed in the robot's *Product manual*.
3.	Connect the FlexPendant to the controller.	The FlexPendant and its major parts and functions are detailed in section *What is a FlexPendant? on page 40* How to connect the FlexPendant to the controller is detailed in section *Connecting a FlexPendant on page 54*
4.	Switch the power on.	Use the main switch on the controller.
5.	If the controller or manipulator have been replaced with spare parts, make sure the calibration values, revolution counters and serial numbers are updated correctly.	Normally, only the revolution counters require updating, which is to be performed as detailed in section *Updating revolution counters on page 288*. If required, transfer the calibration data from the serial measurement board as detailed in *Serial Measurement Board memory on page 294* for systems *without* the Absolute Accuracy option. If required, enter the calibration data as detailed in *Loading calibration data using the FlexPendant on page 290* for systems *with* the Absolute Accuracy option.
6.	This step is only required if the robot system will be connected to a network. Perform an *X-start*. The Boot Application is started.	Detailed in section *Restart and select another system (X-start) on page 270*.

Continues on next page

Continued

	Action	Info
7.	This step is only required if the robot system will be connected to a network. Use the Boot Application to: • set the IP address of the controller cabinet • set the network connections • select the system • restart the system The system is restarted.	How to use the Boot Application is detailed in section *Using the Boot Application on page 266*. At this point, a single system is available.
8.	Install RobotStudio on a PC.	Proceed as detailed in *Operating manual - RobotStudio*. RobotStudio is used to create a system to run on the controller, but at this point (prior to the first start-up) a system is already installed by the manufacturer.
9.	Connect the controller to a PC (through the service port) or to the network (if used).	Proceed as detailed in *Product manual - IRC5*, section *Connecting a PC to the service port*. Also see section *Set up the network connection on page 56*.
10.	Start RobotStudio on the PC.	Proceed as detailed in *Operating manual - RobotStudio*.
11.	Restart the controller.	
12.	The robot system is now ready for operation.	

3.3.3. Jogging

Jogging

This procedure details the main steps required to jog the robot.

The term Jogging is described in section *Introduction to jogging on page 105*.

Note that there may be more information available than the one referred to in the procedure.

	Action	Info
1.	It is possible to jog the robot under the following conditions: • The system has been started as detailed in this manual. • No programmed operation is running • The system is in Manual mode. • The enabling device is pressed and the system is in Motors On state.	The Manual mode is described in section *About the manual mode on page 242*. Starting in the Manual mode is detailed in section *Starting programs on page 227*. How to switch to manual mode is detailed in section *Switching from automatic to manual mode on page 245*.
2.	Many of the mechanical units connected to the controller may be jogged.	How to determine which mechanical unit to jog is detailed in section *Selecting mechanical unit for jogging on page 108*.
3.	The robot may be jogged in several ways, in different coordinate systems. First, determine in which way you want to jog.	The difference between different types of jogging is detailed in section *Introduction to jogging on page 105*. How to jog the robot axis by axis is detailed in section *Jog axis by axis on page 114*. The robot may be jogged in: • *Jog in base coordinates on page 116* • *Jog in tool coordinates on page 120* • *Jog in world coordinates on page 117* • *Jog in work object coordinates on page 119*
4.	Once a mechanical unit has been selected, its axes may be jogged in different ways. These ways may be selected using the QuickSet menu.	
5.	Define the working range for the robot/ robots as well as for any other pieces of equipment working in the robot cell.	The robot's working range is defined by system parameters. See section *Configuring system parameters on page 283* or *Technical reference manual - System parameters*.

Continues on next page

Continued

	Action	Info
6.	Jog the manipulator using the joystick on the FlexPendant.	The FlexPendant and its various parts and sections are described in section *What is a FlexPendant? on page 40*. The joystick and how to map the directions of it, is detailed in section *Selecting motion mode on page 110*. How to prevent causing manipulator movements in certain directions while jogging, is detailed in section *Locking the joystick in specific directions on page 122*. There might be restrictions to how you can jog, see section *Restrictions to jogging on page 106*.
7.	In some cases, more than one manipulator may be jogged simultaneously. This requires the MultiMove option to be installed.	How to jog multiple manipulators is detailed in section *Coordinated jogging on page 107*.

3.3.4. Using RAPID programs

Using the RAPID program

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

Note that there is more information available, than the one referred to in the procedure. The concept RAPID program is described in section *The structure of a RAPID application on page 134*.

	Action	Info
1.	Start by creating a RAPID program.	How to create a RAPID program is detailed in section *Handling of programs on page 166*.
2.	Edit your program.	Proceed as detailed in section *Handling of instructions on page 178*.
3.	To simplify programming and keep an overview of the program, you may want to divide the program into more than one module.	The module concept is described in section *The structure of a RAPID application on page 134*. How to view, add, or delete a module is detailed in section *Handling of modules on page 169*.
4.	To further simplify programming, you may want to divide the module into more than one routine.	The routine concept is described in section *The structure of a RAPID application on page 134*. How to add or delete a routine is detailed in section *Handling of routines on page 173*.
5.	When programming you may want to work with: • Tools • Work objects • Payloads	Also read the following sections: • *Creating a tool on page 144*. • *Creating a work object on page 155*. • *Creating a payload on page 162*.
6.	In order to deal with potential errors that may occur during program execution, you may want to create an error handler.	Error handlers are described in the *RAPID* manuals.
7.	After completing the actual RAPID program, it will require testing before being put into production.	Proceed as detailed in section *Testing on page 205*.
8.	After test running your RAPID program, it may require altering. You may want to modify, or tune, programmed positions, the TCP positions, or paths.	How to modify positions while the program is running is described in section *HotEdit menu on page 75*. How to modify positions in manual mode is described in section *Modifying positions in the Program Editor or Production Window on page 187*.
9.	Programs that are no longer required may be removed.	See *Deleting programs from memory on page 201*. Also see *Deleting programs from hard disk on page 203*.

Continues on next page

Running the program

This procedure specifies how to use an existing RAPID program.

	Action	Info
1.	Load an existing program.	Described in section *Starting programs on page 227*.
2.	When starting program execution, you may choose between running the program once, or running it continuously.	Described in section *Quickset menu, Run Mode on page 208*.
3.	Once the program has been loaded, you may start program execution.	Described in section *Starting programs on page 227* and in *Using multitasking programs on page 231*.
4.	After program execution is completed, the program may be stopped.	Proceed as detailed in section *Stopping programs on page 230*.

3.3.5. Working with inputs and outputs

Working with inputs and outputs

This procedure details the main steps required to set outputs, read inputs and configure I/O units.

Note that there may be more information available than the one referred to in the procedure.

	Action	Info
1.	You can create a new I/O.	I/O signals are created using system parameters, see section *Configuring system parameters on page 283*.
2.	Before using any input or output, the system must be configured to enable the I/O functions.	Configuring the system is done when creating the system. How to do this is detailed in *Operating manual - RobotStudio*.
3.	You can set a value to a specific *digital output*.	Proceed as detailed in section *Simulating and changing signal values on page 248*.
4.	You can set a value to a specific *analog output*.	Proceed as detailed in section *Simulating and changing signal values on page 248*.
5.	You can view the status of a specific *digital input*.	Proceed as detailed in section *Simulating and changing signal values on page 248*.
6.	You can view the status of a specific *analog input*.	Proceed as detailed in section *Simulating and changing signal values on page 248*.
7.	Safety signals.	Signal explanation is detailed in *Safety I/O signals on page 252*
8.	How to edit an I/O.	Proceed as detailed in section*Simulating and changing signal values on page 248*.

3.3.6. Backup and restore

Backup and restore

The contents of a typical backup is specified in section *What is saved on backup? on page 277*. How to perform the backup is detailed in section *Back up the system on page 279*.

Re-introducing the previously saved memory contents from the backup into the robot controller is called *performing a restore*. How to perform the restore is detailed in section *Restore the system on page 280*.

Information about starts is described in *Restart overview on page 265*.

Note that there may be more information available than the one referred to above.

3.3.7. Running in production

Running in production

This instruction details the main steps useful when running the system in automatic mode (production mode).

Note that there may be more information available than the one referred to in the procedure.

	Action	Info
1.	Start the system as detailed in section *System start up on page 59*.	
2.	If the system is using UAS, User Authorization System, the user must log into the system before starting operation.	How to log in is described in section *Logging on and off on page 104*.
3.	Load a program.	How to load a program is described in *Handling of programs on page 166*.
4.	Before starting system choose mode to start in on the controller.	How to choose mode is described in section *Switching from manual to automatic mode on page 243*.
5.	Start by pressing the Start button on the FlexPendant.	The FlexPendant's hardware buttons are described in *What is a FlexPendant? on page 40*.
6.	The controller system communicates with the operator through messages displayed on the FlexPendant screen. Messages can be either event messages or RAPID instructions, e.g. `TPWrite`. Event messages describe an event occurring within the system, and is saved in an event log.	The basic concepts are described in section *Accessing the event log on page 255*. RAPID instructions `TPReadFK` and `TPWrite` are described in *Technical reference manual - RAPID Instructions, Functions and Data types*.
7.	In manual mode, the Modify Position function allows the operator to make adjustments to the robot positions in a RAPID program. The HotEdit function allows the operator to make adjustments to programmed positions in both automatic and manual mode.	How to modify position is described in sections *Modifying positions in the Program Editor or Production Window on page 187* and *HotEdit menu on page 75*.
8.	In a production process you may want to stop the robot.	How to stop production is described in section *Stopping programs on page 230*.
9.	In the Production Window you can supervise the ongoing process.	The Production window is described in section *Production Window on page 81*.
10.	When ending operation, the user should log off.	How to log in is described in section *Logging on and off on page 104*.

3.3.8. Granting access for RobotStudio

About write access on the controller

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

Granting access for RobotStudio

This procedure describes how to grant access for RobotStudio.

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

3.3.9. Upgrading

Upgrading

This procedure details the main steps required to correctly upgrade the system. By upgrading we mean changing hardware, such as replacing circuit board with newer versions, as well as loading software with later releases.

Note that there may be more information available than the one referred to in the procedure.

Type of upgrade	Info
When replacing circuit boards such as buses, I/O boards, etc., with newer versions, the system will automatically reflash the unit. ! xx0100000003 During reflashing, the system may restart several times, and it is vital not to shut down the system, or in any other way interrupt the automatic process.	What happens during reflashing is detailed in section *Reflashing firmware and FlexPendant on page 275*.
When upgrading the robot or controller mechanically, fitting instructions are normally delivered with the kit. If no such instruction are provided, useful information may be found in the Repair section of the *Product Manual* of the equipment in question.	
When upgrading the system software, the system must be changed in order to reflect the additions. A new license key may be required.	How to modify an existing system is detailed in section *How to Modify a System* in the *Operating manual - RobotStudio*. How to create a new system is detailed in section *Creating a new system* in the *Operating manual - RobotStudio*.

3.3.10. Installing software options

Installing software options

The main steps required to correctly install a generic software option or option package is described in *Operating manual - RobotStudio*.

3.3.11. Shutting down

Shutting down

This procedure describes how to shut down the system and turn off power.

	Action	Info
1.	Stop all running programs.	
2.	Shut down the system using the main power switch (On/Off switch) or shut down the system using the FlexPendant, tap **ABB menu** - **Restart** - **Advanced** - **Shutdown**. When using the FlexPendant, wait for 30 seconds after shutting down the system. It is then safe to turn off the main power switch.	The FlexPendant will display " Connecting to controller ...". This can safely be ignored.
3.	If you want to protect the FlexPendant you can unplug it and store it elsewhere when the system has shut down.	How to disconnect the FlexPendant from the controller is detailed in section *Disconnecting a FlexPendant on page 55*.

4 Navigating and handling FlexPendant

4.1. About this chapter

Overview

This chapter will help you to work efficiently with the FlexPendant. The important elements for navigation illustrated in *Touch screen elements on page 43* will now be further detailed.

All views of the ABB menu, the main element for navigation, are described in overview with references to further details on how to use their functions. Note that this manual only covers views of a basic RobotWare system. Process applications such as arc welding, dispense, or plastics are started from the ABB menu but not described in this manual, as all software options are detailed in their respective application manual.

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

4.2. Overview, personalizing the FlexPendant

Personalizing

The FlexPendant can be personalized in a number of ways. How to do this is described in the following sections:

How to:	is described in section:
change the language used in windows and dialogs	*Changing language on page 313*.
change the display's brightness and contrast	*Changing brightness and contrast on page 308*.
rotate the FlexPendant for Left/Right handheld use	*Procedure on page 309*
configure views for program start	*Defining a view to be shown at operating mode change on page 301*.
recalibrate the touch screen	*Calibrating the touch screen on page 316*.
configure programmable keys	*Changing programmable keys on page 314*.
configure most common I/O list	*Configuring Most Common I/O on page 312*.
change background image	*Changing the background image on page 302*.
change the date and time	*Changing date and time on page 311*.

4.3 The ABB menu

4.3.1. HotEdit menu

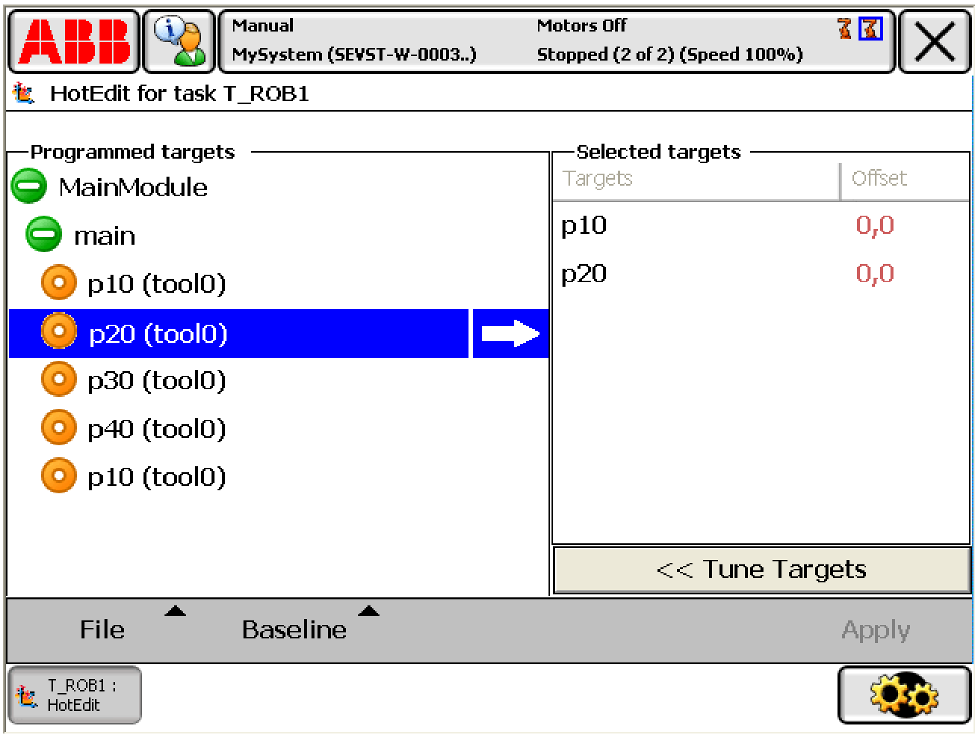
HotEdit

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

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

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

Illustration of the HotEdit view



en0500001542

Functions available in HotEdit

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

Continues on next page

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

CAUTION!

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

Before you start using the HotEdit functionality it is strongly recommended to read *Tuning positions with HotEdit on page 190*, where HotEdit limitations and procedures as well as the baseline concept is detailed.

Related information

See section *Modifying and tuning positions on page 186* for a general overview on how to modify programmed positions.

For modifying positions by jogging the robot to the new position, see section *Modifying positions in the Program Editor or Production Window on page 187*.

For detailed information about HotEdit, see *Tuning positions with HotEdit on page 190*.

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

Technical reference manual - System parameters, section *Topic Controller - Type ModPos Settings*.

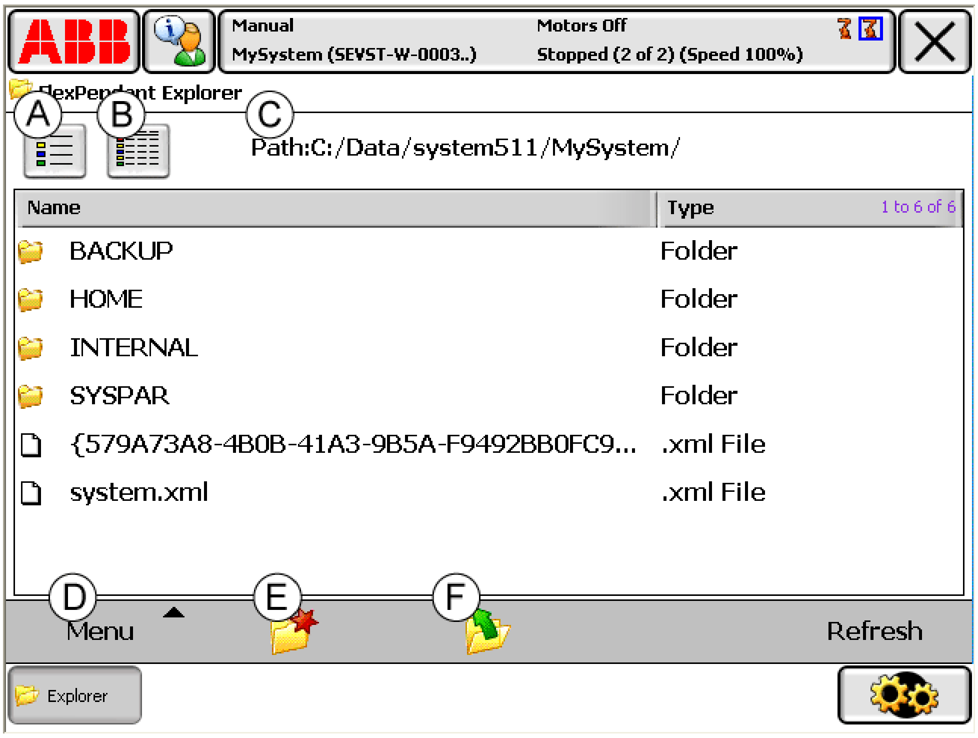
4.3.2. FlexPendant Explorer

FlexPendant Explorer

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

Illustration FlexPendant Explorer

The illustration details the FlexPendant Explorer.



Path:C:/Data/system511/MySystem/

Name	Type
BACKUP	Folder
HOME	Folder
INTERNAL	Folder
SYSPAR	Folder
{579A73A8-4B0B-41A3-9B5A-F9492BB0FC9...	.xml File
system.xml	.xml File

1 to 6 of 6

en0400001130

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

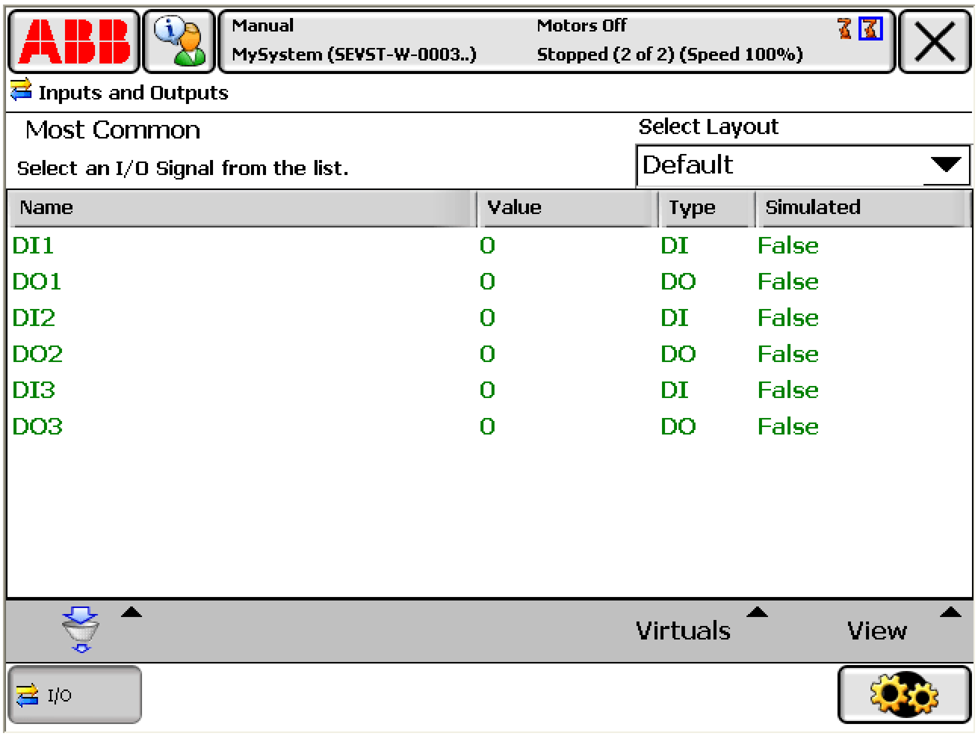
4.3.3. Inputs and Outputs, I/O

Inputs and outputs

Inputs and outputs, I/O, are signals used in the robot system. Signals are configured with system parameters, see section *Configuring system parameters on page 283*.

Illustration Inputs and Outputs view

This illustration details the Inputs and Outputs view.



Manual
MySystem (SEVST-W-0003..)
Motors Off
Stopped (2 of 2) (Speed 100%)
Inputs and Outputs
Most Common
Select an I/O Signal from the list.
Select Layout
Default

Name	Value	Type	Simulated
DI1	0	DI	False
DO1	0	DO	False
DI2	0	DI	False
DO2	0	DO	False
DI3	0	DI	False
DO3	0	DO	False

en04000000770

What is a signal

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

- Real I/O signal located on a fieldbus I/O unit that is connected to a fieldbus within the robot system.
- Virtual I/O signal i.e. an I/O signal not located on any fieldbus I/O unit.

By specifying an I/O signal, a logical representation of the real or virtual I/O signal is created. The I/O signal configuration defines the specific system parameters for the I/O signal that will control the behavior of the signal.

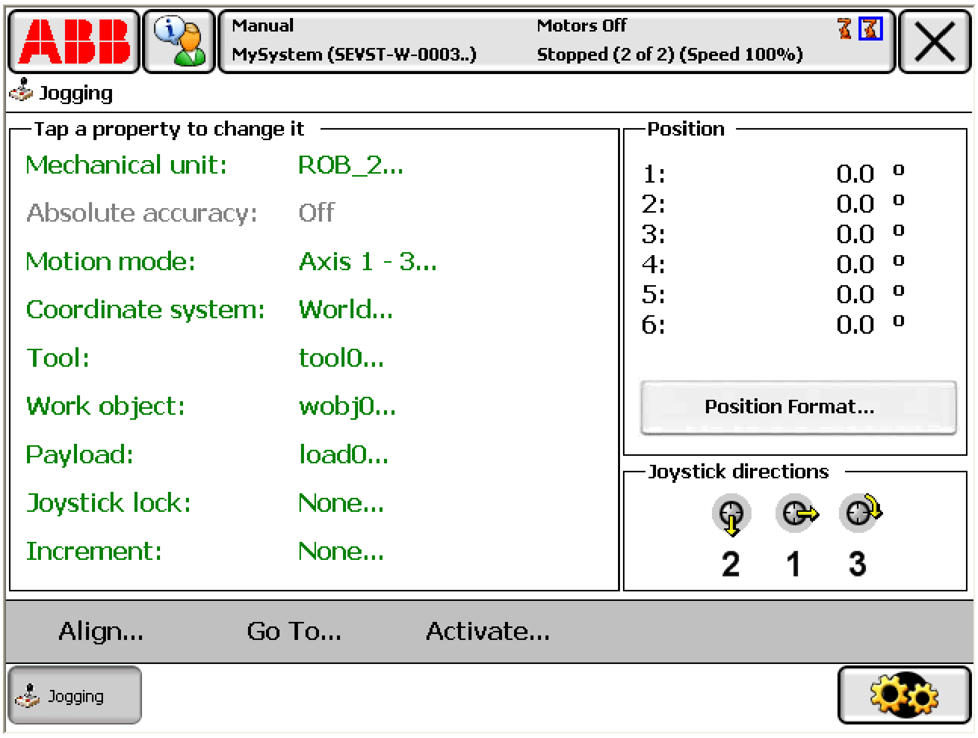
4.3.4. Jogging

Overview

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

Jogging menu

The illustration shows the functions available under the Jogging menu:



Manual MySystem (SEVST-W-0003..)

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

Jogging

Tap a property to change it

Mechanical unit: ROB_2...

Absolute accuracy: Off

Motion mode: Axis 1 - 3...

Coordinate system: World...

Tool: tool0...

Work object: wobj0...

Payload: load0...

Joystick lock: None...

Increment: None...

Position

1: 0.0 º
2: 0.0 º
3: 0.0 º
4: 0.0 º
5: 0.0 º
6: 0.0 º

en0400000654

Property/button	Function
Mechanical unit	Select mechanical unit active for jogging, described in section *Selecting mechanical unit for jogging on page 108*.
Absolute accuracy	**Absolute Accuracy: Off** is default. If the robot has the *Absolute Accuracy* option, then **Absolute Accuracy: On** is displayed.
Motion mode	Select motion mode, described in section *Selecting motion mode on page 110*.
Coordinate system	Select coordinate system, e.g. described in section *Jog in base coordinates on page 116*.
Tool	Select tool, described in section *Selecting tool, work object, and payload on page 112*.
Work object	Select work object, described in section *Selecting tool, work object, and payload on page 112*.
Payload	Select payload, described in section *Selecting tool, work object, and payload on page 112*.
Joystick lock	Select locking joystick directions, described in section *Locking the joystick in specific directions on page 122*.
Increment	Select movement increments, described in section *Incremental movement for precise positioning on page 124*.

Continues on next page

Continued

Property/button	Function
Position	Displays each axis position in relation to the selected coordinate system, described in section *Reading the exact position on page 126*. If the position values are displayed in red, then the revolution counters must be updated. See section *Updating revolution counters on page 288*.
Position format	Select position format, described in section *Reading the exact position on page 126*.
Joystick directions	Displays current joystick directions, depending on setting in Motion mode. See section *Selecting motion mode on page 110*.
Align	Align the current tool to a coordinate system. See section *Aligning tools on page 196*.
Go To	Move the robot to a selected position/target. See section *Moving the robot to a programmed position on page 195*.
Activate	Activate a mechanical unit. See section *Activating mechanical units on page 204*.

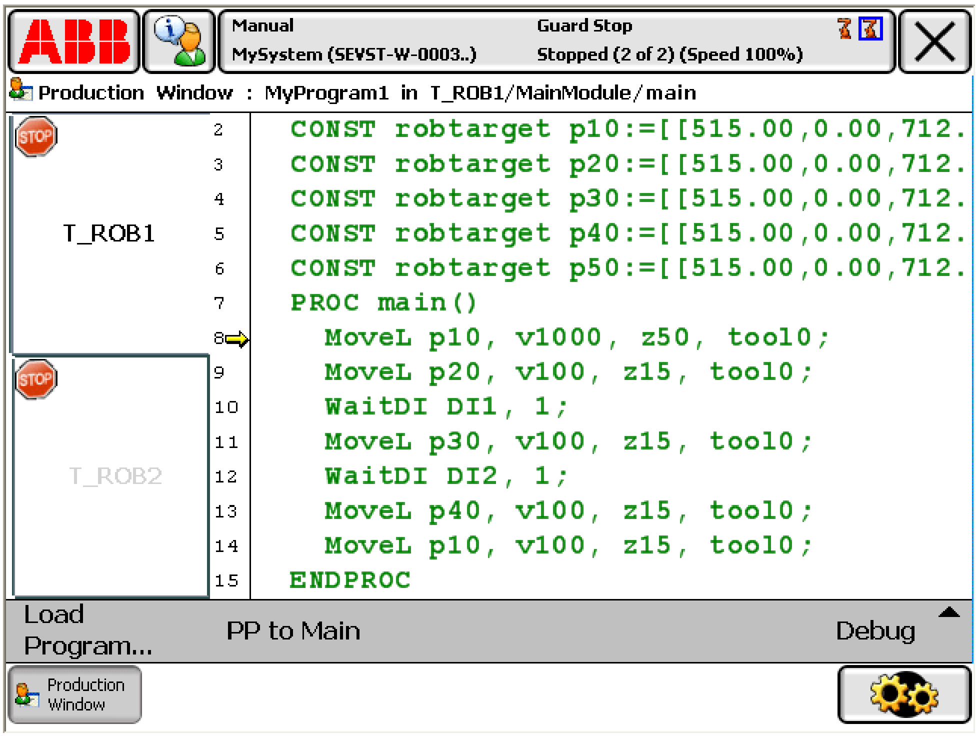
4.3.5. Production Window

Overview

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

Illustration Production Window

This section illustrates the Production Window.


Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)
Production Window : MyProgram1 in T_ROB1/MainModule/main
STOP
T_ROB1
STOP
T_ROB2
Load Program...
PP to Main
Debug
Production Window

```
CONST robtarget p10:=[[515.00,0.00,712.
CONST robtarget p20:=[[515.00,0.00,712.
CONST robtarget p30:=[[515.00,0.00,712.
CONST robtarget p40:=[[515.00,0.00,712.
CONST robtarget p50:=[[515.00,0.00,712.
PROC main()
  MoveL p10, v1000, z50, tool0;
  MoveL p20, v100, z15, tool0;
  WaitDI DI1, 1;
  MoveL p30, v100, z15, tool0;
  WaitDI DI2, 1;
  MoveL p40, v100, z15, tool0;
  MoveL p10, v100, z15, tool0;
ENDPROC
```

en0400000955

Load Program	Load a new program.
PP to Main	Move the program pointer to the routine Main.
Debug	The Debug menu is only available in manual mode. **Modify Position**, see *Modifying positions in the Program Editor or Production Window on page 187*. **Show Motion Pointer**and **Show Program Pointer**, see *About the Program and Motion Pointers on page 136*. **Edit Program**, see *Program Editor on page 84*.

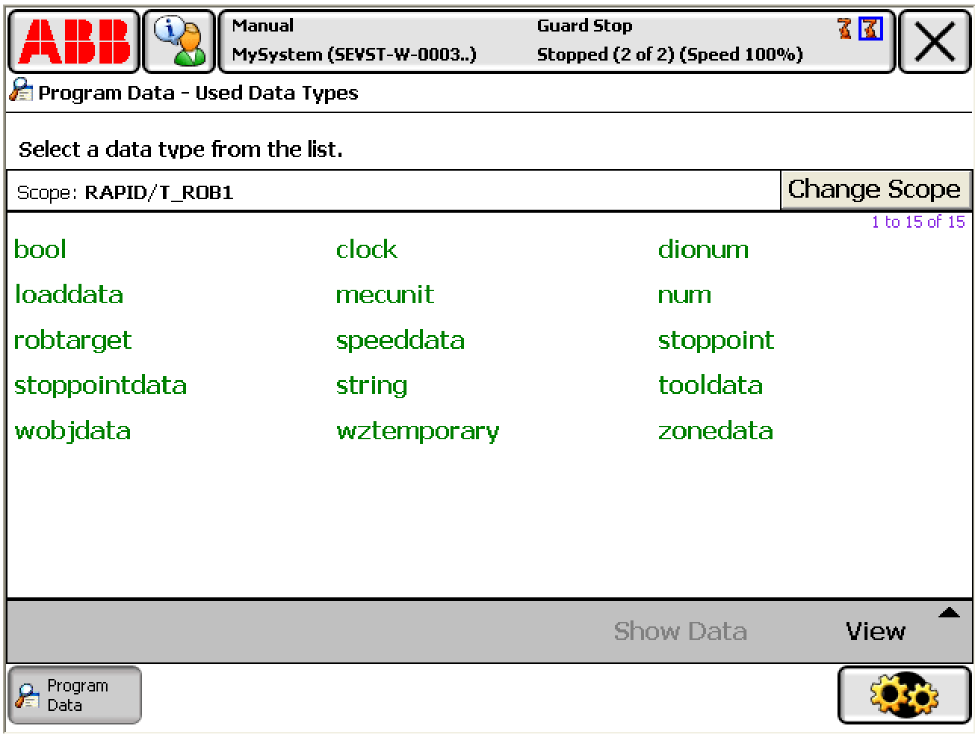
4.3.6. Program Data

Overview

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

Illustration of Program Data

This section illustrates the Program Data view.



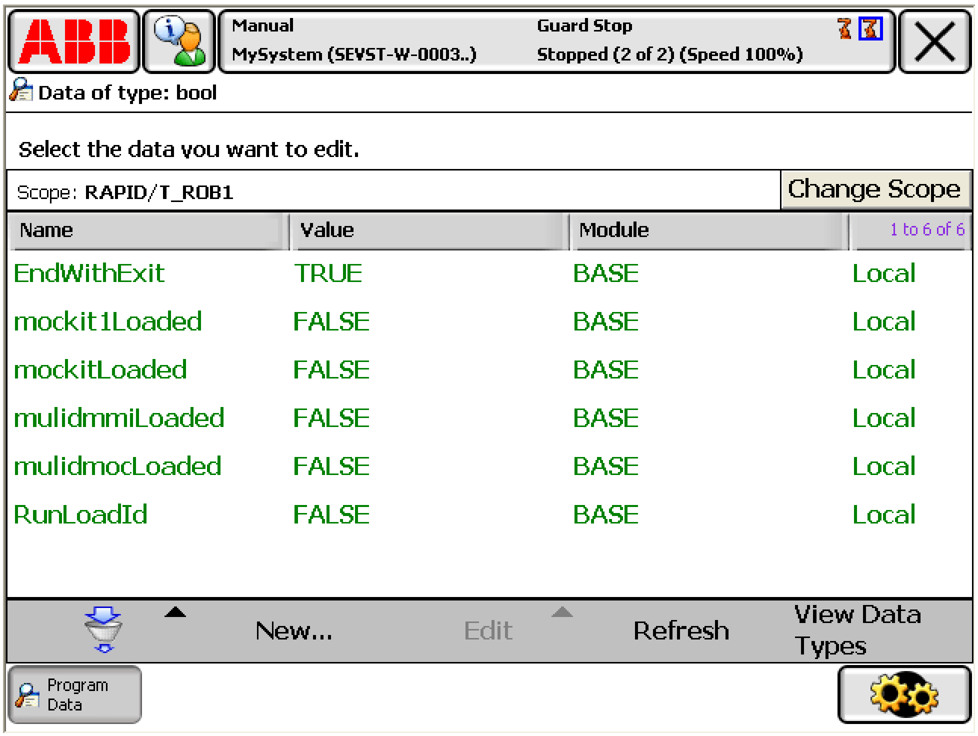
en0400000659

Change Scope	Changes scope of data types in the list, see *Viewing data in specific tasks, modules, or routines on page 137*.
Show Data	Shows all instances of the selected data type.
View	Shows all or only used data types.

Continues on next page

Illustration of a data type instances

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



Select the data you want to edit.

Scope: RAPID/T_ROB1

Name	Value	Module	1 to 6 of 6
EndWithExit	TRUE	BASE	Local
mockit1Loaded	FALSE	BASE	Local
mockitLoaded	FALSE	BASE	Local
mulidmmiLoaded	FALSE	BASE	Local
mulidmocLoaded	FALSE	BASE	Local
RunLoadId	FALSE	BASE	Local

en0500001571

Filter	Filters the instances, see *Filtering data on page 101*.
New	Creates a new instance of the selected data type, see *Creating new data instance on page 138*.
Refresh	Refreshes the list of instances.
Edit	Edits the selected instances, see *Editing data instances on page 140*.
View Data Types	Returns to the Program Data menu.

4.3.7. Program Editor

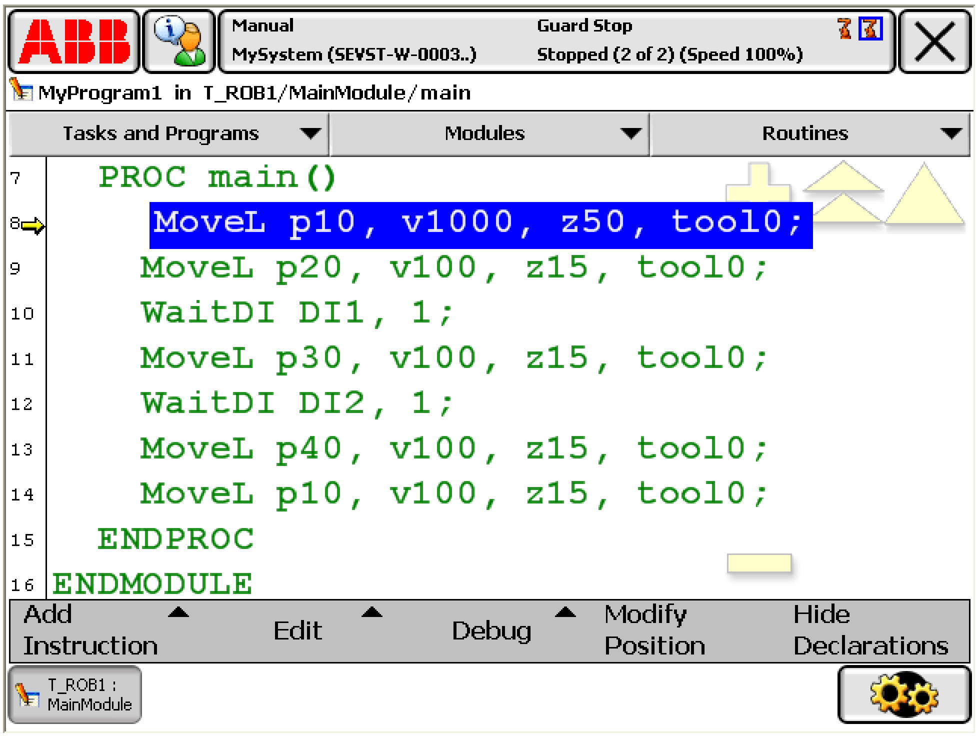
Overview

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

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

Illustration of Program Editor

This section illustrates the Program Editor view.



```
  PROC main()
    MoveL p10, v1000, z50, tool0;
    MoveL p20, v100, z15, tool0;
    WaitDI DI1, 1;
    MoveL p30, v100, z15, tool0;
    WaitDI DI2, 1;
    MoveL p40, v100, z15, tool0;
    MoveL p10, v100, z15, tool0;
  ENDPROC
ENDMODULE
```

en0400001143

Tasks and Programs	Menu for program operations, see *Handling of programs on page 166*.
Modules	Lists all modules, see *Handling of modules on page 169*.
Routines	Lists all routines, see *Handling of routines on page 173*.
Add Instruction	Opens instruction menu, see *Handling of instructions on page 178*.
Edit	Opens edit menu, see *Handling of instructions on page 178*.
Debug	Functions for moving the program pointer, service routines etc., see *Running a service routine on page 215*, and *About the Program and Motion Pointers on page 136*.
Modify Position	See *Modifying positions in the Program Editor or Production Window on page 187*.
Hide Declarations	See *Hiding declarations in program code on page 200*.

Continues on next page

Automatically activate mechanical unit for jogging

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

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

Mechanical units are manually activated for jogging in the Jogging window or in the Quickset menu, see *Selecting mechanical unit for jogging on page 108*.

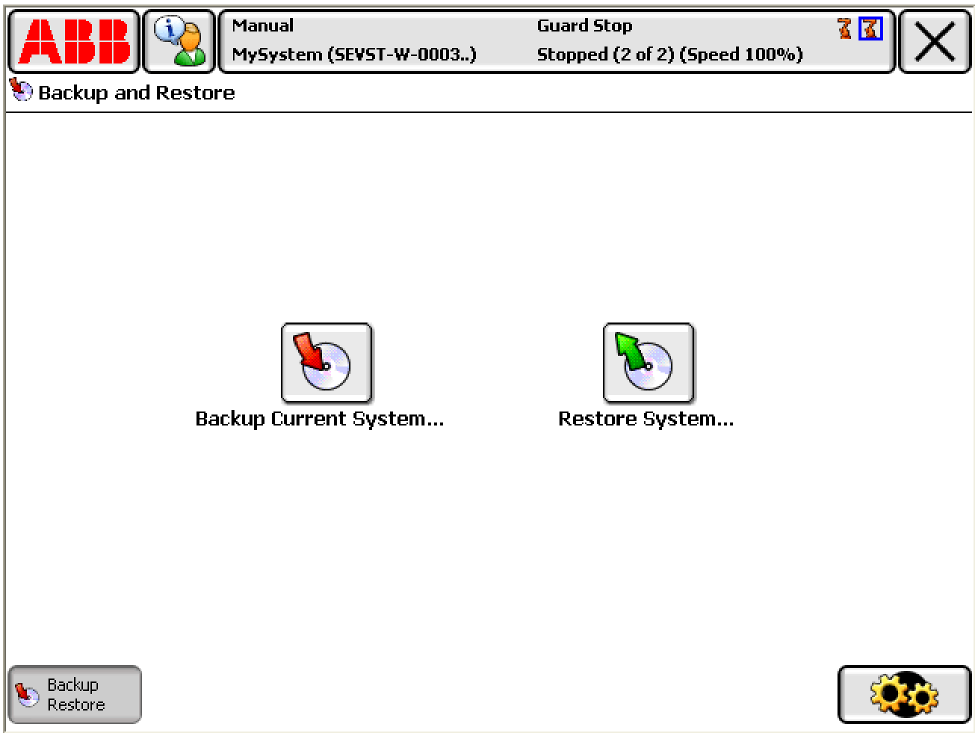
4.3.8. Backup and Restore

About backups

The Backup and Restore menu is used for performing backups and restoring the system. See section *Back up and restore systems on page 277*.

Illustration of Backup and Restore

This is the Backup and Restore menu.



Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)

Backup and Restore

xx0300000440

Backup Current System	See *Back up the system on page 279*.
Restore System	See *Restore the system on page 280*.

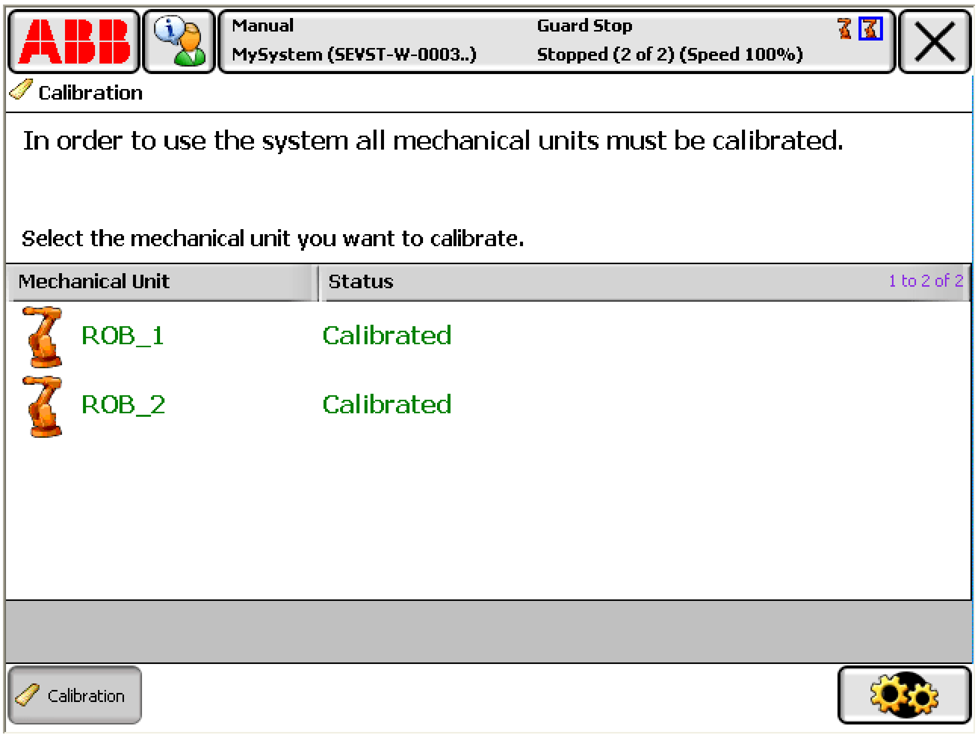
4.3.9. Calibration

About calibration

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

Illustration of Calibration menu

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



In order to use the system all mechanical units must be calibrated.

Select the mechanical unit you want to calibrate.

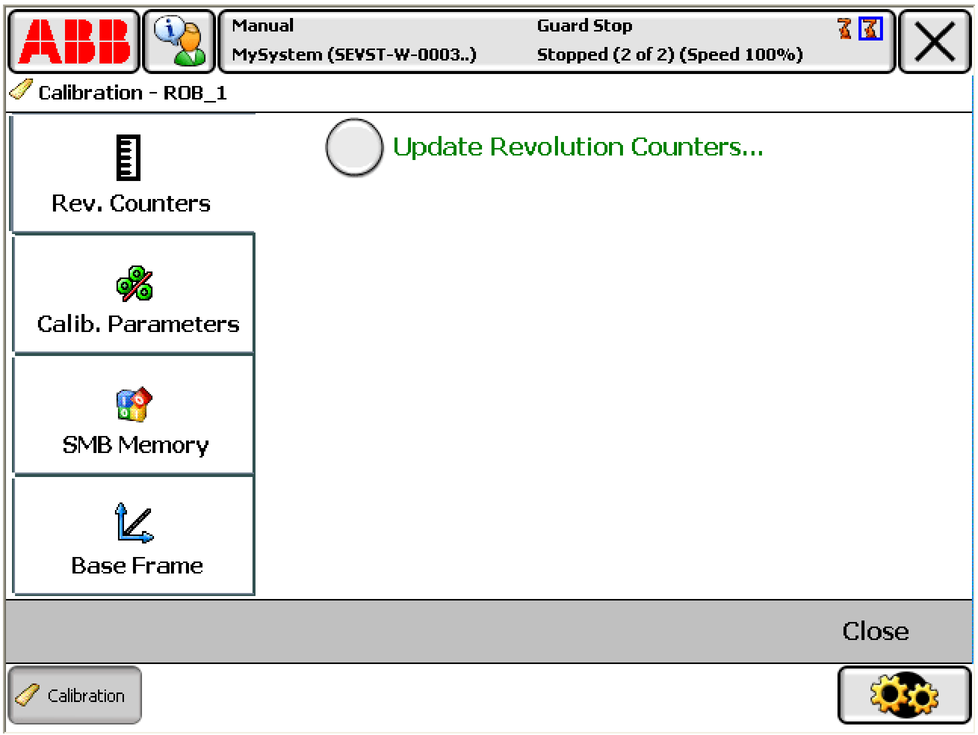
Mechanical Unit	Status
ROB_1	Calibrated
ROB_2	Calibrated

en0400001146

Continues on next page

Calibration menu options

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



en0400000771

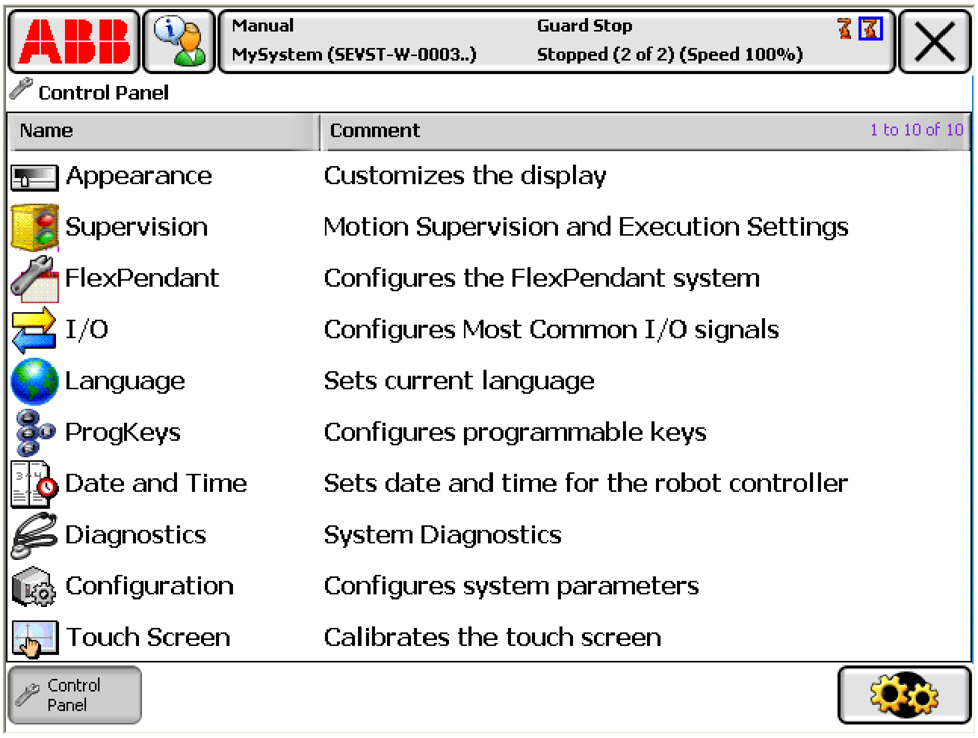
Revolution Counters	See section *Updating revolution counters on page 288*.
Calibration Parameters	See sections *Loading calibration data using the FlexPendant on page 290*, *Editing motor calibration offset on page 291*, and *Fine calibration procedure on FlexPendant on page 292*.
SMB Memory	See section *Serial Measurement Board memory on page 294*.
Base Frame	See section *4 points XZ calibration on page 297*.

4.3.10. Control Panel

Control Panel

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

Illustration Control Panel



Name	Comment
Appearance	Customizes the display
Supervision	Motion Supervision and Execution Settings
FlexPendant	Configures the FlexPendant system
I/O	Configures Most Common I/O signals
Language	Sets current language
ProgKeys	Configures programmable keys
Date and Time	Sets date and time for the robot controller
Diagnostics	System Diagnostics
Configuration	Configures system parameters
Touch Screen	Calibrates the touch screen

en0400000914

Appearance	Settings to customize the display's brightness and contrast. See *Changing brightness and contrast on page 308*.
Configuration	Configuration of the system parameters configuration. See *Configuring system parameters on page 283*.
Date and Time	Settings for date and time for the robot controller. See *Changing date and time on page 311*.
Diagnostic	Creates a diagnostic file, useful for trouble shooting. See *Create a diagnostic file on page 282*.
FlexPendant	Configuration of views for operating mode switch and UAS, User Authorization System. See *Defining a view to be shown at operating mode change on page 301*.
I/O	Settings for configuring the Most Common I/O list. See *Configuring Most Common I/O on page 312*.
Language	Settings for current language for the robot controller. See *Changing language on page 313*.
ProgKeys	Settings for the four programmable keys on the FlexPendant. See *Changing programmable keys on page 314*.
Supervision	Settings for motion supervision and execution settings. See *Using motion supervision and non motion execution on page 233*.
Touch Screen	Recalibration settings for the touch screen. See *Calibrating the touch screen on page 316*.

4.3.11. Event Log

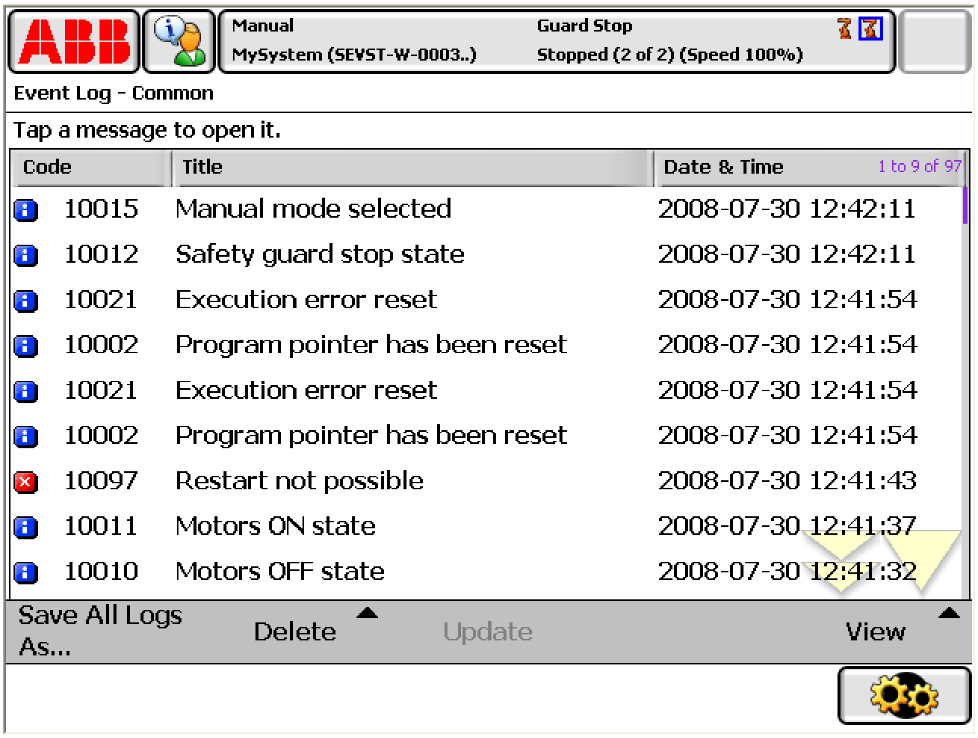
The Event Log

Robot systems are often operated without any personnel present. The logging function is a way to store information about past events for future reference in order to facilitate trouble shooting.

How to open the event log is described in *Accessing the event log on page 255*.

Illustration Event Log

The table is a brief summary of all actions that may be performed with the event log.



Event Log - Common

Tap a message to open it.

	Code	Title	Date & Time (1 to 9 of 97)
ⓘ	10015	Manual mode selected	2008-07-30 12:42:11
ⓘ	10012	Safety guard stop state	2008-07-30 12:42:11
ⓘ	10021	Execution error reset	2008-07-30 12:41:54
ⓘ	10002	Program pointer has been reset	2008-07-30 12:41:54
ⓘ	10021	Execution error reset	2008-07-30 12:41:54
ⓘ	10002	Program pointer has been reset	2008-07-30 12:41:54
☒	10097	Restart not possible	2008-07-30 12:41:43
ⓘ	10011	Motors ON state	2008-07-30 12:41:37
ⓘ	10010	Motors OFF state	2008-07-30 12:41:32

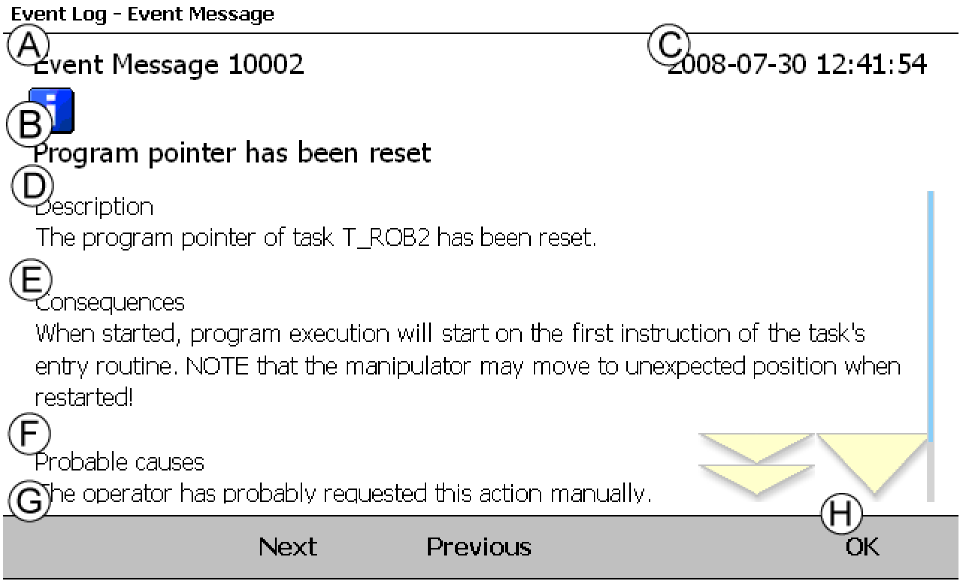
xx0300000447

Function	Description
View a message	Tap the message. The message structure is described in *An event log message on page 91*.
Scroll or zoom a message	See *Scrolling and zooming on page 100*.
Delete the log	See *Deleting log entries on page 256*.
Save the log	See *Saving log entries on page 257*.
Close the log	See *Accessing the event log on page 255*.

Continues on next page

An event log message

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



Event Log - Event Message

Event Message 10002

2008-07-30 12:41:54

Program pointer has been reset

Description

The program pointer of task T_ROB2 has been reset.

Consequences

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

Probable causes

The operator has probably requested this action manually.

en0300000454

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

Related information about logs

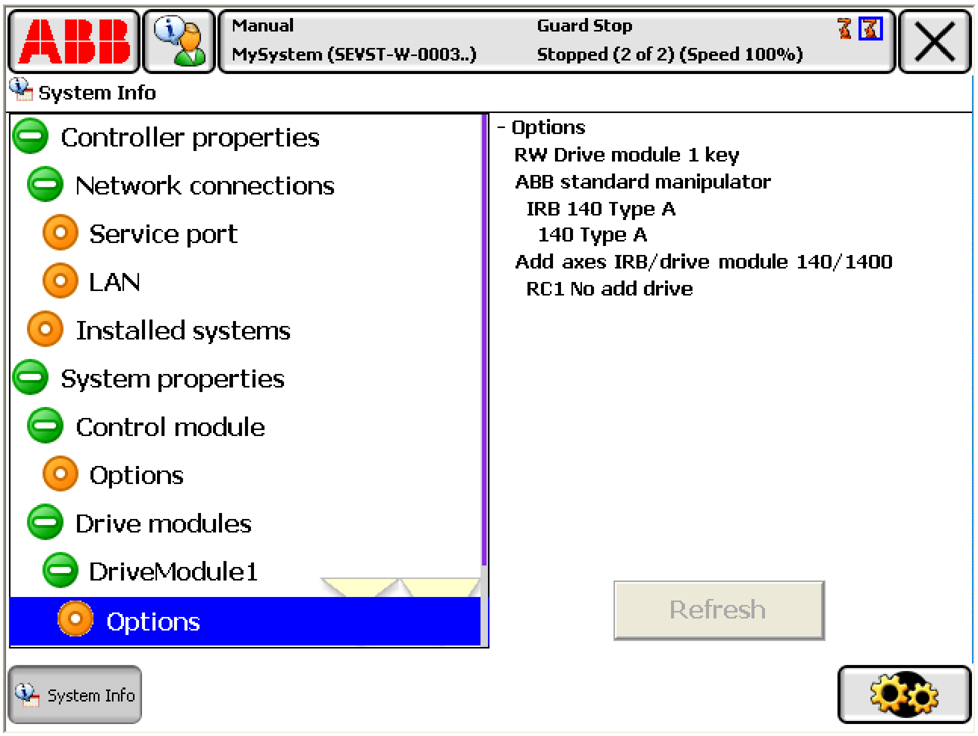
Event log messages and more information about the event log are described in *Operating manual - Trouble shooting*.

4.3.12. System Info

About System Info

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

Illustration of System Info view



- Options
RW Drive module 1 key
ABB standard manipulator
IRB 140 Type A
140 Type A
Add axes IRB/drive module 140/1400
RC1 No add drive

en0400000968

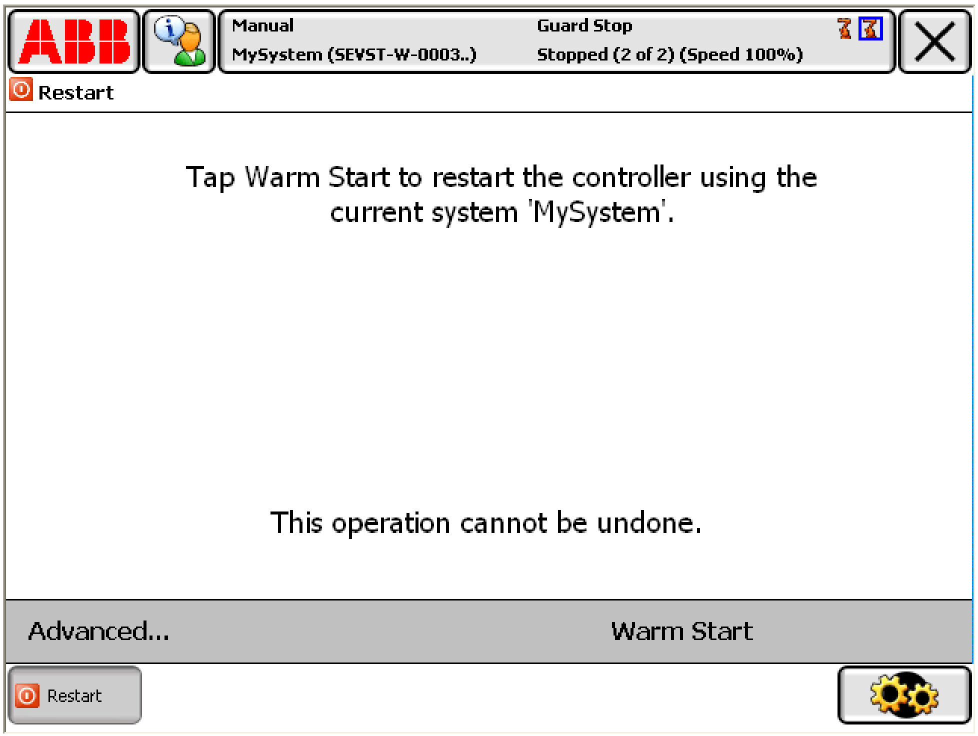
Controller properties	Name of the controller.
Network connections	Service port and Local Area Network properties.
Installed systems	List of installed systems.
System properties	Information about the system currently in use.
Control module	Name and key of the Control Module.
Options	Installed RobotWare options and languages.
Drive modules	Lists all Drive Modules.
Drive module x	Name and key of Drive Module x.
Options	Drive Module x options, with type of robot etc.
Additional options	Any additional installed options.

4.3.13. Restart

Restart

A running system normally does not need to be restarted.

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



Tap Warm Start to restart the controller using the current system 'MySystem'.

This operation cannot be undone.

en0500001557

Related information

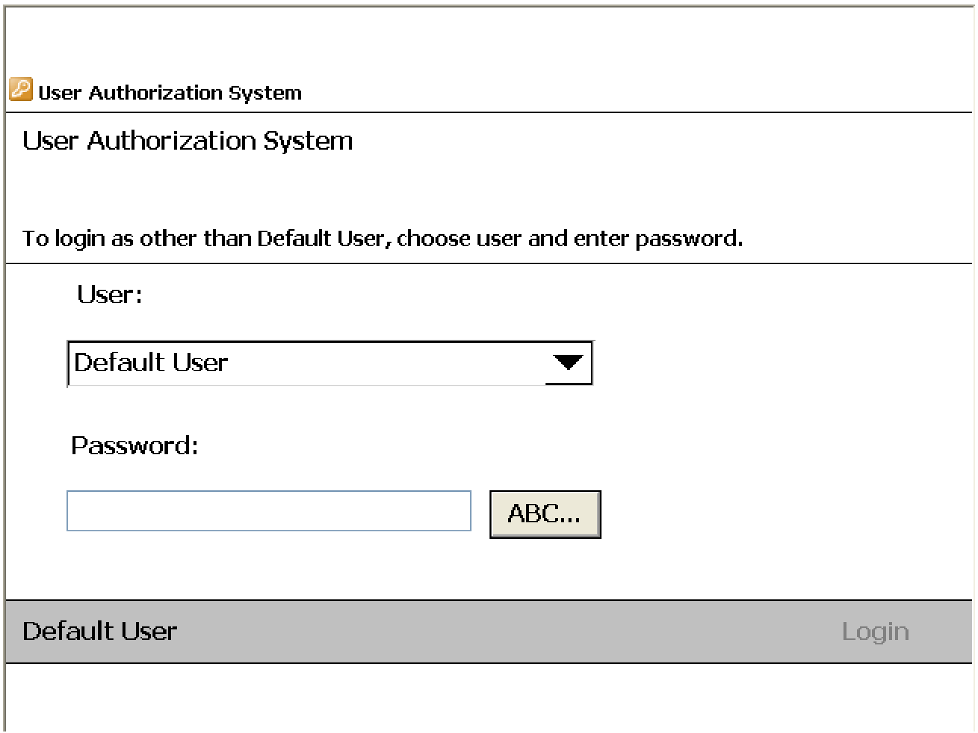
Restart overview on page 265.

4.3.14. Log Off

The Log Off menu

This section details the Log Off menu. More about using this menu is described in *Logging on and off on page 104*.

Log Off is available under the ABB menu.



User Authorization System

User Authorization System

To login as other than Default User, choose user and enter password.

User: Default User

Password:

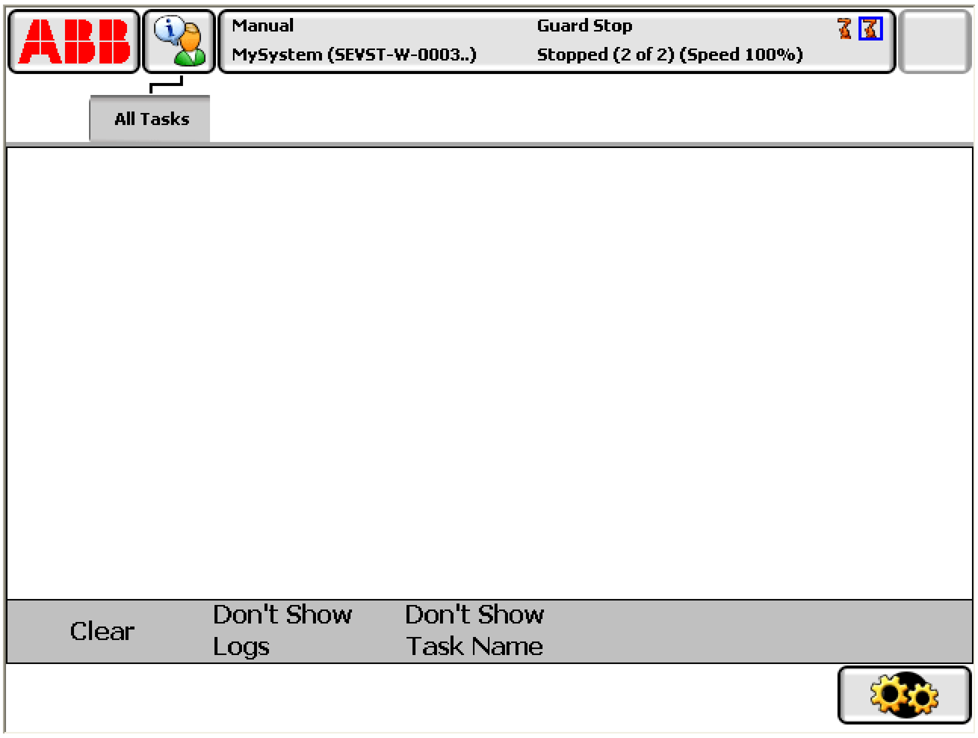
en0400000947

4.4. Operator window

Operator window

The operator window displays messages from the program. With *Multitasking* installed, all tasks’ messages are displayed in the same operator window. If a message requires action then a separate window for that task will be displayed.

The operator window is opened by tapping the icon to the right of the ABB logo in the status bar. The illustration shows an example of an operator window:



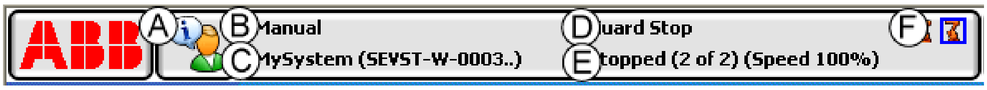
en0400000975

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

4.5. Status bar

Illustration of status bar

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



en0300000490

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

4.6. The Quickset menu

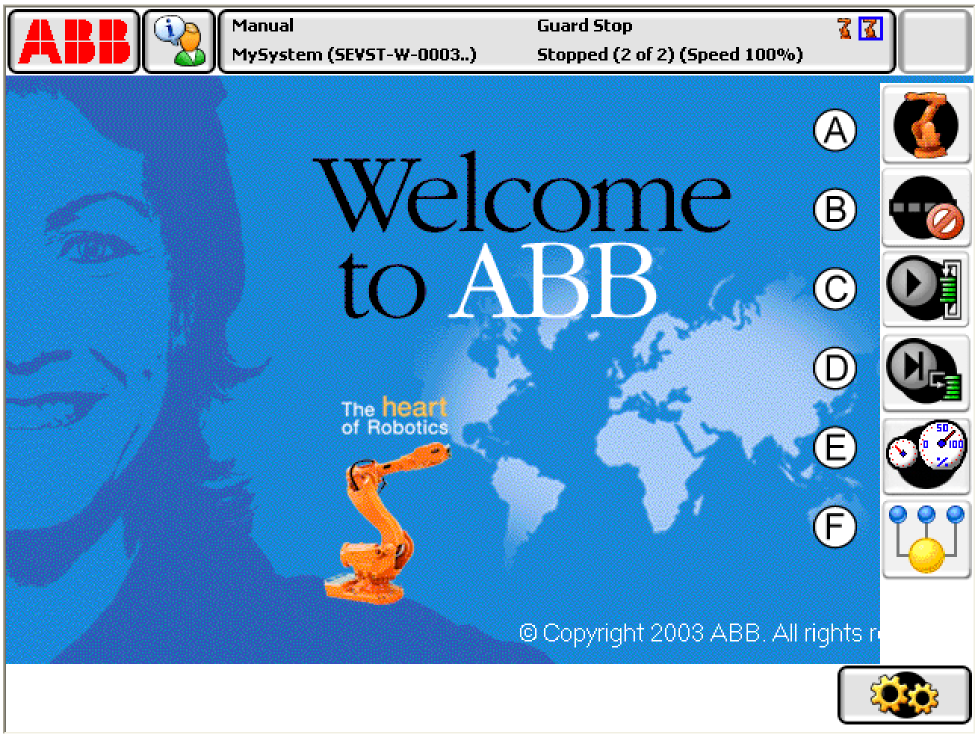
Quickset menu

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

Each item of the menu uses a symbol to display the currently selected property value or setting. Tap the Quickset button to display available property values.

Illustration of the Quickset menu

This section describes the buttons in the Quickset menu.



Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)

en0300000471

A	Mechanical unit, see section *Quickset menu, Mechanical unit on page 128*.
B	Increment, see section *Quickset menu, Increment on page 132*.
C	Run Mode, see section *Quickset menu, Run Mode on page 208*.
D	Step Mode, see section *Quickset menu, Step Mode on page 209*.
E	Speed, see section *Quickset menu, Speed on page 213*.
F	Tasks, see section *Quickset menu, Tasks on page 214*.

4.7 Basic procedures

4.7.1. Using the soft keyboard

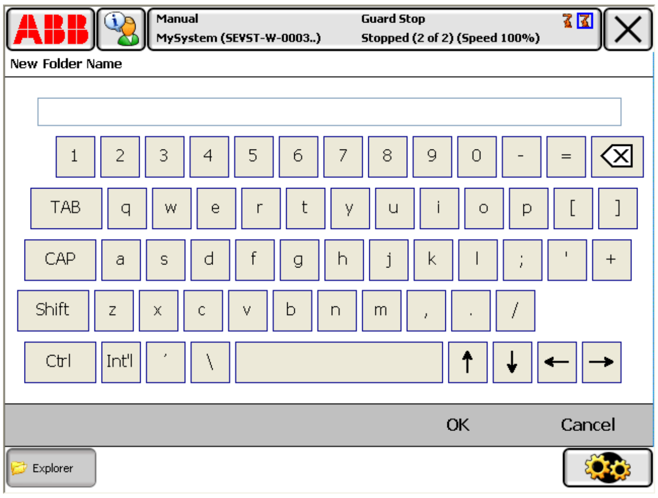
Soft keyboard

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

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

Illustration soft keyboard

This illustration shows the soft keyboard on the FlexPendant.


Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)
New Folder Name
1 2 3 4 5 6 7 8 9 0 - =
TAB q w e r t y u i o p [ ]
CAP a s d f g h j k l ; ' +
Shift z x c v b n m , . /
Ctrl Int'l ` \
OK
Cancel
Explorer

en0300000491

Using international characters

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

Changing the insertion point

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

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

Deleting

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

⌫

xx0300000494

4.7.2. Messages on the FlexPendant

Overview of messages

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

Event log messages

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

How to work with the event log messages is described in section *Handling the event log on page 255*. All messages are also described in *Operating manual - Trouble shooting*.

System messages

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

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

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

Program messages

RAPID programs can send out messages to the Operator window, see section *Operator window on page 95*.

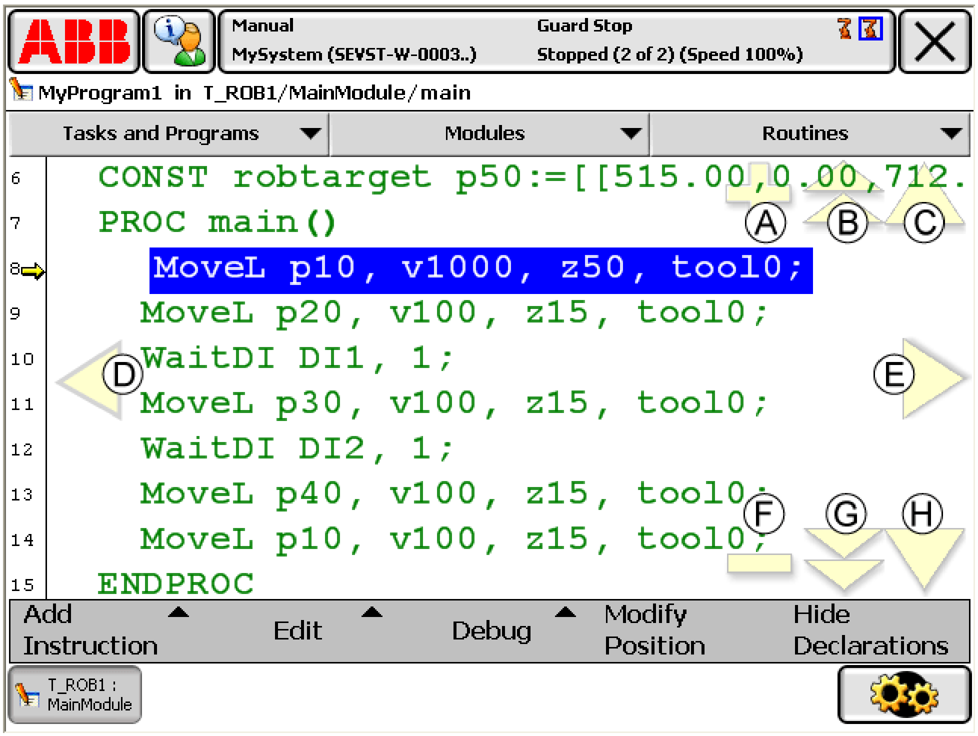
How to generate program messages is described in *Technical reference manual - RAPID Instructions, Functions and Data types*.

4.7.3. Scrolling and zooming

Overview

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

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



Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)

MyProgram1 in T_ROB1/MainModule/main

```
CONST robtarget p50:=[[515.00,0.00,712.
PROC main()
  MoveL p10, v1000, z50, tool0;
  MoveL p20, v100, z15, tool0;
  WaitDI DI1, 1;
  MoveL p30, v100, z15, tool0;
  WaitDI DI2, 1;
  MoveL p40, v100, z15, tool0;
  MoveL p10, v100, z15, tool0;
ENDPROC
```

en0400000685

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

4.7.4. Filtering data

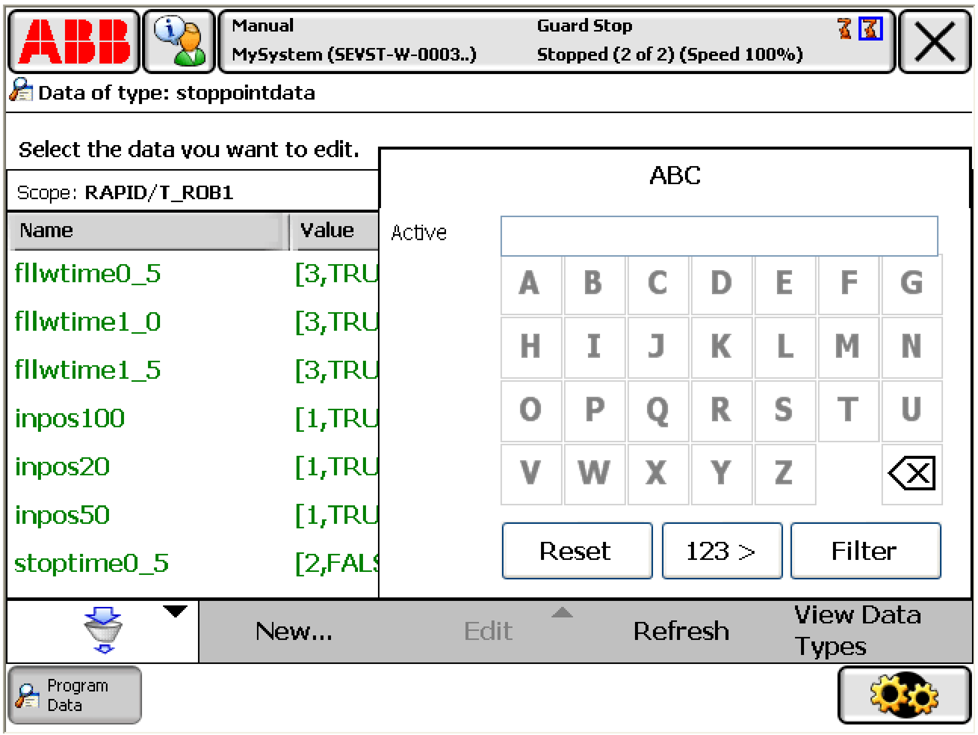
Filtering data

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

When filtering I/O signals there are more options than for many other types of data. For example the filtering function can be displayed automatically if the number of signals displayed exceeds a predefined number. See also *Creating I/O categories on page 250*.

Illustration of filtering

The filter function is switched on until the active filter is removed (e.g. by tapping **Reset**).


Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)

Data of type: stoppointdata

Select the data you want to edit.

Scope: RAPID/T_ROB1

Name	Value
fllwtime0_5	[3,TRU
fllwtime1_0	[3,TRU
fllwtime1_5	[3,TRU
inpos100	[1,TRU
inpos20	[1,TRU
inpos50	[1,TRU
stoptime0_5	[2,FAL!

ABC

Active

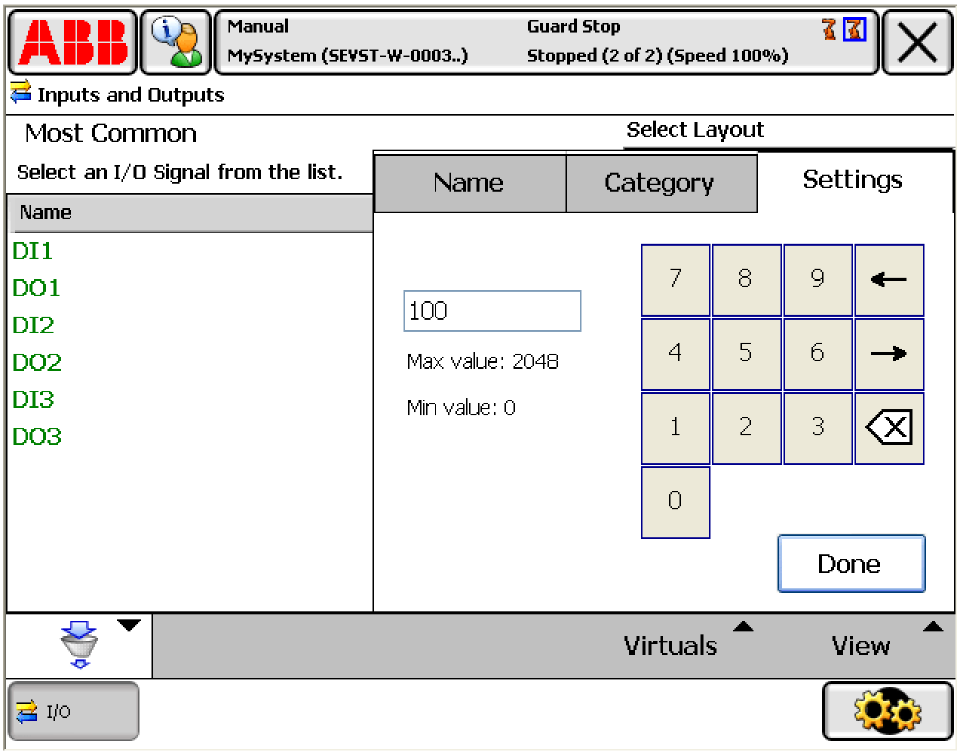
en0500001539

Active filter	Displays the current filter. It is also displayed at the top of the list of items.
123 / ABC / Name / Category / Settings	Depending on type of data, there may be one or several ways to filter data, e.g. numeric, alphabetic, or by category.
Reset	Removes the filter string.
Filter	Applies the filter.

Continues on next page

Illustration of automatic filter display

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



en0600002643

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

4.7.5. Process applications

Process applications

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

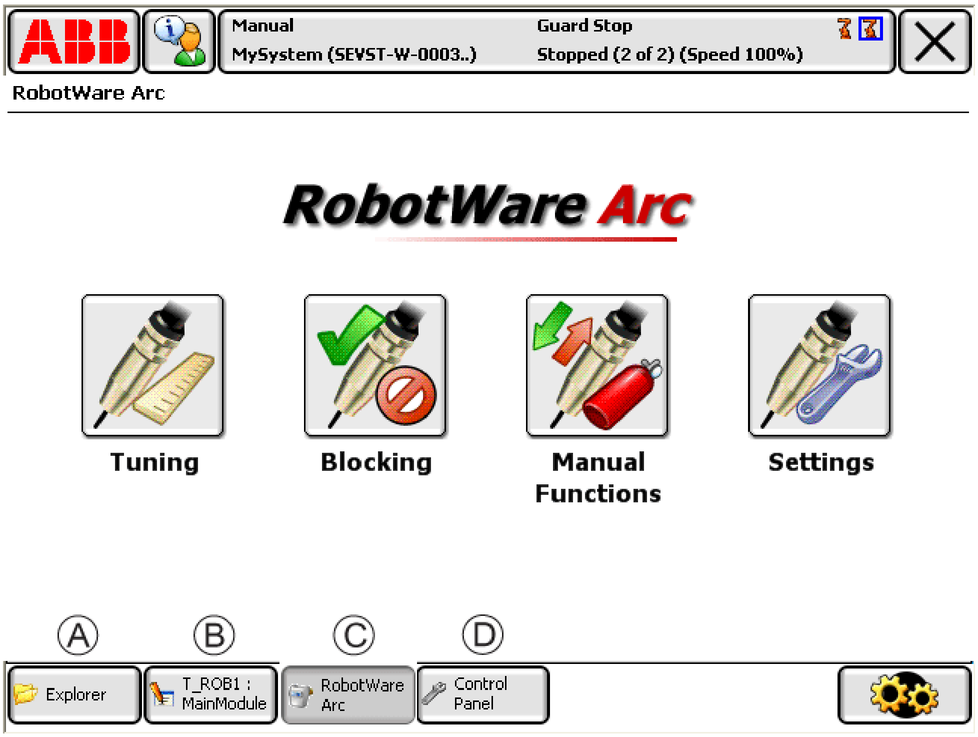
Start a process application

Use this procedure to start a process application.

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

Switch between started process applications

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



en0400000768

The views and process application running in this case are:

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

4.7.6. Logging on and off

Logout procedure

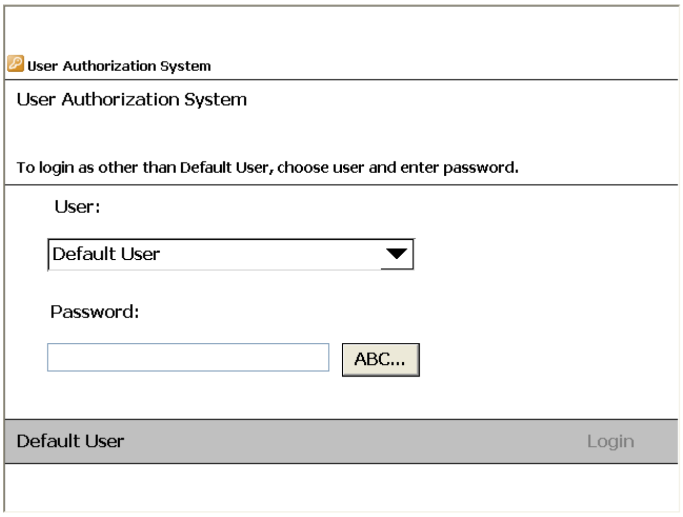
Use this procedure to log off the system.

	Action
1.	On the **ABB** menu, tap **Log Off**.
2.	Tap **Yes** to confirm.

Login procedure

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

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


User Authorization System
User Authorization System
To login as other than Default User, choose user and enter password.
User:
Default User
Password:
ABC...
Default User
Login

en0400000947

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

Handling users and authorization levels

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

How to edit what views or functions are hidden for certain users is described in *Defining a view to be shown at operating mode change on page 301*.

5 Jogging

5.1. Introduction to jogging

What is jogging?

To jog is to manually position or move robots or external axes using the FlexPendant joystick.

When can I jog?

You jog in manual mode. Jogging is possible regardless of what view is displayed on the FlexPendant, however you cannot jog during program execution.

About motion modes and robots

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

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

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

About motion modes and additional axes

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

Additional axes are not affected by the selected coordinate system.

About coordinate systems

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

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

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

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

Read more about coordinate systems in section *What is a coordinate system? on page 323*.

5.2. Restrictions to jogging

Jog additional axes

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

Jog mechanical units that are not calibrated

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

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

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

CAUTION!

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

Jog robot axes in independent mode

It is not possible to jog axes in independent mode. You need to return the axes to normal mode in order to jog. Please see *Application manual - Motion functions and events* for details.

Jog while using world zones

With the option *World Zones* installed, defined zones will restrict motion while you jog. Please see *Application manual - Motion functions and events* for details.

Jog with axis loads not set

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

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

Jog with tool or payload weights not set

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

5.3. Coordinated jogging

Coordination

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

Coordinated jogging

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

Set up coordination

	Action	Info
1.	Select the robot that is to be coordinated to another mechanical unit.	See *Selecting mechanical unit for jogging on page 108*.
2.	Set **Coordinate system** to Work Object.	See *Jog in work object coordinates on page 119*.
3.	Set **Work object** to the work object moved by the other mechanical unit.	See *Selecting tool, work object, and payload on page 112*.
4.	Select the mechanical unit that moves the work object.	Any jogging, while this mechanical unit is selected, will also affect the robot that is coordinated with it.

Coordinating robots

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

5.4 Basic settings for jogging

5.4.1. Selecting mechanical unit for jogging

Examples of use

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

If your system only has a single robot without additional axes, then you do not need to select mechanical unit.

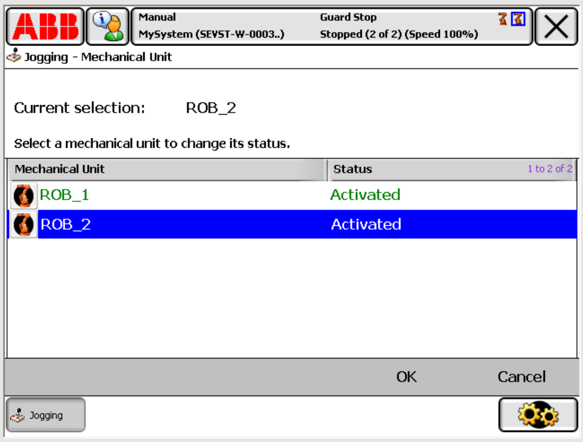
Identifying mechanical units

Each mechanical unit that can be jogged is represented in the mechanical units list. The name of the unit is defined in the system configuration. Each unit also has a symbol that is used in the Status bar, see section *Status bar on page 96*.

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

Selecting mechanical unit

This procedure describes how to select a mechanical unit to jog.

	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Tap **Mechanical Unit**.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Jogging - Mechanical Unit OK Cancel Jogging Current selection: ROB_2 Select a mechanical unit to change its status. Mechanical Unit — Status — 1 to 2 of 2 ROB_1 — Activated ROB_2 — Activated en0400000653 A list of available mechanical units is displayed.
3.	Tap the mechanical unit to be jogged, and then tap **OK**. The selected mechanical unit is active until you select another unit, even if you close the **Jogging** window.

TIP!

Use the **QuickSet** menu to switch between mechanical units faster.

Continues on next page

How jogging properties apply

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

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

Related information

If the system uses *Multitasking*, and has more than one motion task, and uses more than one mechanical unit, then the selected mechanical unit can be switched automatically when switching between Program Editor windows. See section *Program Editor on page 84*.

Mechanical units can be activated or deactivated with the Activate function in the Jogging window, see section *Activating mechanical units on page 204*.

5.4.2. Selecting motion mode

Overview

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

CAUTION!

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

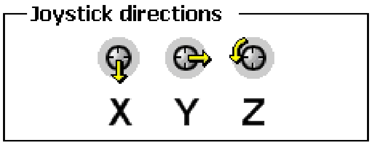
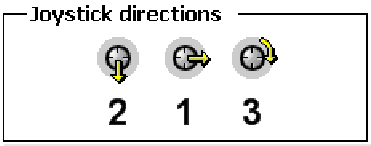
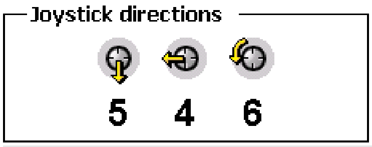
Selecting motion mode

This procedure describes how to select motion mode.

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

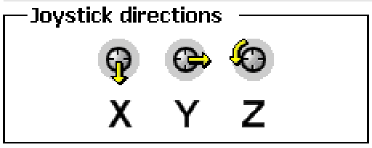
Joystick directions

The significance of the joystick directions depends on what motion mode has been selected. The following are available:

Motion mode	Illustration joystick	Description
Linear	 en0400001131	Linear mode is described in section *Setting the tool orientation on page 113*.
Axis 1-3 (default for robots)	 en0300000536	Axis 1-3 mode is described in section *Jog axis by axis on page 114*.
Axis 4-6	 en0300000537	Axis 4-6 mode is described in section *Jog axis by axis on page 114*.

Continues on next page

Continued

Motion mode	Illustration joystick	Description
Reorient	 en0400001131	Reorient mode is described in section *Setting the tool orientation on page 113*.

Default settings

The linear and reorientation motion modes have default settings for coordinate systems, valid per mechanical unit. These are always set after a restart. If you change the coordinate system for one of these motion modes, the change will be remembered until the next restart (warm start).

Motion mode	Default coordinate system
Linear	Base coordinate system
Reorientation	Tool coordinate system

5.4.3. Selecting tool, work object, and payload

Overview

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

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

Selecting tool, work object, and payload

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

5.4.4. Setting the tool orientation

Examples of use

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

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

Definition of tool orientation

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

Setting the tool orientation

	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Tap **Motion Mode**, then tap **Reorient** followed by **OK**.
3.	If not already selected, select the proper tool by following the procedure in *Selecting tool, work object, and payload on page 112*.
4.	Press and hold the enabling device to activate the mechanical unit's motors. Move the joystick and the tool's orientation changes.

TIP!

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

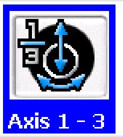
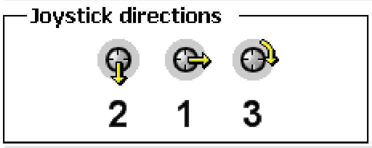
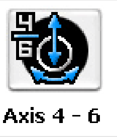
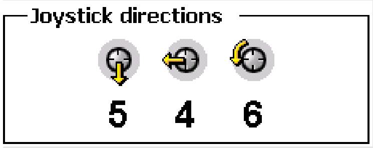
5.4.5. Jog axis by axis

Examples of use

Use axis by axis jogging when you need to:

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

How to determine which axis will be moved

If you want to move…	then tap…	for joystick directions…
axis 1, 2 or 3	 en0300000534	 en0300000536
axis 4, 5 or 6	 en0300000535	 en0300000537

Start jogging

This is how you get started:

	Action
1.	On the **ABB** menu, tap **Jogging** to view jogging properties.
2.	Tap **Motion Mode**, then select axes 1-3 or 4-6.
3.	Tap **OK** to complete.
4.	How to use the joystick when jogging axis by axis is displayed in the **Joystick directions** area. See also *Illustration of axes and joystick directions on page 115*.
5.	Press the enabling device and start jogging.

CAUTION!

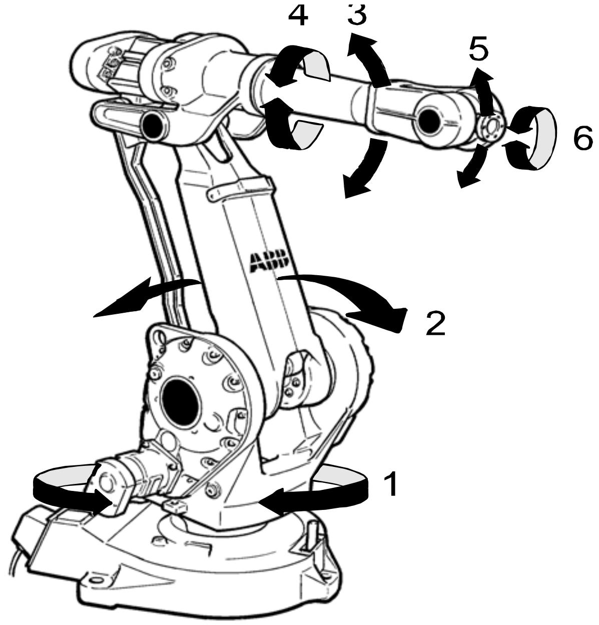
The orientation of any mounted tool will be affected by this procedure. If the resulting orientation is important, perform the procedure described in *Setting the tool orientation on page 113* when finished.

Continues on next page

Illustration of axes and joystick directions

The regular six axes of a generic manipulator may be jogged manually using the three dimensions of the joystick as specified below. Please check your plant or cell documentation to determine the physical orientation of any additional axes.

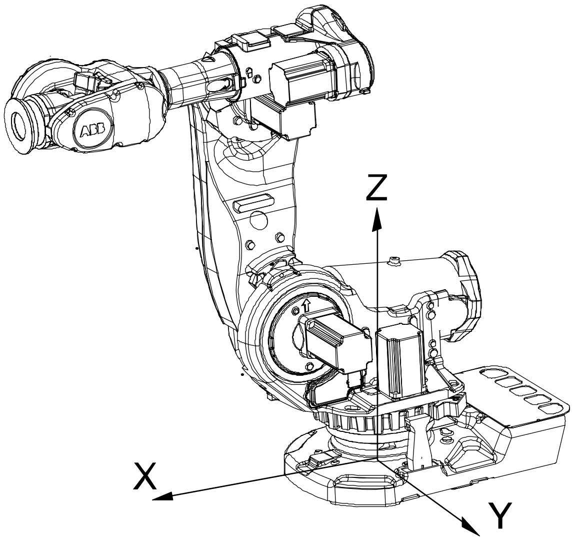
The illustration shows the movement patterns of each of the manipulator axes.



xx0300000520

5.4.6. Jog in base coordinates

Base coordinates definition



xx0300000495

Jog in base coordinates

	Action
1.	On the **ABB** menu, tap **Jogging** to view jogging properties.
2.	Tap **Motion Mode**, then tap **Linear** followed by **OK**. You don't need to perform this step if you previously selected linear motion.
3.	Tap **Coordinate System**, then tap **Base** followed by **OK**.
4.	Press and hold the enabling device to activate the manipulator's motors.
5.	Move the joystick and the mechanical unit moves along.

TIP!

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

Default settings

For each mechanical unit the system will by default use the base coordinate system for the linear motion mode. If you change coordinate system in the jogging properties, this will automatically be reset to the base coordinate system for linear motion mode after a restart.

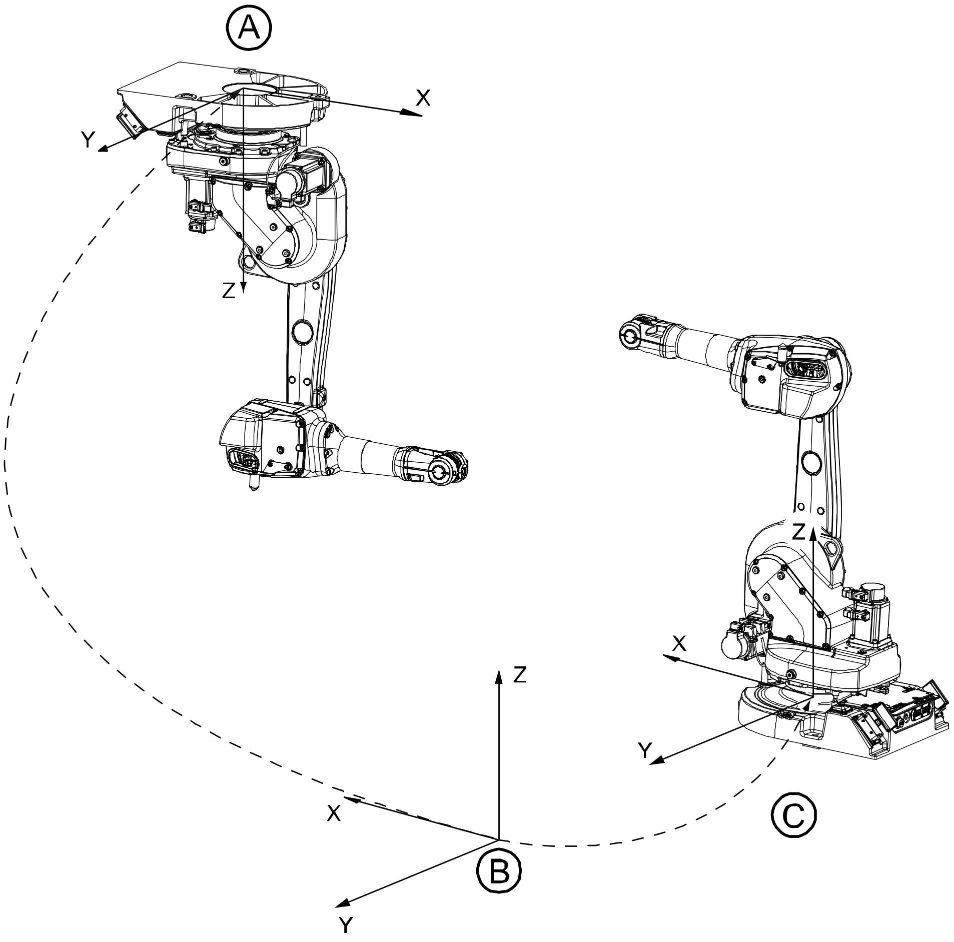
5.4.7. Jog in world coordinates

Examples of use

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

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

World coordinates definition



en0300000496

A	Base coordinate system
B	World coordinate system
C	Base coordinate system

Jog in world coordinates

	Action
1.	On the **ABB** menu, tap **Jogging** to view jogging properties.
2.	Tap **Motion Mode**, then tap **Linear** followed by **OK**. You don't need to perform this step if you previously selected linear motion.

Continues on next page

Continued

	Action
3.	Tap **Coordinate system**, then tap **World** followed by **OK**.
4.	Press and hold the enabling device to activate the manipulator's motors.
5.	Move the joystick and the mechanical unit moves along.

TIP!

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

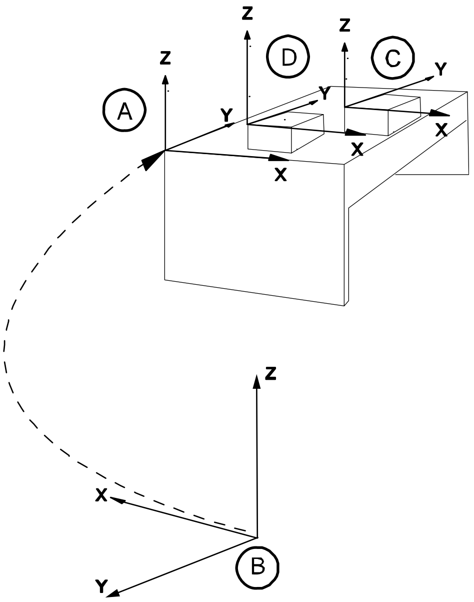
5.4.8. Jog in work object coordinates

Examples of use

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

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

Work object coordinates definition



en0300000498

A	User coordinate system
B	World frame
C	Work object coordinate system
D	Work object coordinate system

Jog in work object coordinates

	Action
1.	On the **ABB** menu, tap **Jogging** to view jogging properties.
2.	Tap **Motion Mode**, then tap **Linear** followed by **OK**. This is not required if you previously selected linear motion.
3.	Tap **Work object** to select work object.
4.	Tap **Coordinate system**, then tap **Work Object** followed by **OK**.
5.	Press and hold the enabling device to activate the manipulator's motors.
6.	Move the joystick and the mechanical unit moves along.

TIP!

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

5.4.9. Jog in tool coordinates

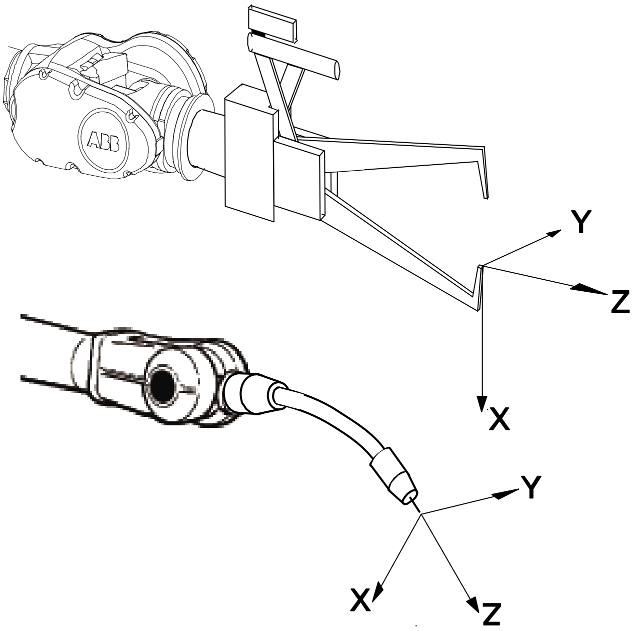
Examples of use

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

Tool coordinates definition

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

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



en0300000497

Jog in tool coordinates

	Action
1.	On the **ABB** menu, tap **Jogging** to view jogging properties.
2.	Tap **Motion mode**, then tap **Linear** followed by **OK**. You do not need to perform this step if you previously selected linear motion.
3.	Select the proper tool, and if using a stationary tool the proper work object by following the procedure in *Selecting tool, work object, and payload on page 112*. You do not need to perform this step if you previously selected the tool and/or work object.
4.	Tap **Coordinate System**, then tap **Tool** followed by **OK**.
5.	Press and hold the enabling device to activate the mechanical unit's motors.
6.	Move the joystick and the mechanical unit moves along.

Continues on next page

TIP!

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

Jog with a stationary tool

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

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

Default settings

For each mechanical unit the system will by default use the tool coordinate system for the reorientation motion mode. If you change coordinate system in the jogging properties, this will automatically be reset to the tool coordinate system for reorientation motion mode after a restart.

5.4.10. Locking the joystick in specific directions

Overview

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

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

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

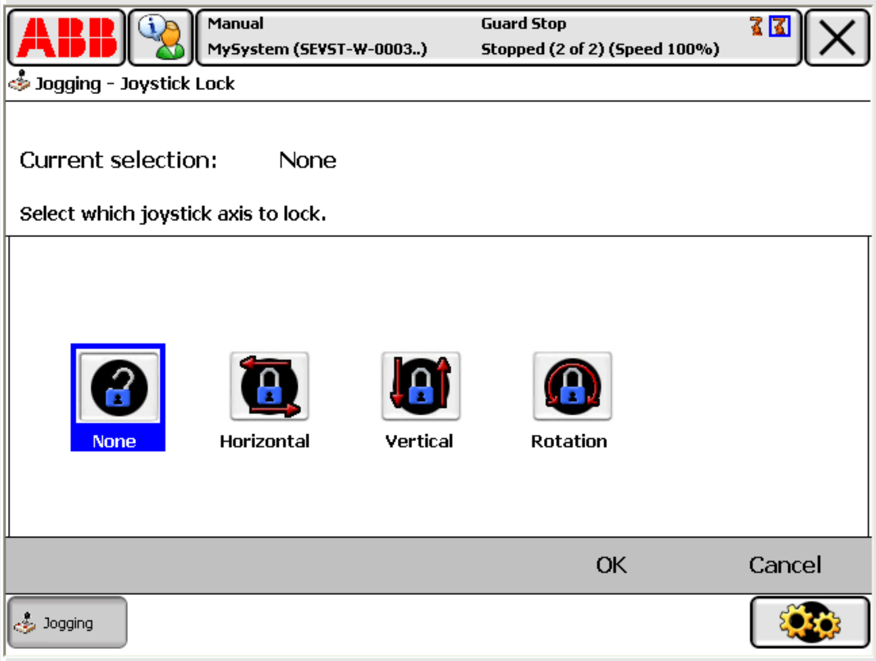
Which axes are locked?

This section describes how to see which joystick directions are locked

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

Locking the joystick in specific directions

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

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Joystick lock**.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Jogging - Joystick Lock Current selection: None Select which joystick axis to lock. None Horizontal Vertical Rotation OK Cancel Jogging en0300000488
3.	Tap the joystick axis or axes that should be locked. The axis toggles between locked and unlocked each time you tap.
4.	Tap **OK** to lock.

Continues on next page

Continued

Unlocking all axes

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

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

5.4.11. Incremental movement for precise positioning

Incremental movement

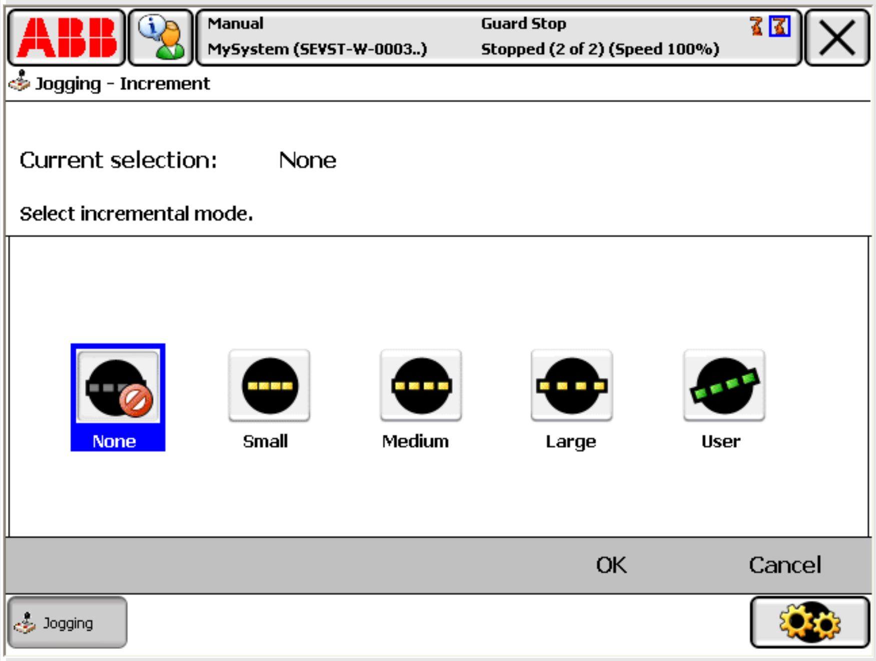
Use incremental movement to jog the robot in small steps, which enables very precise positioning.

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

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

Set the incremental movement size

This procedure details how to specify the size of the incremental movement.

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Increment**.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Jogging - Increment Current selection: None Select incremental mode. None Small Medium Large User OK Cancel Jogging en0400000971
3.	Tap the desired increment mode, see description in section *Incremental movement sizes on page 124*.
4.	Tap **OK**.

Incremental movement sizes

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

Increment	Distance	Angular
Small	0.05 mm	0.005°

Continues on next page

Continued

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

5.4.12. Reading the exact position

About positions and revolution counters

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

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

CAUTION!

If the positions are displayed in red text then the values from the revolution counters are lost and instead the values stored on the SMB are displayed. Be careful when jogging the robot if the values are displayed in red text. Watch the robot closely and do not use the displayed values! If the mechanical unit is uncalibrated then the actual position can be very different from the position values stored by the SMB. You must update the revolution counters before a program can be started. See *Updating revolution counters on page 288*. See *Serial Measurement Board memory on page 294* for more information about data stored on the SMB.

NOTE!

If no positions are displayed then the mechanical unit is uncalibrated. Instead the text “Unit not calibrated" is displayed.

How robot positions are displayed

Positions are always displayed as:

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

How additional axes’ positions are displayed

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

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

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

Reading the exact position

This procedure describes how to read the exact position.

	Action
1.	On the **ABB** menu tap **Jogging**.
2.	The position is displayed in the **Position** area properties in the right hand side of the window. See illustration in *Jogging on page 79*.

Continues on next page

Position format

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

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

- World
- Base
- Work object

The *Orientation format* can be set to:

- Quaternion
- Euler angles

The *Position angle format* can be set to:

- Angles

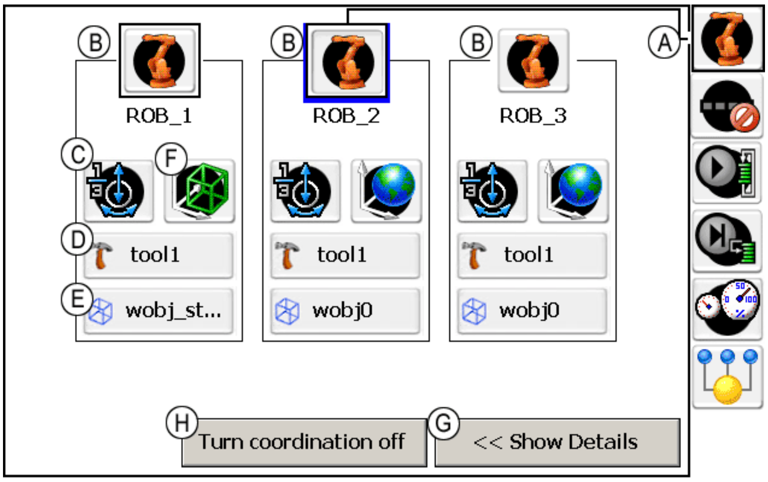
The *Presentation angle unit* can be set to:

- Degrees
- Radians

5.4.13. Quickset menu, Mechanical unit

Illustration Mechanical unit button

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



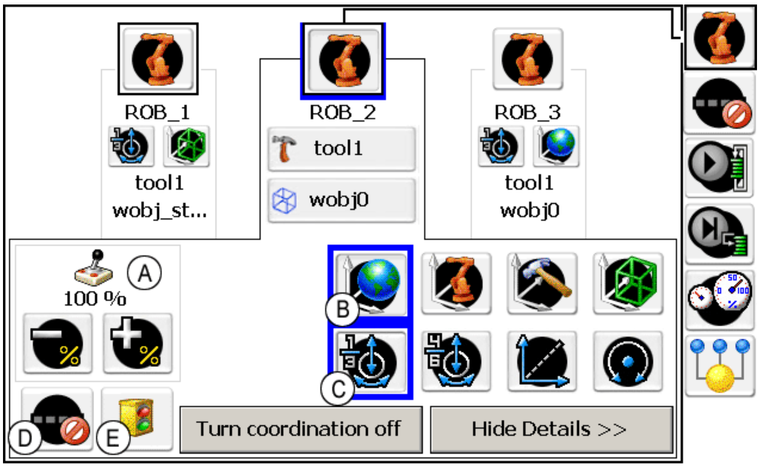
en0300000539

A	Mechanical unit menu button
B	Mechanical unit, a selected unit is highlighted
C	Motion mode settings (axes 1-3 motion mode currently selected)
D	Tool settings (tool 0 currently selected)
E	Work object settings (work object 0 currently selected)
F	Coordinate system settings (world coordinate currently selected)
G	Show details
H	Turn coordination off (for information about this button, see *Turn coordination off on page 131*).

Each button is described below.

Illustration Show Details

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



en0500002354

A	Override jog speed settings (100% currently selected)
B	Coordinate system settings (world coordinate currently selected)

Continues on next page

C	Motion mode settings (axes 1-3 motion mode currently selected)
D	Turn on or off user increment
E	Turn on or off jog supervision

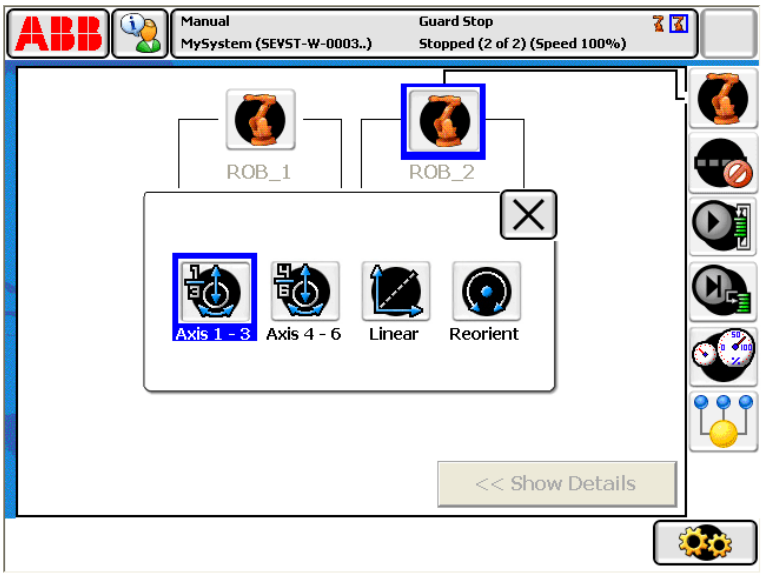
If any of the settings are not available, they will crossed over.

Motion mode and coordinate settings may be changed by tapping the button required.

Tap **Hide Details** after making any selection to return to the basic display.

Illustration Motion mode settings

To view or change any motion mode functionality, tap the Motion mode settings button.



Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)

en0300000540

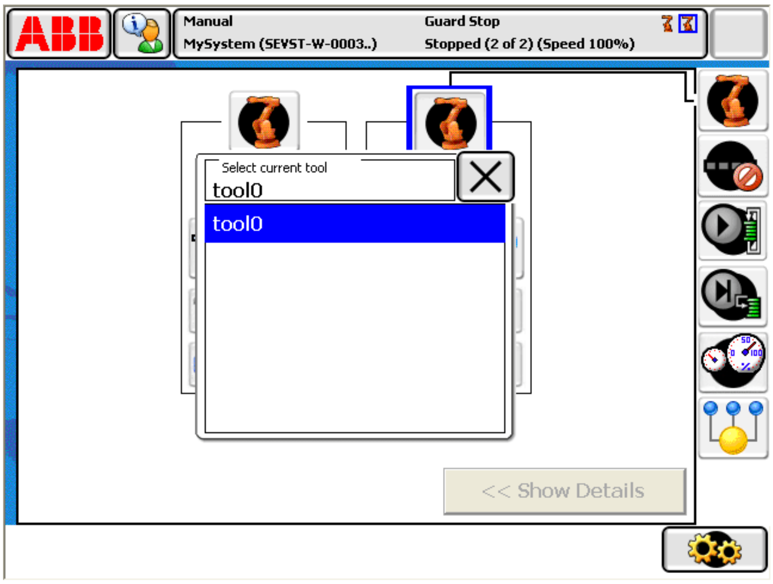
Select motion mode setting:

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

Continues on next page

Illustration Tool settings

To view or change the available tools, tap the Tool settings button.

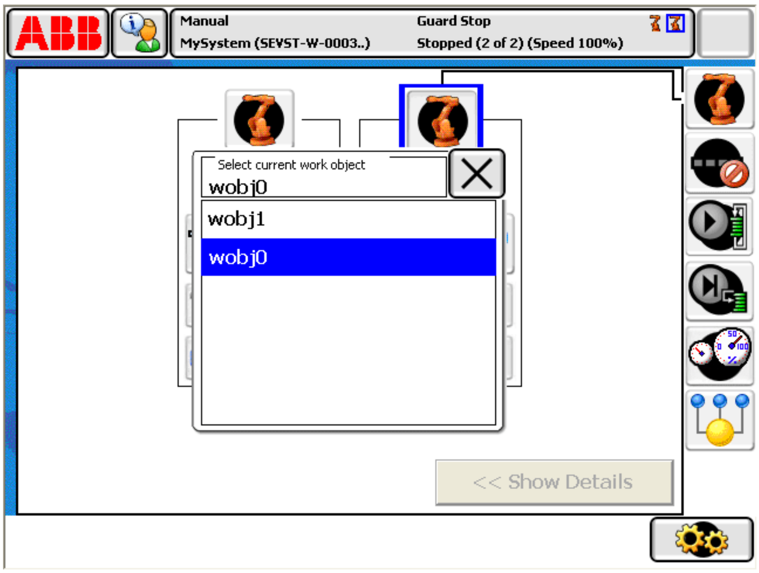


en0400000988

Select a tool to use.

Illustration Work object settings

To view or change the available work objects, tap the Work object settings button.



Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)
Select current work object
wobj0
wobj1
wobj0
<< Show Details

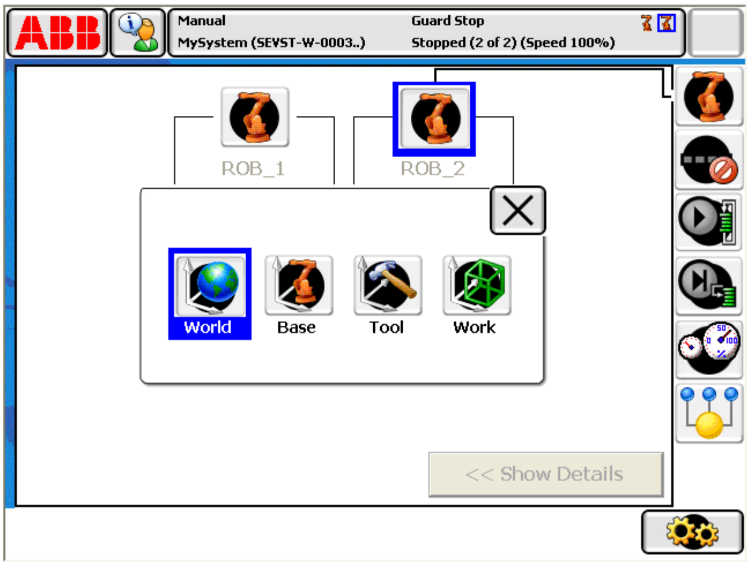
en0400000989

Select a work object to use.

Continues on next page

Illustration Coordinate system settings

To view or change Coordinate system functionality, tap the Coordinate system settings button.



en0300000541

Select a coordinate system setting:

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

Turn coordination off

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

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

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

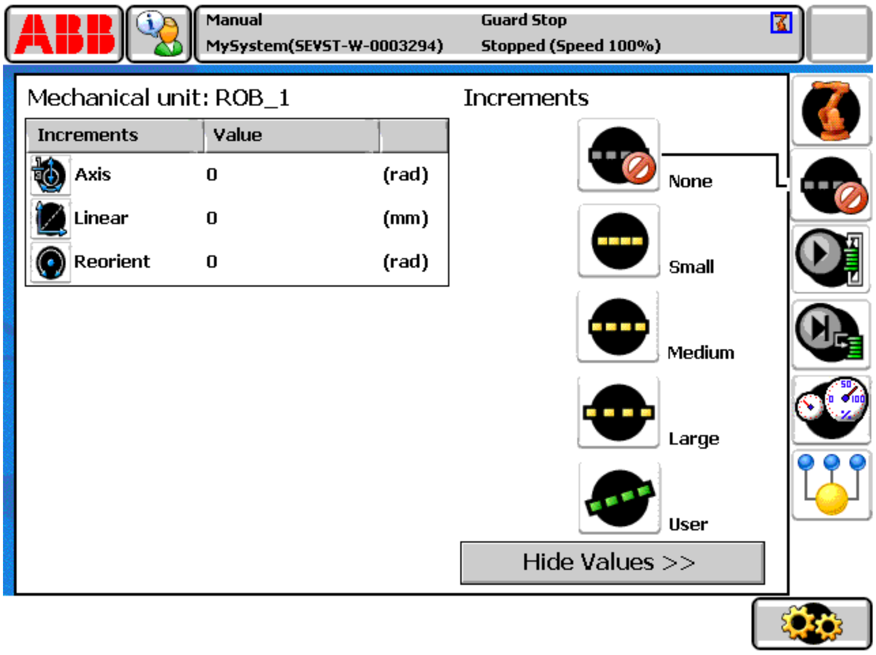
For information about coordination between MultiMove robots, see *Application manual - MultiMove*.

5.4.14. Quickset menu, Increment

Increment settings

All functions under this button may also be reached from the Jogging menu.

Illustration Increment



Manual
MySystem(SEVST-W-0003294)

Guard Stop
Stopped (Speed 100%)

Mechanical unit: ROB_1

Increments	Value	
Axis	0	(rad)
Linear	0	(mm)
Reorient	0	(rad)

en0300000542

None	No increments
Small	Small movements
Medium	Medium movement
Large	Large movements
User	User defined movements
Show Values	Displays increment values

6 Programming and testing

6.1. Before you start programming

Programming tools

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

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

Define tools, payloads, and work objects

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

Define coordinate systems

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

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

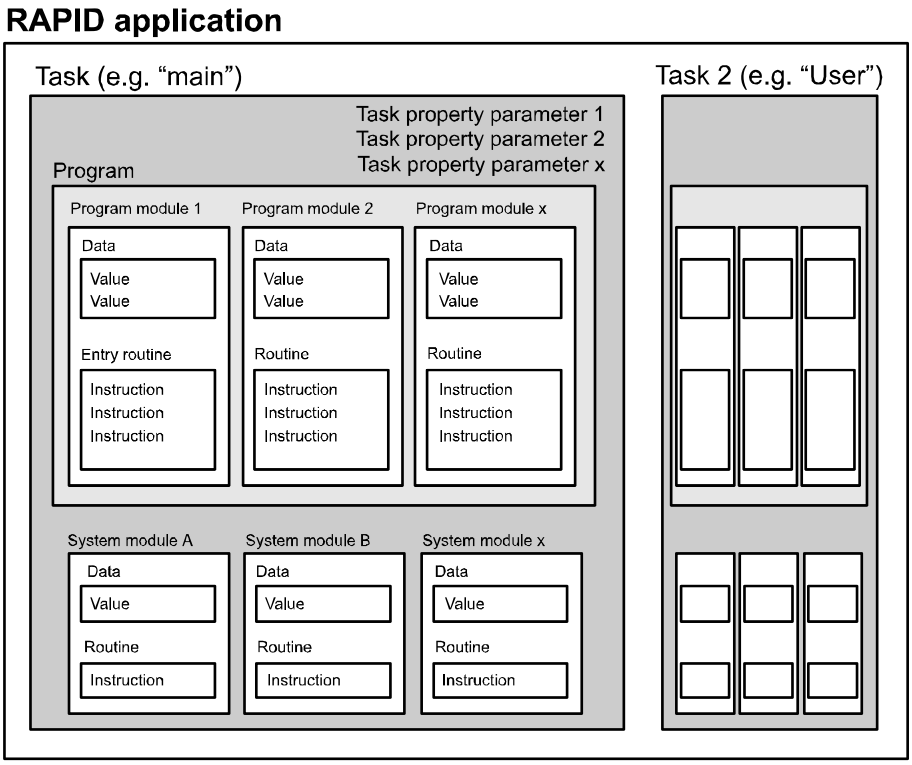
TIP!

Need to know more about the RAPID language and structure? See *Technical reference manual - RAPID overview* and *Technical reference manual - RAPID Instructions, Functions and Data types*.

6.2 Programming concept

6.2.1. The structure of a RAPID application

Illustration of a RAPID application



en0300000576

Parts

Part	Function
Task	Each task usually contains a RAPID program and system modules aimed at performing a certain function, e.g. spot welding or manipulator movements. A RAPID application may contain one task. If you have the *Multitasking* option installed, then there can be more than one task. Read more about *Multitasking* in *Application manual - Engineering tools*.
Task property parameter	The task property parameters set certain properties for all task contents. Any program stored in a certain task, assumes the properties set for that task. The task property parameters are specified in *Technical reference manual - RAPID overview*.
Program	Each program usually contains program modules with RAPID code for different purposes. Any program must have an entry routine defined to be executable.

Continues on next page

Continued

Part	Function
Program module	Each program module contains data and routines for a certain purpose. The program is divided into modules mainly to enhance overview and facilitate handling the program. Each module typically represents one particular robot action or similar. All program modules will be removed when deleting a program from the controller program memory. Program modules are usually written by the user.
Data	Data are values and definitions set in program or system modules. The data are referenced by the instructions in the same module or in a number of modules (availability depending on data type). Data type definitions are specified in the *Technical reference manual - RAPID Instructions, Functions and Data types*.
Routine	A routine contains sets of instructions, i.e. defines what the robot system actually does. A routine may also contain data required for the instructions.
Entry routine	A special type of routine, in English sometimes referred to as "main", defined as the program execution starting point. **NOTE!** Each program **must** have an entry routine called "main", or it will not be executable. How to appoint a routine as entry routine is specified in *Technical reference manual - RAPID overview*. The default name for main can be changed by the system parameter configurations, type *Task*. See *Technical reference manual - System parameters*.
Instruction	Each instruction is a request for a certain event to take place, e.g. "Run the manipulator TCP to a certain position" or "Set a specific digital output". The instructions, their syntax and function is thoroughly described in the *Technical reference manual - RAPID Instructions, Functions and Data types*.
System module	Each system module contains data and routines to perform a certain function. The program is divided into modules mainly to enhance overview and facilitate handling the program. Each module typically represents one particular robot action or similar. All system modules will be retained when "Delete program" is ordered. System modules are usually written by the robot manufacturer or line builder.

6.2.2. About the Program and Motion Pointers

The Program Pointer

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

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

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

The Motion Pointer

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

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

The cursor

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

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

Program Editor

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

The same behavior applies to the Production Window.

Related information

Production Window on page 81.

Program Editor on page 84.

Stepping instruction by instruction on page 210.

Starting programs on page 227.

6.3 Data types

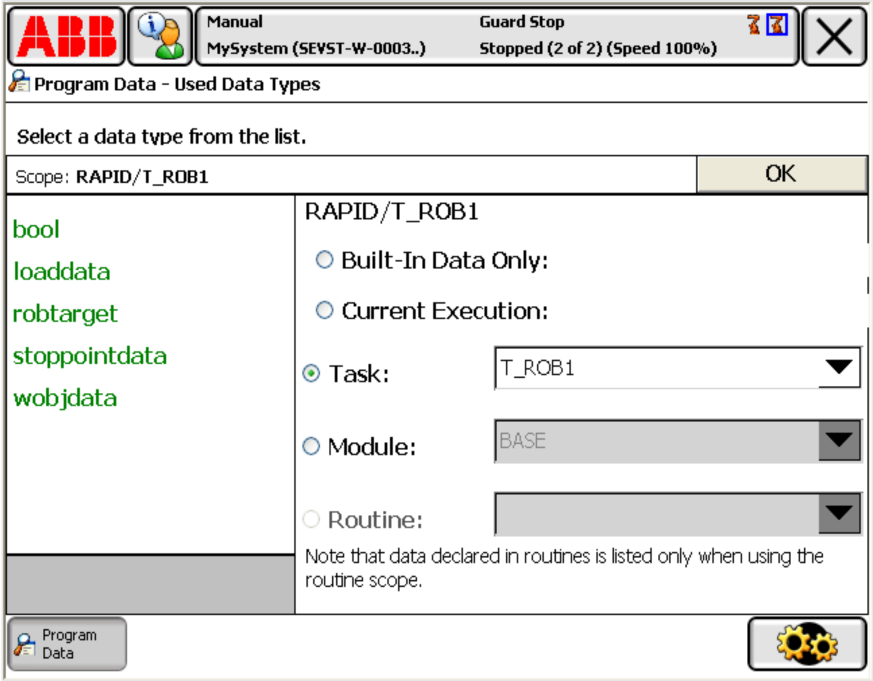
6.3.1. Viewing data in specific tasks, modules, or routines

Overview

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

Viewing data in specific tasks, modules, or routines

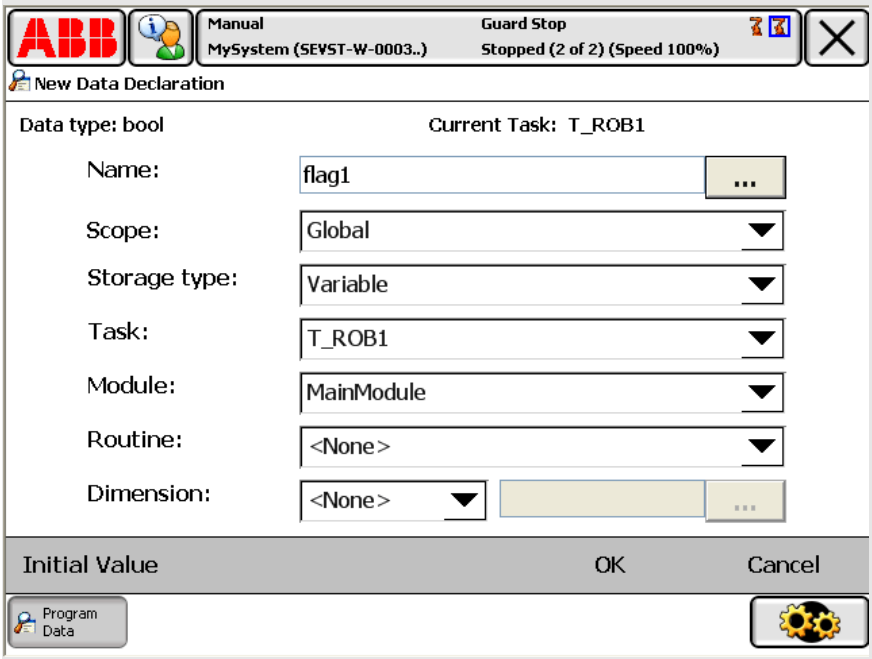
This section details how to view data instances in specific modules or routines.

	Action
1.	In the **ABB** menu, tap **Program Data**.
2.	Tap **Change Scope**. The following screen is displayed:  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Program Data - Used Data Types Select a data type from the list. Scope: RAPID/T_ROB1 bool loaddata robtarget stoppointdata wobjdata RAPID/T_ROB1 ○ Built-In Data Only: ○ Current Execution: ● Task: T_ROB1 ○ Module: BASE ○ Routine: Note that data declared in routines is listed only when using the routine scope. en0400000661
3.	Select the required scope by selecting: • Built-In Data Only: Shows all data types used by the specific system • Current execution: Shows all data types used in the current execution • Task: Shows all data types used by a specific task • Module: Shows all data types used by a specific module • Routine: Shows all data types used by a specific routine
4.	Tap **OK** to confirm your choice.
5.	Tap twice to select a data type and view its instances.

6.3.2. Creating new data instance

Creating new data instance

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

	Action
1.	In the **ABB** menu, tap **Program Data**. A list of all available data types is displayed.
2.	Tap the data instance type to be created, i.e. **bool** and then tap **Show data**. A list of all instances of the data type is displayed.
3.	Tap **New**.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) New Data Declaration Data type: bool Current Task: T_ROB1 Name: flag1 Scope: Global Storage type: Variable Task: T_ROB1 Module: MainModule Routine: <None> Dimension: <None> ... Initial Value OK Cancel Program Data en0400000663
4.	Tap **...** the right of **Name** to define the data instance's name.
5.	Tap the **Scope** menu to set accessibility for the data instance. Select: • **Global** • **Local** • **Task**
6.	Tap the **Storage type** menu to select type of memory used for the data instance. Select: • **Persistent** if the data instance is persistent • **Variable** if the data instance is variable • **Constant** if the data instance is constant
7.	Tap the **Module** menu to select module.
8.	Tap the **Routine** menu to select routine.

Continues on next page

	Action
9.	If you want to create an array of data instances, then tap the **Dimensions** menu and select the number of dimensions in the array, 1-3. • 1 • 2 • 3 • None Then tap **...** to set the **Size** of the array’s axes. Arrays are described in section *What is a data array? on page 334*
10.	Tap **OK**.

6.3.3. Editing data instances

Overview

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

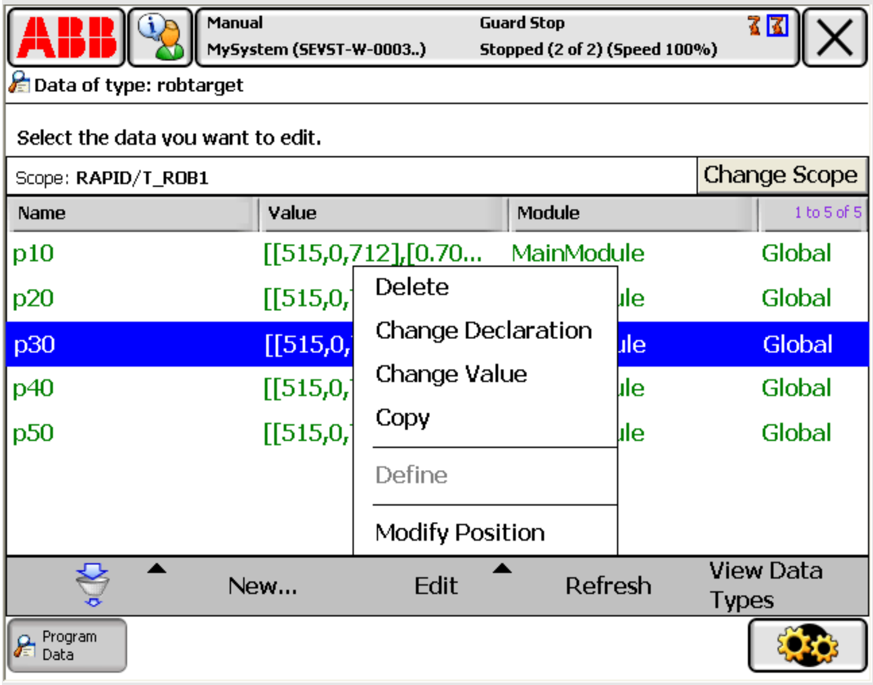
For the data types `tooldata`, `wobjdata` and `loaddata` also see sections *Tools on page 144*, *Work objects on page 155* or *Payloads on page 162*.

Viewing data instances

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

Action

1. In the **ABB** menu, tap **Program Data**.
2. Tap the data type you want view and then tap **Show Data**.
3. Tap the data instance you want to edit, and then tap **Edit**.



Manual — MySystem (SEVST-W-0003..)

Guard Stop — Stopped (2 of 2) (Speed 100%)

Data of type: robtarget

Select the data you want to edit.

Scope: RAPID/T_ROB1

Name	Value	Module	1 to 5 of 5
p10	[[515,0,712],[0.70...	MainModule	Global
p20	[[515,0,]	ule	Global
p30	[[515,0,	ule	Global
p40	[[515,0,]	ule	Global
p50	[[515,0,]	ule	Global

en0400000671

4. Depending on what you want to do, tap one of the following menu items:
   - Tap **Delete** to remove the data instance.
   - Tap **Change Declaration** to change the declaration of the data instance.
   - Tap **Change Value** to edit the value of the data instance.
   - Tap **Copy** to copy the data instance.
   - Tap **Define** to define the instance (only available for `tooldata`, `wobjdata` and `loaddata`).
   - Tap **Modify Position** to modify a position (only available for `robtarget` and `jointtarget`).

   Proceed as described in the respective section following below.

Continues on next page

Editing the value of a data instance

This section describes how to edit a data instance value.

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

NOTE!

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

Deleting a data instance

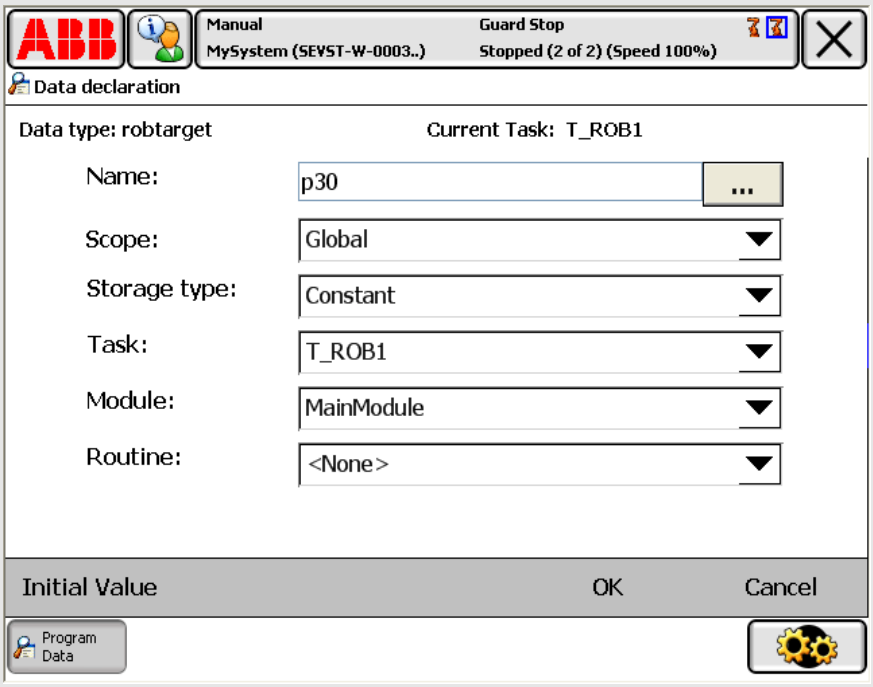
This section details how to delete a data instance.

	Action
1.	Tap **Delete** in the menu for the data instance to be deleted, as detailed in section *Viewing data instances on page 140*. A dialog box is displayed.
2.	Tap **Yes** if you are sure the data instance is to be deleted.

Continues on next page

Changing the declaration of a data instance

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

	Action
1.	Tap **Change Declaration** in the menu for the data instance to be deleted, as detailed in section *Viewing data instances on page 140*.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Data declaration Data type: robtarget Current Task: T_ROB1 Name: p30 Scope: Global Storage type: Constant Task: T_ROB1 Module: MainModule Routine: <None> en0400000672
2.	Select what data instance values to be changed: • Name: Tap **...** to bring out the soft keyboard and change the name. • Scope • Storage type • Module • Routine

Copying a data instance

This section details how to copy a data instance.

	Action
1.	Tap **Copy** in the menu for the data instance to be copied, as detailed in section *Viewing data instances on page 140*. A copy of the data instance is created. The copy has the same values as the original, but the name is unique.

Defining a data instance

How to define the tool frame or work object frame is described in the sections *Defining the tool frame on page 146* and *Defining the work object coordinate system on page 156*.

Continues on next page

Modifying position of a data instance

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

More information about modifying positions is detailed in *Modifying and tuning positions on page 186*.

NOTE!

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

6.4 Tools

6.4.1. Creating a tool

What happens when you create a tool?

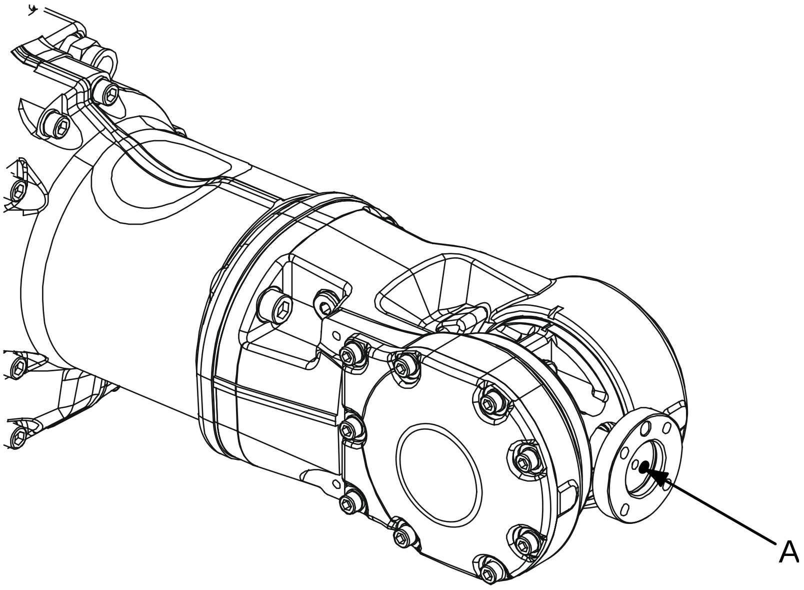
When you create a new tool a variable of the data type `tooldata` is created. The variable name will be the name of the tool. For more information on data types, see *Technical reference manual - RAPID Instructions, Functions and Data types*.

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

How to create a tool

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

By creating a new tool you define another tool center point. For more information about tools and the tool center points see *What is a tool? on page 320*and *What is the tool center point? on page 321*.

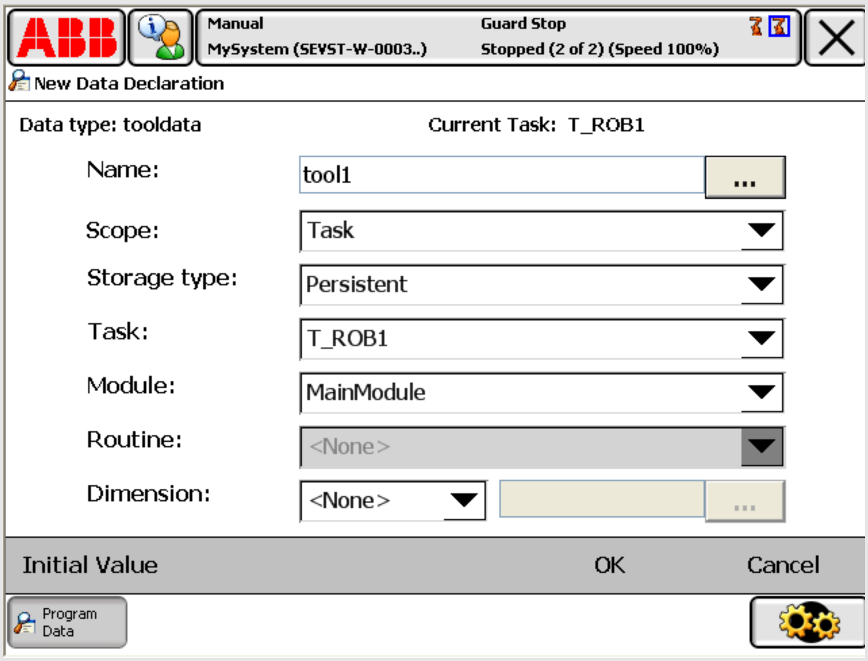


en0400000779

A	Tool center point, TCP, for tool0

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

Continues on next page

	Action
3.	Tap **New...** to create a new tool.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) New Data Declaration Initial Value OK Cancel Program Data Data type: tooldata Current Task: T_ROB1 Name: tool1 Scope: Task Storage type: Persistent Task: T_ROB1 Module: MainModule Routine: <None> Dimension: <None> en0300000544 Enter values for each field, see table below.
4.	Tap **OK**.

Tool declaration settings

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

NOTE!

The created tool is not useful until you have defined the tool data (TCP coordinates, orientation, weight etc.). See *Editing the tool data on page 149* and *LoadIdentify, load identification service routine on page 221* to learn more about how to do it.

6.4.2. Defining the tool frame

Preparations

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

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

Available methods

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

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

How to select a method

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

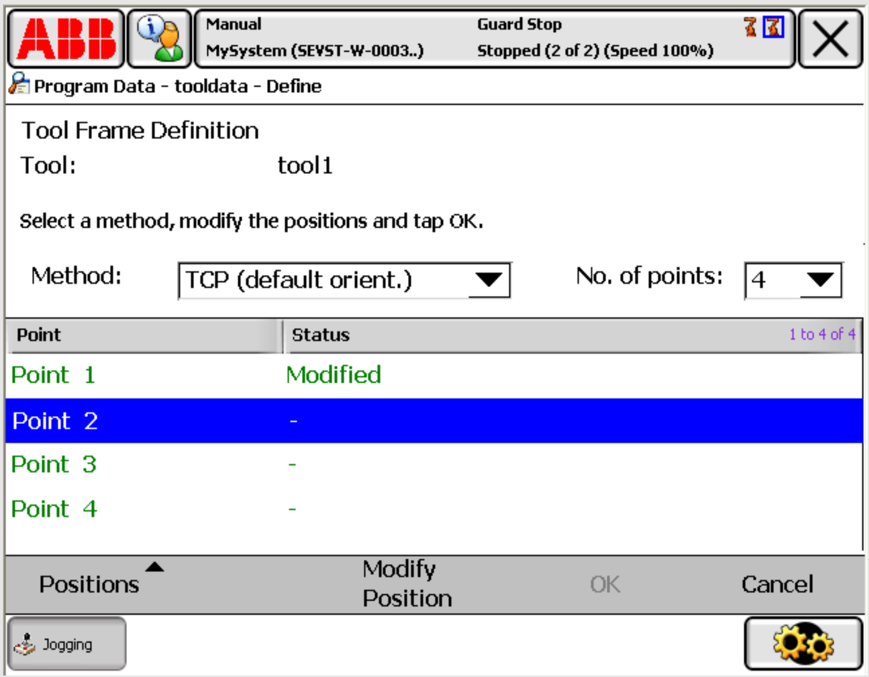
	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Tap **Tool** to display a list of available tools.
3.	Select the tool you want to define.
4.	In the Edit menu, tap **Define...**
5.	In the dialog box which appears, select the method to use. 

ABB
Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)
Program Data - tooldata - Define

Tool Frame Definition

Tool: tool1

Select a method, modify the positions and tap OK.

Method: TCP (default orient.)

No. of points: 4

Point	Status
Point 1	Modified
Point 2	-
Point 3	-
Point 4	-

1 to 4 of 4
Positions
Modify Position
OK
Cancel
Jogging

en0600003147

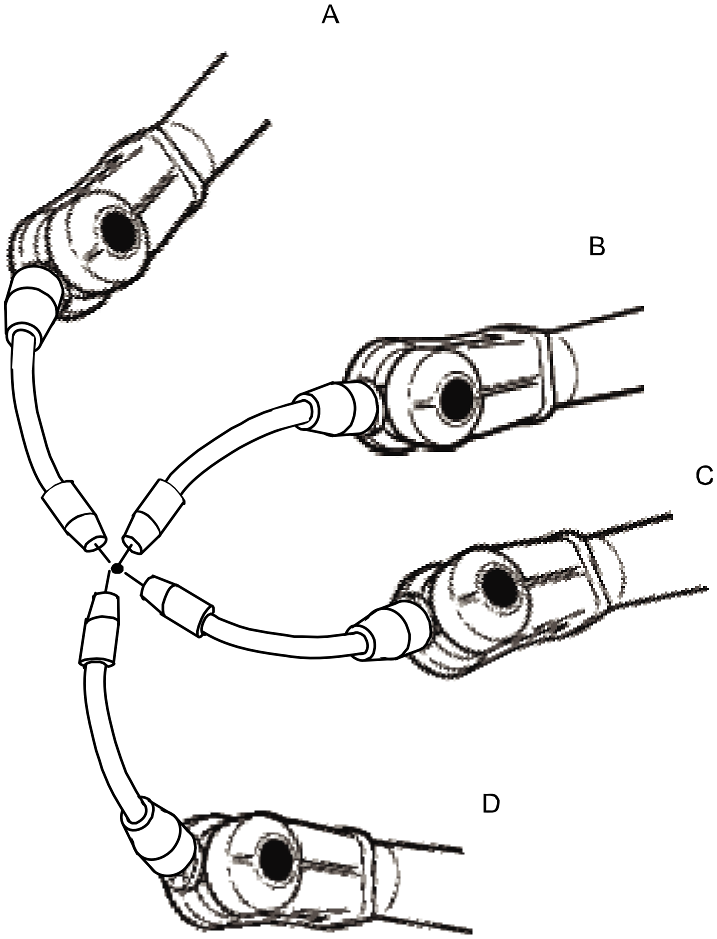
Continues on next page

Continued

	Action
6.	Select the number of approach points to use. Usually 4 points is enough. If you choose more points to get a more accurate result, you should be equally careful when defining all of them.
7.	See *How to proceed with tool frame definition on page 147* for information on how to gather positions and perform the tool frame definition.

How to proceed with tool frame definition

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



en0400000906

	Action	Info
1.	Jog the robot to an appropriate position, A, for the first approach point.	Use small increments to accurately position the tool tip as close to the reference point as possible.
2.	Tap **Modify Position** to define the point.	
3.	Repeat step 1 and 2 for each approach point to be defined, positions B, C, and D.	Jog away from the fixed world point to achieve the best result. Just changing the tool orientation will not give as good a result.
4.	If the method you are using is TCP & Z or TCP & Z, X orientation must be defined as well.	Follow the instructions in *How to define elongator points on page 148*.

Continues on next page

Continued

	Action	Info
5.	If, for some reason, you want to redo the calibration procedure described in step 1-4, tap **Positions** and then **Reset All**.	
6.	When all points are defined you can save them to file, which enables you to reuse them later. On the **Positions** menu, tap **Save**.	
7.	Tap **OK**. The **Calculation Result** dialog box will now be displayed, asking you to cancel or to confirm the result before it is written to the controller.	For further information see *Is the calculated result good enough? on page 148*

How to define elongator points

This procedure describes how to define the orientation of the tool frame by specifying the direction of the z and/or x axis. You need to do this only if you the tool orientation should differ from that of the robot base. The tool coordinate system by default resembles the coordinate system of tool0, as illustrated in *Measuring the tool center point on page 149*.

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

Is the calculated result good enough?

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

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

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

TIP!

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

6.4.3. Editing the tool data

Tool data

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

This can also be done automatically with the service routine LoadIdentify. See sections *Running a service routine on page 215*, or *LoadIdentify, load identification service routine on page 221*.

Displaying the tool data

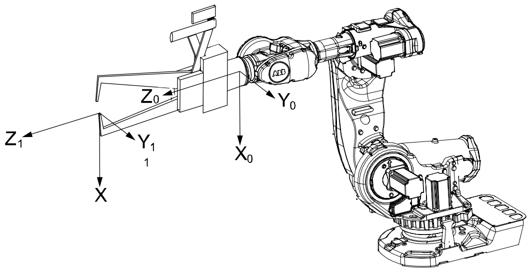
This section details how to display the tool data.

	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Tap **Tool** to display the list of available tools.
3.	Tap the tool you want to edit, then tap **Edit**. A menu appears. • Change Declaration • Change Value • Delete • Define
4.	In the menu, tap **Change Value**. The data that defines the tool appears. Green text indicates that the value can be changed.
5.	Proceed with changing the data as described below.

Measuring the tool center point

The easiest way to define the tool center point, TCP, is usually to use the predefined method described in *Defining the tool frame on page 146*. If you use this method, you do not have to write any values for the frame as these are supplied by the method.

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



en0400000881

X_0	X axis for tool0
Y_0	Y axis for tool0
Z_0	Z axis for tool0
X_1	X axis for the tool you want to define
Y_1	Y axis for the tool you want to define
Z_1	Z axis for the tool you want to define

Continues on next page

Continued

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

Editing the tool definition

	Action	Instance	Unit

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

2.	If necessary, enter the tool frame orientation.	`tframe.rot.q1`
		`tframe.rot.q2`
		`tframe.rot.q3`
		`tframe.rot.q4`
			None

3.	Enter the weight of the tool.	`tload.mass`	[kg]

4.	If necessary, enter the tool's center of gravity.	`tload.cog.x`
		`tload.cog.y`
		`tload.cog.z`
			[mm]

5.	If necessary, enter the orientation of the axis of moment	`tload.aom.q1`
		`tload.aom.q2`
		`tload.aom.q3`
		`tload.aom.q4`
			None

6.	If necessary, enter the tool's moment of inertia.	`tload.ix`
		`tload.iy`
		`tload.iz`
			[kgm^2]

7.	Tap **OK** to use the new values, **Cancel** to leave the definition unchanged.		

6.4.4. Editing the tool declaration

Tool declaration

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

Displaying the tool declaration

	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Tap **Tool** to see the list of available tools.
3.	Tap the tool you want to edit, then tap **Edit**. A menu appears. • Change Declaration • Change Value • Delete • Define
4.	In the menu, tap **Change Declaration**. The tool's declaration appears.
5.	Edit the tool declaration as listed in section *Creating a tool on page 144*.

NOTE!

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

6.4.5. Deleting a tool

Deleting a tool

This section describes how to delete a tool.

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Tool** to display the list of available tools.
3.	Tap the tool you want to delete, then tap **Edit**.
4.	Tap **Delete** to delete the selected tool. A confirmation dialog box appears.
5.	In the dialog box, tap **Yes** to delete the tool, **No** to keep the tool.

CAUTION!

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

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

6.4.6. Setup for stationary tools

Stationary tools

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

With stationary tools, the robot holds the work object.

Make a tool stationary

This section describes how to make a tool stationery.

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

Make a work object robot held

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

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

Differences in coordinate system referencing

This section describes differences in coordinate system referencing.

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

Continues on next page

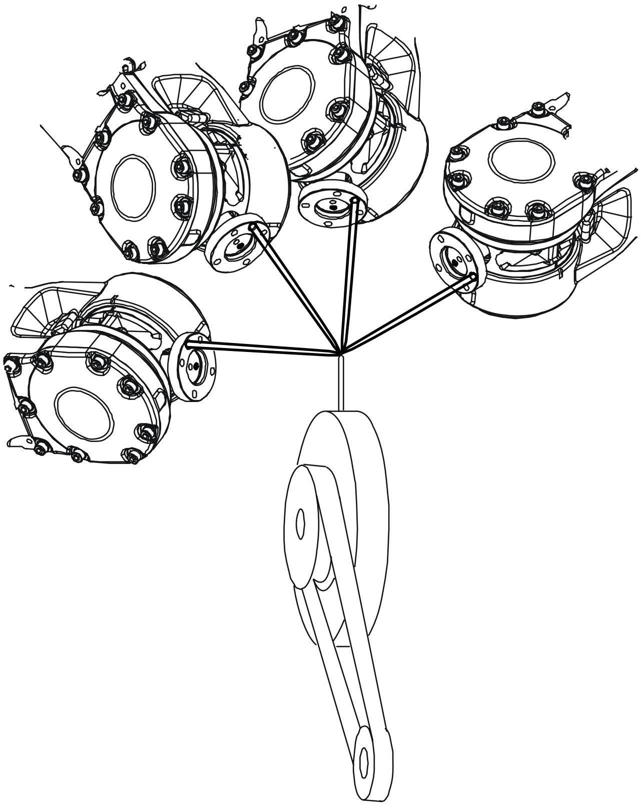
Set up the tool coordinate system

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

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

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

You may enter the stationary tool’s definition manually.



en0400000990

6.5 Work objects

6.5.1. Creating a work object

What happens when I create a work object?

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

See also *What is a work object? on page 322* for more details.

Creating a work object

The work object's coordinate system is now identical with the world coordinate system. To define the position and orientation of the work object's coordinate system, see *Editing the work object declaration on page 160*.

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

Work object declaration settings

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

6.5.2. Defining the work object coordinate system

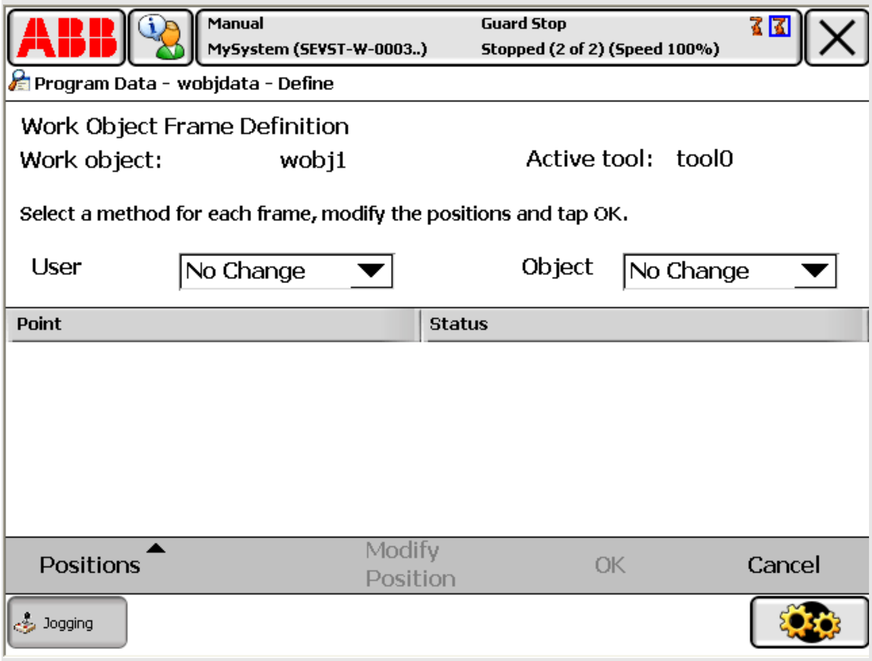
Overview

Defining a work object means that the robot is used to point out the location of it. This is done by defining three positions, two on the x-axis and one on the y-axis.

When defining a work object you can use either the user frame or the object frame or both. The user select frame and the object frame usually coincides. If not, the object frame is displaced from the user frame.

How to select method

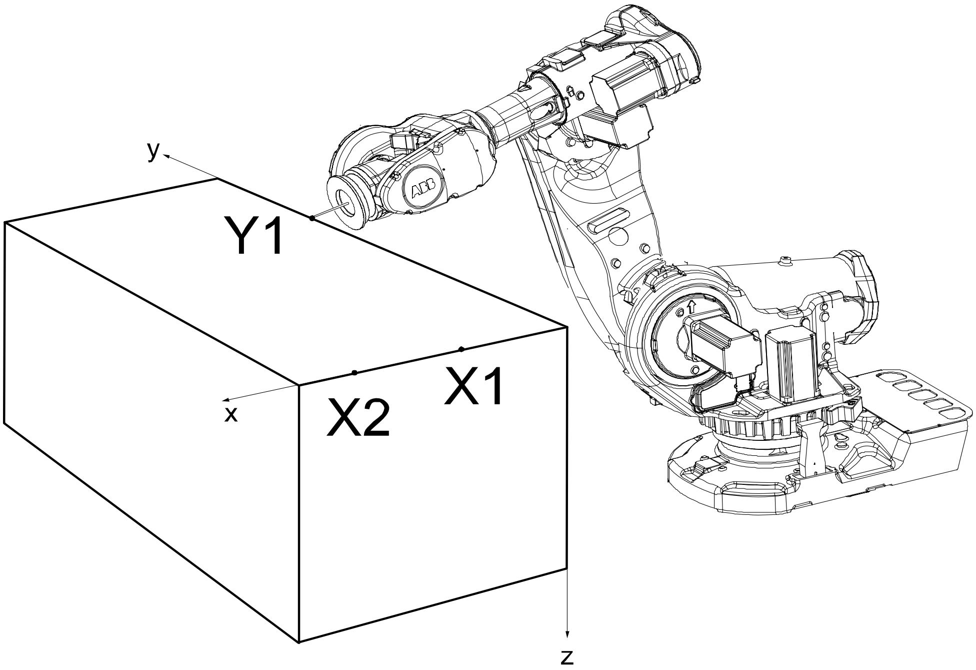
This procedure describes how to select method for defining either user frame or object frame or both. Note that this only works for a user created work object, not the default work object, wobj0. Defining work object can also be done from the Program Data window.

	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Tap **Work object** to display the list of available work objects.
3.	Tap the work object you want to define, then tap **Edit**.
4.	In the menu, tap **Define...**.
5.	Select method from the **User method** and/or the **Object method** menu. See *How to define the user frame on page 157* and *How to define the object frame on page 158*  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Program Data - wobjdata - Define Work Object Frame Definition Work object: wobj1 Active tool: tool0 Select a method for each frame, modify the positions and tap OK. User: No Change Object: No Change Point Status Positions Modify Position OK Cancel Jogging en0400000893

Continues on next page

How to define the user frame

This section details how to define the user frame.



en0400000887

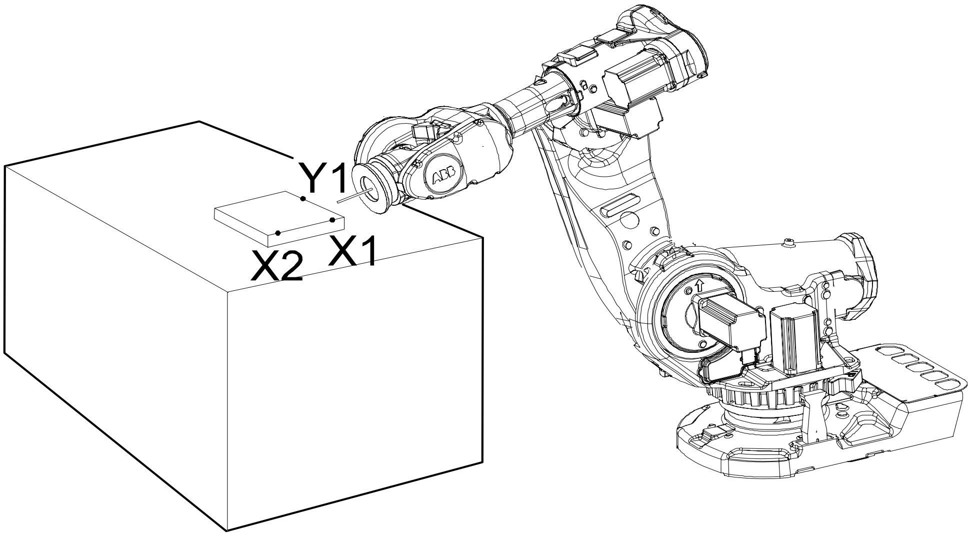
The x axis will go through points X1-X2, and the y axis through Y1.

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

Continues on next page

How to define the object frame

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



en0400000899

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

	Action
1.	In the **Object method** pop up menu, tap **3 points**.
2.	See steps 2 to 4 in the description of *How to define the user frame on page 157*.

6.5.3. Editing the work object data

Overview

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

How to display the work object data

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

How to set user and object frame values manually

The easiest way to set the work object and user coordinate systems position is to use the method described in *Defining the work object coordinate system on page 156*. You can however edit the values manually using the guide below.

Values	Instance	Unit

The cartesian coordinates of the position of the object frame	`oframe.trans.x`
	`oframe.trans.y`
	`oframe.trans.z`
		mm

The object frame orientation	`oframe.rot.q1`
	`oframe.rot.q2`
	`oframe.rot.q3`
	`oframe.rot.q4`
		-

The cartesian coordinates of the position of the user frame	`uframe.trans.x`
	`uframe.trans.y`
	`uframe.trans.z`
		mm

The user frame orientation	`uframe.rot.q1`
	`uframe.rot.q2`
	`uframe.rot.q3`
	`uframe.rot.q4`
		-

NOTE!

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

6.5.4. Editing the work object declaration

Overview

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

Displaying the work object declaration

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Work object** to see the list of available work objects.
3.	Tap the work object you want to edit, then tap **Edit**.
4.	In the menu, tap **Change Declaration**.
5.	The work object's declaration appears.
6.	Edit the tool declaration as listed in section *Creating a work object on page 155*.

NOTE!

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

6.5.5. Deleting a work object

Deleting a work object

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Work object** to display the list of available work objects.
3.	Tap the work object you want to delete, then tap **Edit**.
4.	Tap **Delete** to delete the work object. A confirmation dialog box appears.
5.	In the dialog box, tap **Yes** to delete the work object, **No** to keep it.

CAUTION!

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

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

6.6 Payloads

6.6.1. Creating a payload

What happens when I create a payload?

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

Adding a new payload and setting data declaration

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

	Action
1	In the **ABB** menu tap **Jogging**.
2	Tap **Payload** to display the list of available payloads.
3	Tap **New** to create a new payload. Enter data, see table below.
4	Tap **OK**.

Payload declaration settings

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

6.6.2. Editing the payload data

Overview

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

This can also be done automatically with the service routine LoadIdentify. See sections *Running a service routine on page 215*, or *LoadIdentify, load identification service routine on page 221*.

Displaying the payload definition

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

Changing the payload data

This procedure describes how to manually enter the payload data. This can also be done automatically by running the service routine LoadIdentify. How to run a service routine is described in section *Running a service routine on page 215*.

	Action	Instance	Unit
1.	Enter the weight of the payload.	`load.mass`	[kg]

2.	Enter the payload's center of gravity.	`load.cog.x`
		`load.cog.y`
		`load.cog.z`
			[mm]

3.	Enter the orientation of the axis of moment.	`load.aom.q1`
		`load.aom.q2`
		`load.aom.q3`
		`load.aom.q3`	

4.	Enter the payload's moment of inertia.	`ix`
		`iy`
		`iz`
			[kgm^2]

5.	Tap **OK** to use the new values, **Cancel** to leave the data unchanged.	-	-

6.6.3. Editing the payload declaration

Overview

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

Displaying the payload declaration

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Payload** to see the list of available payloads.
3.	Tap the payload you want to edit, then tap **Edit**.
4.	In the menu, tap **Change declaration**.
5.	The payload’s declaration appears. See *Creating a payload on page 162*.

NOTE!

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

6.6.4. Deleting a payload

Deleting a payload

	Action
1.	In the **ABB** menu, tap **Jogging**.
2.	Tap **Payload** to display the list of available payloads.
3.	Tap the payload you want to delete, then tap **Edit**.
4.	Tap **Delete**. A confirmation dialog box appears.
5.	In the dialog box, tap **Yes** to delete the payload, **No** to keep the payload.

CAUTION!

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

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

6.7 Programming

6.7.1. Handling of programs

Overview

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

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

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

How to create a new program *when no program is available* is detailed in section *Creating a new program on page 166*.

About program files

When saving a program to the controller hard disk, it is by default saved to the directory HOME in the system’s folder unless otherwise stated. How to set another default path is detailed in section *Setting default paths on page 299*.

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

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

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

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

Creating a new program

This section describes how to create a new program.

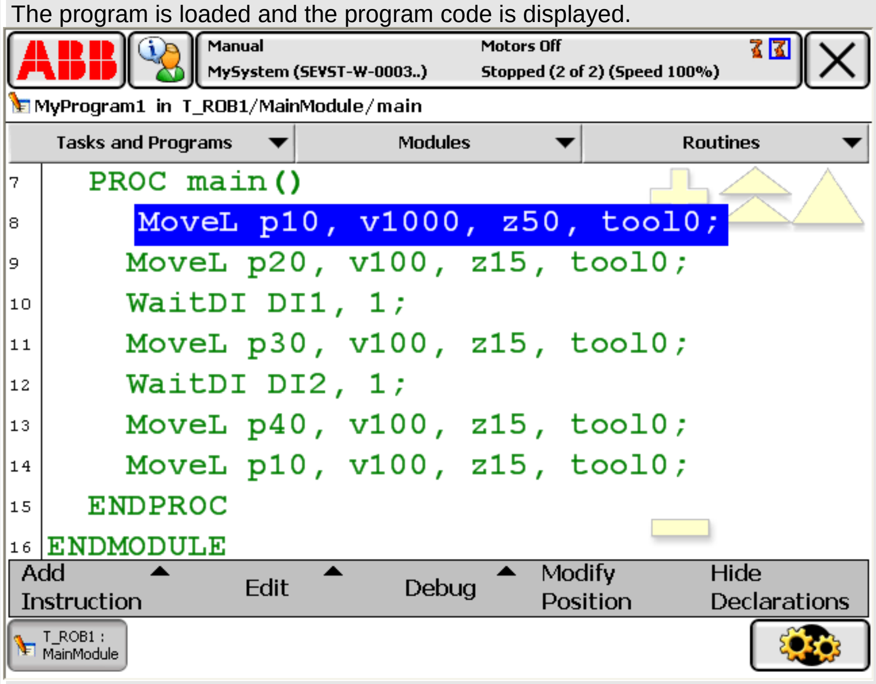
	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Tasks and Programs**.
3.	Tap **File**, then **New Program**. If there was already a program loaded, a warning dialog appears. • Tap **Save** to save the loaded program. • Tap **Don't save** to close loaded program without saving it, i.e. delete from program memory. • Tap **Cancel** to leave the program loaded.
4.	Use the soft keyboard to name the new program. Then tap **OK**.
5.	Continue by adding instructions, routines, or modules.

Continues on next page

Loading an existing program

This section describes how to load an existing program.

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Tasks and Programs**.
3.	Tap **File**, then **Load Program**. If there was already a program loaded, a warning dialog appears. • Tap **Save** to save the loaded program. • Tap **Don't save** to close loaded program without saving it, i.e. delete from program memory. • Tap **Cancel** to leave the program loaded.

4.	Use the file searching tool to locate the program file to be loaded (file type pgf). Then tap **OK**.
	The program is loaded and the program code is displayed.
	
	Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%)
	MyProgram1 in T_ROB1/MainModule/main
	`PROC main()`
	`  MoveL p10, v1000, z50, tool0;`
	`  MoveL p20, v100, z15, tool0;`
	`  WaitDI DI1, 1;`
	`  MoveL p30, v100, z15, tool0;`
	`  WaitDI DI2, 1;`
	`  MoveL p40, v100, z15, tool0;`
	`  MoveL p10, v100, z15, tool0;`
	`ENDPROC`
	`ENDMODULE`
	 Tasks and Programs
	Modules
	Routines
	Add Instruction
	Edit
	Debug
	Modify Position
	Hide Declarations
	T_ROB1 : MainModule
	en0400000699

Saving a program

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

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

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

Continues on next page

Renaming a loaded program

This section describes how to rename a loaded program.

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

6.7.2. Handling of modules

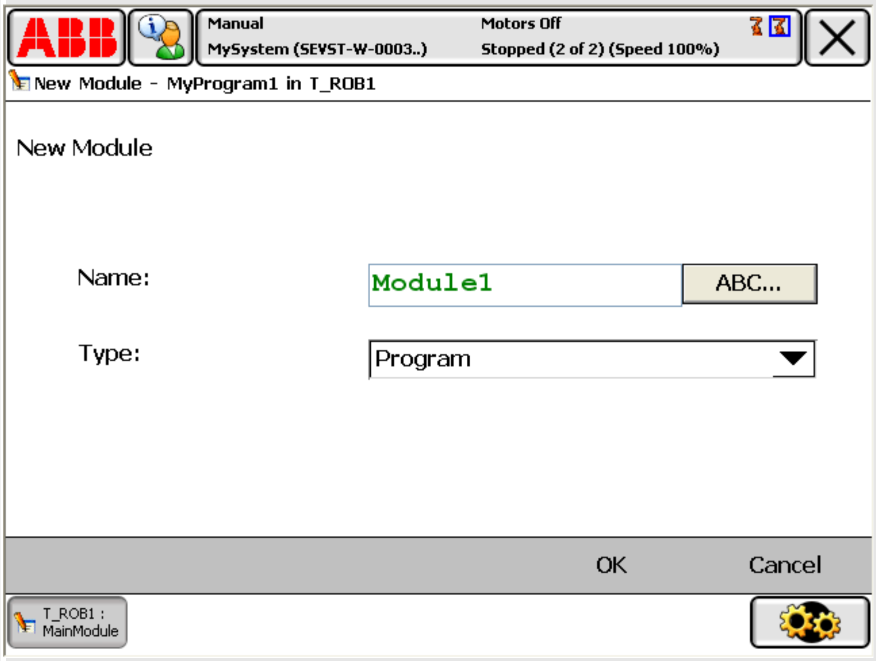
Overview

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

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

Creating a new module

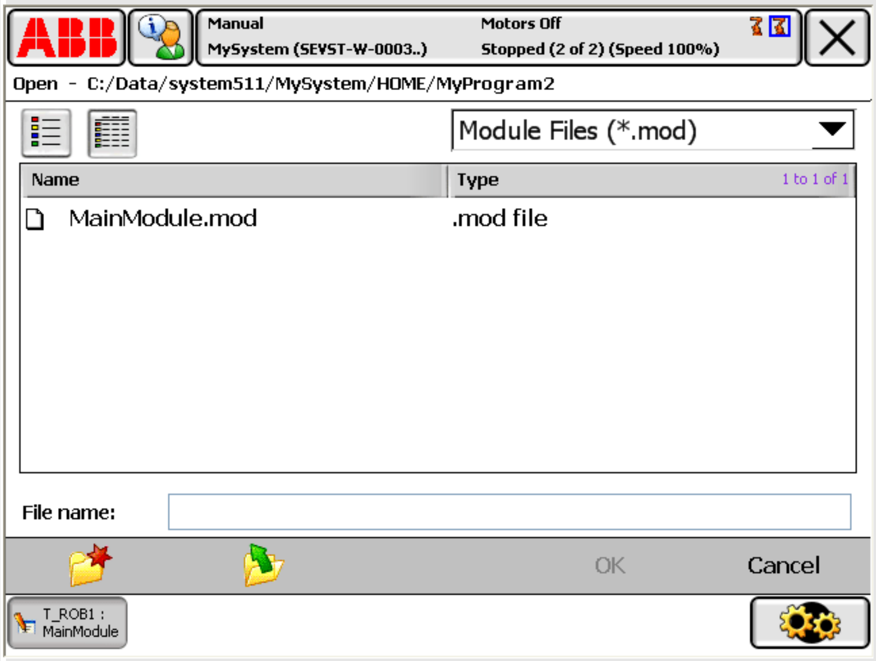
This section describes how to create a new module.

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Modules**.
3.	Tap **File**, then tap **New Module**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) ABC... OK Cancel T_ROB1 : MainModule New Module - MyProgram1 in T_ROB1 New Module Name: Module1 Type: Program en0400000688
4.	Tap **ABC...** and use the soft keyboard to enter the new module's name. Then tap **OK** to close the soft keyboard.
5.	Select which type of module to be created: • Program • System Then tap **OK**. The differences between module types are described in section *The structure of a RAPID application on page 134*. How to later switch between these types is detailed in section *Changing type of module on page 171*.

Continues on next page

Loading an existing module

This section describes how to load an existing module.

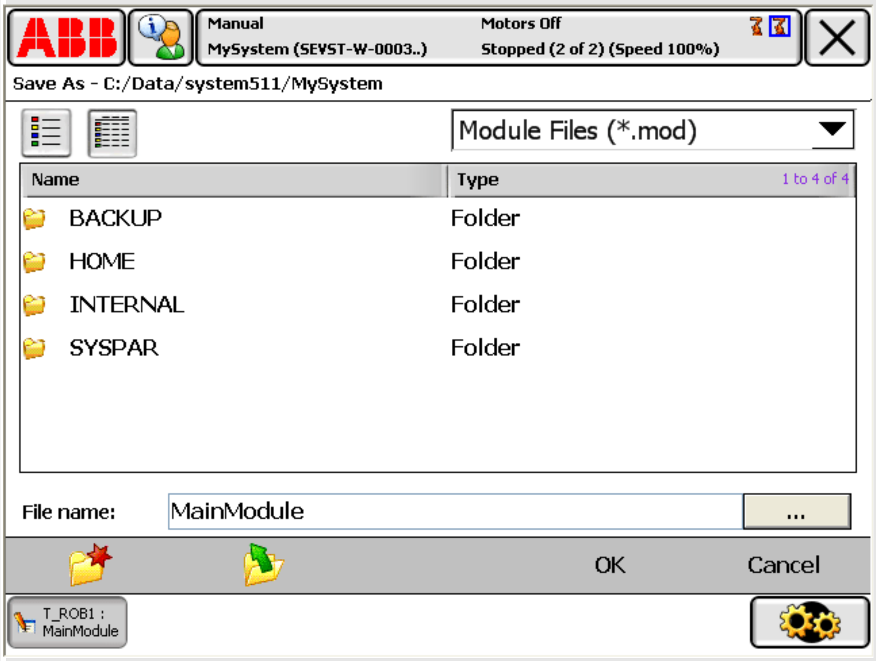
	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Modules**.
3.	Tap **File**, then **Load Module**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Open - C:/Data/system511/MySystem/HOME/MyProgram2 Module Files (*.mod) Name Type 1 to 1 of 1 MainModule.mod .mod file File name: OK Cancel T_ROB1 : MainModule en0400000689 Locate the module to be loaded. See section *FlexPendant Explorer on page 77*. A default path may be defined as detailed in section *Setting default paths on page 299*.
4.	Tap **OK** to load the selected module. The module is loaded.

Saving a module

This section describes how to save a module.

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Modules** and tap to select the module you want to load.

Continues on next page

	Action
3.	Tap **File**, then **Save Module As...**  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Save As - C:/Data/system511/MySystem Module Files (*.mod) Name | Type | 1 to 4 of 4 BACKUP | Folder HOME | Folder INTERNAL | Folder SYSPAR | Folder File name: MainModule ... OK Cancel T_ROB1 : MainModule en0400000690
4.	Tap on the suggested file name and use the soft keyboard to enter the module's name. Then tap **OK**.
5.	Use the file searching tool to locate where you want to save the module. See section *FlexPendant Explorer on page 77*. The default location is on the controller disk, but any other location may be set as default as detailed in section *Setting default paths on page 299*. Then tap **OK**. The module is saved.

Renaming a module

This section describes how to rename a module.

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

Changing type of module

This section describes how to change the type of module.

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Modules** and select the module to be changed.

Continues on next page

Continued

	Action
3.	Tap **File**, then **Change declaration...**
4.	Tap **Type** and select module type.
5.	Tap **OK**.

Deleting a module

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

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Modules** and tap to select the module you want to delete.
3.	Tap **File**, then **Delete Module...** A dialog box is displayed.
4.	Tap **OK** to delete the module without saving it. If you want to save the module first, tap **Cancel** and save the module first. How to save the module is detailed in section *Saving a module on page 170*.

6.7.3. Handling of routines

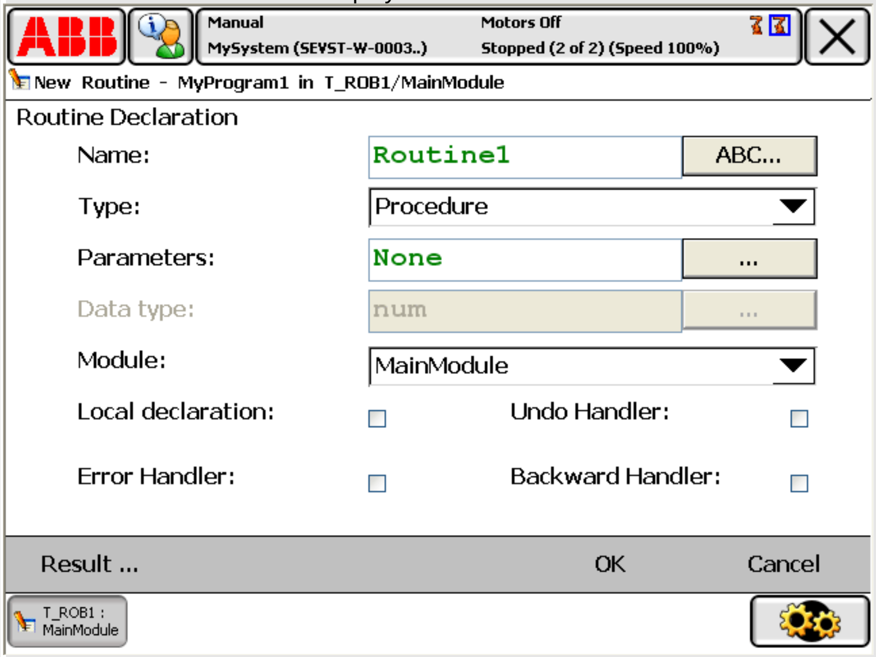
Overview

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

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

Creating a new routine

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

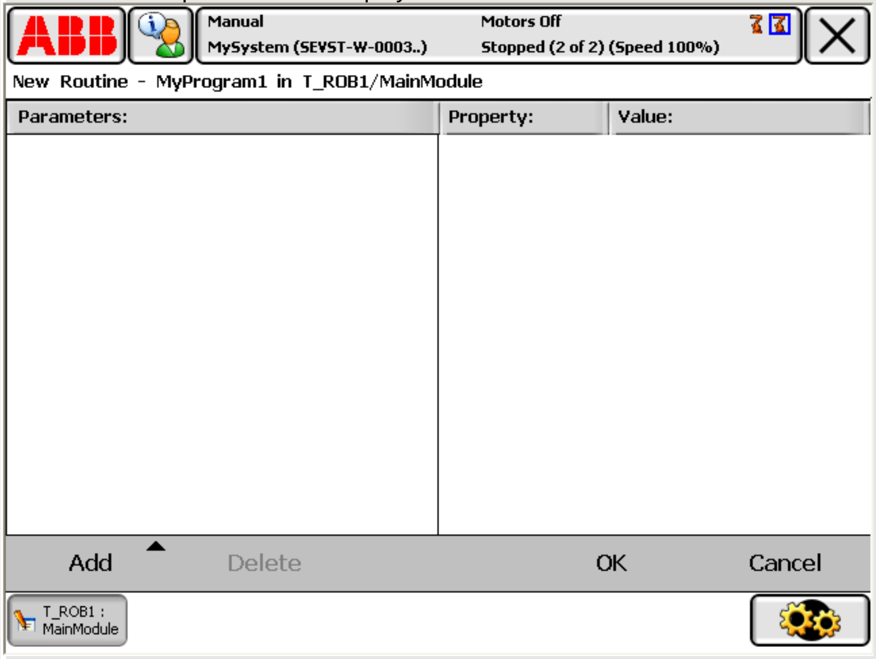
	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Routines**.
3.	Tap **File**, then **New Routine**. A new routine is created and displayed with default declaration values.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) New Routine - MyProgram1 in T_ROB1/MainModule Routine Declaration Name: Routine1 ABC... Type: Procedure Parameters: None ... Data type: num ... Module: MainModule Local declaration: □ Undo Handler: □ Error Handler: □ Backward Handler: □ Result ... OK Cancel T_ROB1 : MainModule en0400000692
4.	Tap **ABC...** and use the soft keyboard to enter the new routines' name. Then tap **OK**.
5.	Select the type of routine: • Procedure: used for a normal routine without return value • Function: used for a normal routine with return value • Trap: used for an interrupt routine
6.	Do you need to use any parameters? If YES; tap **...** and proceed as detailed in section *Defining parameters in routine on page 174*. If NO; proceed to the next step.
7.	Select module to add the routine to.

Continues on next page

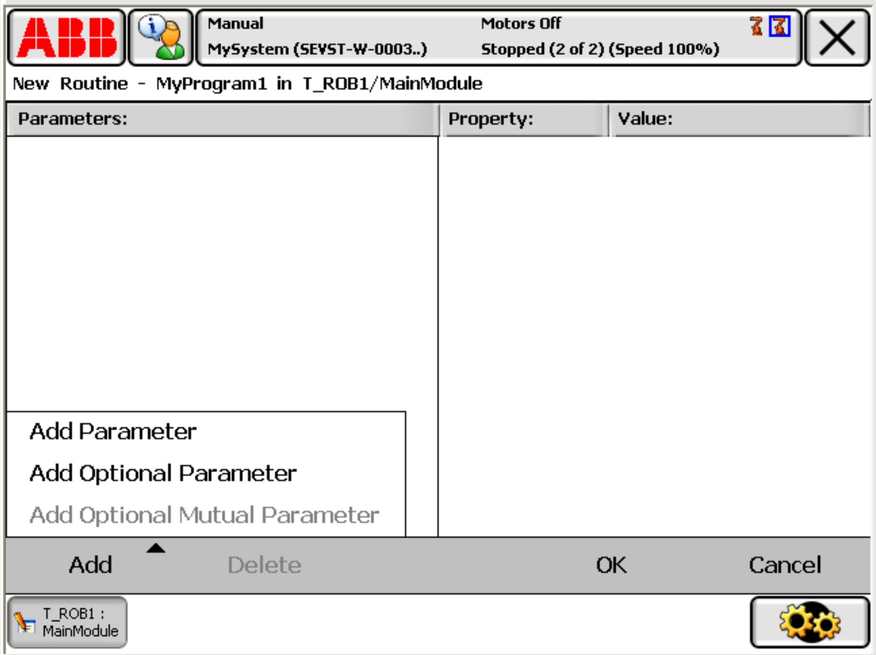
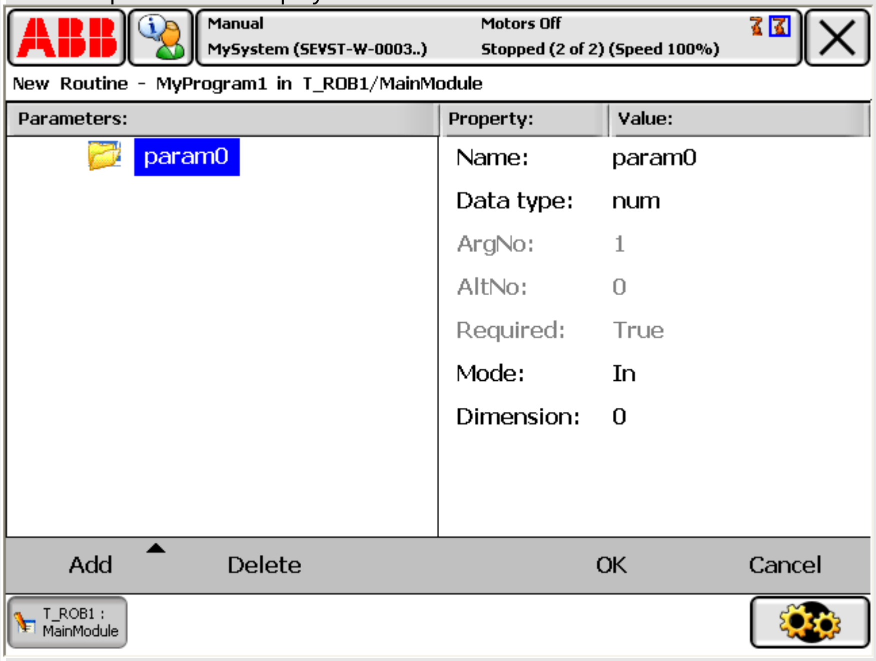
	Action
8.	Tap the checkbox to select **Local declaration** if the routine should be local. A local routine can only be used in the selected module.
9.	Tap **OK**.

Defining parameters in routine

This section describes how to define parameters in a routine.

	Action
1.	In the routine declaration, tap **...** to define parameters. A list of defined parameters is displayed.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) New Routine - MyProgram1 in T_ROB1/MainModule Parameters: Property: Value: Add Delete OK Cancel T_ROB1 : MainModule en0400000693

Continues on next page

	Action
2.	If no parameters are shown, tap **Add** to add a new parameter. • Add optional parameter adds a parameter that is optional • Add optional mutual parameter adds a parameter that is mutually optional with another parameter Read more about routine parameters in the RAPID reference manuals.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) New Routine - MyProgram1 in T_ROB1/MainModule Parameters: Property: Value: Add Parameter Add Optional Parameter Add Optional Mutual Parameter Add Delete OK Cancel T_ROB1 : MainModule en0400000695
3.	Use the soft keyboard to enter the name of the new parameter and then tap **OK**. The new parameter is displayed in the list.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) New Routine - MyProgram1 in T_ROB1/MainModule Parameters: Property: Value: param0 Name: param0 Data type: num ArgNo: 1 AltNo: 0 Required: True Mode: In Dimension: 0 Add Delete OK Cancel T_ROB1 : MainModule en0400000696
4.	Tap to select a parameter. To edit values, tap the value.
5.	Tap **OK** to return to the routine declaration.

Continues on next page

Creating a copy of a routine

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

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Routines**.
3.	Highlight the routine by tapping it.
4.	Tap **File**, then **Copy Routine**. The new routine is displayed. The name of the new routine is set to the same as the original with the suffix *Copy*.
5.	Make any changes in the declarations for the new routine copy. Then tap **OK**. How to make all declarations is detailed in section *Creating a new routine on page 173*.

Changing the declaration of a routine

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

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Routines**.
3.	Highlight the routine by tapping it.
4.	Tap **File**, then **Change Declaration**
5.	Change any declaration values for the routine. Then tap **OK**. Declaration settings are described in section *Creating a new routine on page 173*.

Moving a routine

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

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

Deleting a routine

This section describes how to delete a routine from memory.

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

Continues on next page

	Action
5.	Tap: • **OK** to delete the routine without saving any changes made to it. • **Cancel** to revert without deleting the routine.

6.7.4. Handling of instructions

Instructions

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

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

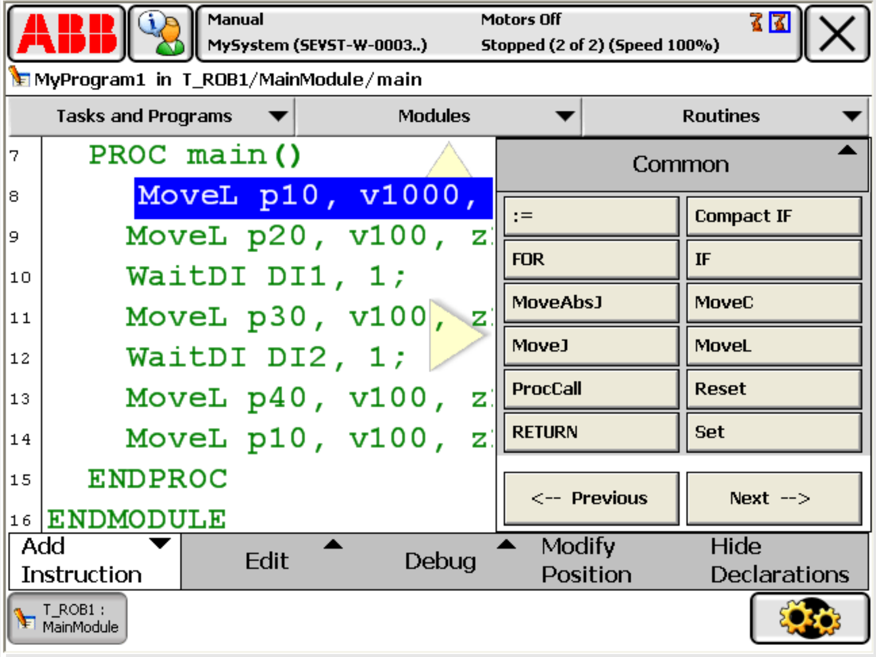
Undo and redo

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

Adding instructions

This section describes how to add instructions.

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap to highlight the instruction under which you want to add a new instruction.
3.	Tap **Add instruction**. A category of instructions is displayed.



Manual
MySystem (SEVST-W-0003..)
Motors Off
Stopped (2 of 2) (Speed 100%)

MyProgram1 in T_ROB1/MainModule/main

```
  PROC main()
    MoveL p10, v1000,
    MoveL p20, v100, z
    WaitDI DI1, 1;
    MoveL p30, v100, z
    WaitDI DI2, 1;
    MoveL p40, v100, z
    MoveL p10, v100, z
  ENDPROC
ENDMODULE
```

en0400000697

A large number of instructions, divided into several categories, are available. The default category is **Common**, where the most common instructions are listed.

You can create three personalized lists using the system parameters of the type *Most Common Instruction* in the topic *Man-machine Communication*. The system parameters are described in *Technical reference manual - System parameters*.

4.	Tap **Common** to display a list of the available categories. You can also tap **Previous/Next** at the bottom of the list of instructions to move to the next/previous category.

Continues on next page

Action

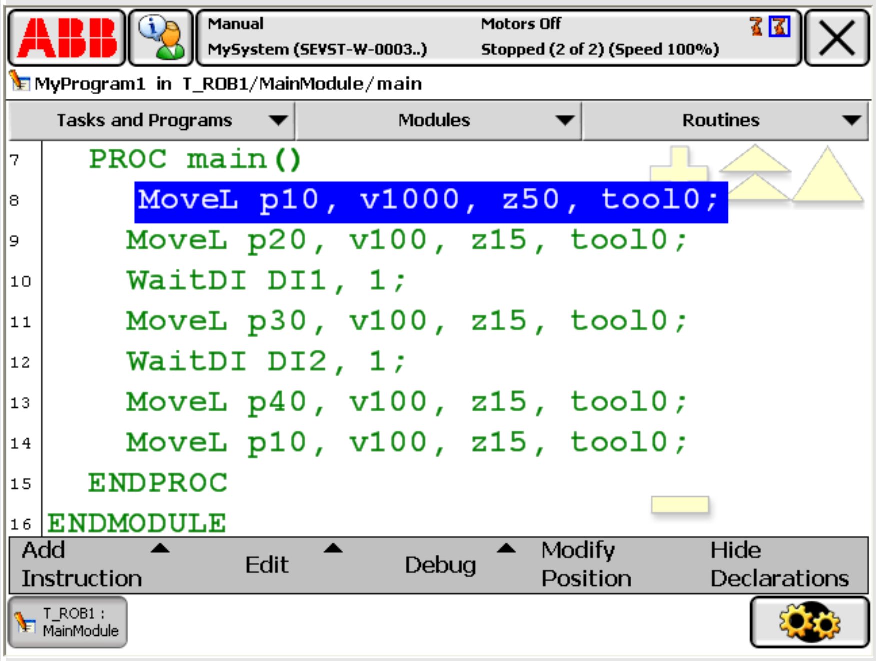
5. Tap the instruction you want to add.
   The instruction is added to the code.

Editing instruction arguments

This section describes how to edit instruction arguments.

Action

1. Tap the instruction to edit.



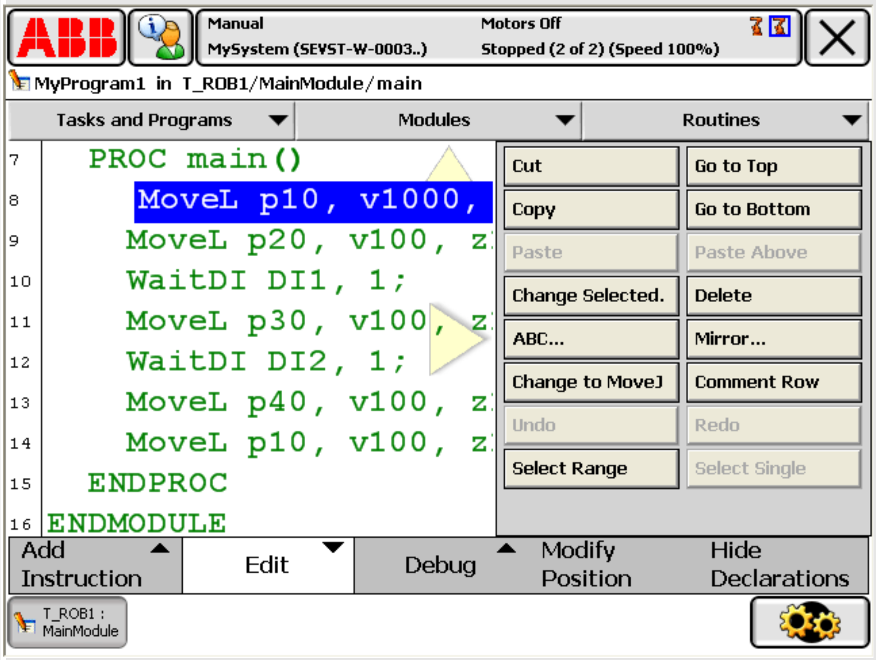
```
  PROC main()
    MoveL p10, v1000, z50, tool0;
    MoveL p20, v100, z15, tool0;
    WaitDI DI1, 1;
    MoveL p30, v100, z15, tool0;
    WaitDI DI2, 1;
    MoveL p40, v100, z15, tool0;
    MoveL p10, v100, z15, tool0;
  ENDPROC
ENDMODULE
```

en0400000699

Continues on next page

Action

2. Tap **Edit**.

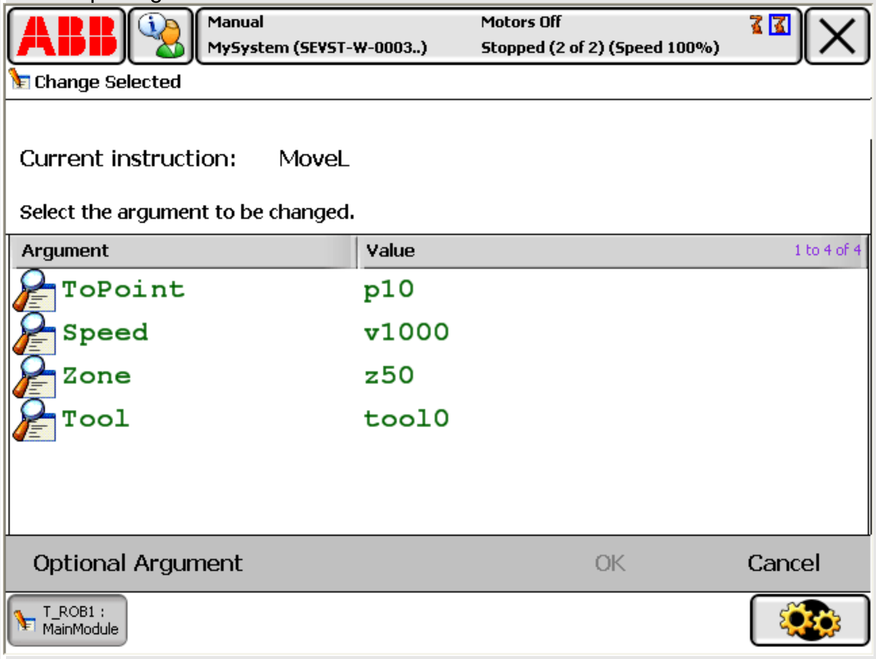


```
  PROC main()
    MoveL p10, v1000,
    MoveL p20, v100, z
    WaitDI DI1, 1;
    MoveL p30, v100, z
    WaitDI DI2, 1;
    MoveL p40, v100, z
    MoveL p10, v100, z
  ENDPROC
ENDMODULE
```

en0400000701

3. Tap **Change Selected**.

Depending on the type of instruction, the arguments have different data types. Use the soft keyboard to change string values or proceed to the next steps for other data types or multiple argument instructions.



Current instruction: MoveL

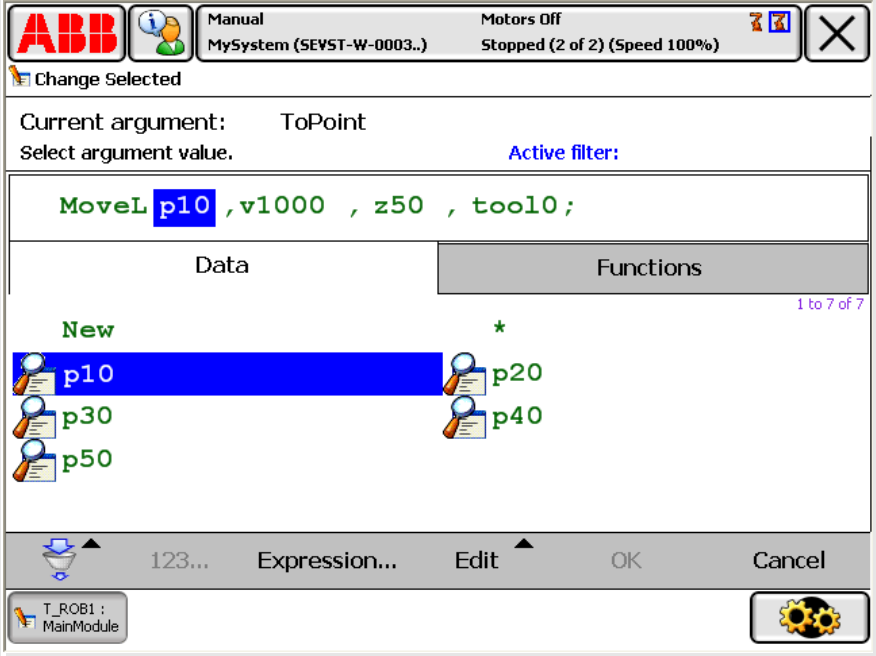
Select the argument to be changed.

Argument	Value
ToPoint	p10
Speed	v1000
Zone	z50
Tool	tool0

en0400000702

Continues on next page

Continued

	Action
4.	Tap the argument to be changed. A number of options are displayed.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Change Selected Current argument: ToPoint Select argument value. Active filter: MoveL p10 ,v1000 , z50 , tool0; Data Functions 1 to 7 of 7 New * p10 p20 p30 p40 p50 123... Expression... Edit OK Cancel T_ROB1 : MainModule en0400000703
5.	Tap an existing data instance to select and then tap **OK** to complete, or tap **Expression** See more about expressions in section *Editing instruction expressions and declarations on page 197*. To edit a particular data instance, see *Editing instruction expressions and declarations on page 197*.

TIP!

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

Copying and pasting instructions or arguments

This section describes how to paste instructions or arguments.

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

Continues on next page

Cutting an instruction

This section describes how to cut an instruction.

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

Changing motion mode for a move instruction

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

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

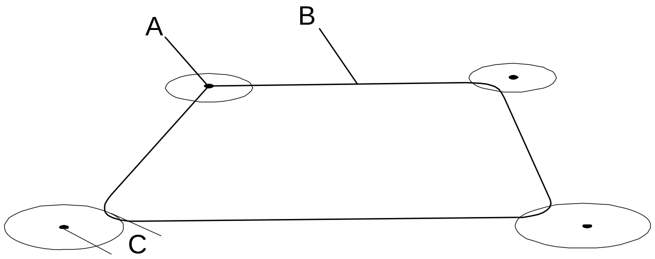
Commenting instruction rows

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

6.7.5. Example: Add movement instructions

Overview

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



en0400000801

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

Add movement instructions

This section details how to add movement instructions.

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

Result

Your program code should look like this:

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

6.8 Advanced programming

6.8.1. Mirroring a program, module, or routine

Mirroring

Mirroring creates a copy of a program, module, or routine in a specific mirror plane. The mirror function can be applied to any program, module, or routine.

Mirroring can be performed in two different ways:

- Default against the base frame coordinate system. The mirror operation will be performed across the xz-plane in the base frame coordinate system. All positions and work object frames that are used in an instruction in the selected program, module or routine are mirrored. The position orientation axes x and z will be mirrored.
- Advanced against a specific mirror frame. The mirror operation will be performed across the xy-plane in a specified work object frame, mirror frame. All positions in the selected program, module or routine are mirrored. If the work object argument in an instruction is another work object than specified in the mirror dialog, the work object in the instruction is used in the mirror operation. It is also possible to specify which axis in the position orientation that will be mirrored, x and z or y and z.

Mirroring is described in section *What is mirroring? on page 329*.

Mirroring a routine

This section describes how to mirror a routine.

	Action
1.	In the **ABB** menu, tap **Program Editor**.
2.	Tap **Edit** and tap **Mirror**.
3.	To define the mirror. • Tap the **Module** menu to select in which module the routine to mirror is used. • Tap the **Routine** menu to select which routine you want to mirror. • Tap **...** to open the soft keyboard and enter the name for the new routine.
4.	If you want to mirror in base frame then proceed to the next step. If you want to define another type of mirror then tap **Advanced options** and proceed as follows. To define the type of mirror: • Deselect the **Base Mirror** checkbox. • Tap **...** to the right of **Work object** to select the work object frame to which all positions which are to be mirrored are related to. • Tap **...** to the right of **Mirror frame** to select the mirror plane to which all positions will be mirrored. • Tap the **Axis to mirror** menu to specify how to mirror the position orientation. x means that x and z axes will be mirrored. y means that y and z axes will be mirrored. • Tap **OK** to save the advanced options.
5.	Tap **OK**. A dialogue box is displayed.
6.	Tap **Yes** to apply the selected mirror to the routine, or tap **No** to cancel.

Continues on next page

Mirroring a module or program

This section describes how to mirror a module or program.

	Action
1.	In the **ABB** menu, tap **Program Editor**.
2.	Tap **Edit** and tap **Mirror**.
3.	To define the mirror. • Tap the **Module** menu to select module to mirror. • Tap **...** to open the soft keyboard and enter the name for the new module or program.
4.	If you want to mirror in base frame then proceed to the next step. If you want to define another type of mirror then tap **Advanced options** and proceed as follows. To define the type of mirror: • Deselect the **Base Mirror** checkbox. • Tap **...** to the right of **Work object** to select the work object frame to which all positions which are to be mirrored are related to. • Tap **...** to the right of **Mirror frame** to select the mirror plane to which all positions will be mirrored. • Tap the **Axis to mirror** menu to specify how to mirror the position orientation. x means that x and z axes will be mirrored. y means that y and z axes will be mirrored. • Tap **OK** to save the advanced options.
5.	Tap **OK**. A dialogue box is displayed.
6.	Tap **Yes** to apply the selected mirror to the module, or tap **No** to cancel.

6.8.2. Modifying and tuning positions

Overview

Positions are instances of the data type robtarget or jointtarget. See *Technical reference manual - RAPID Instructions, Functions and Data types*.

The positions can be tuned using the function HotEdit, where you enter offset values using a soft keyboard.The offset value is used together with the original position value. See *Tuning positions with HotEdit on page 190*. The HotEdit menu is described in section *HotEdit menu on page 75*.

The positions can also be modified using the Modify positions function in the Program Editor or the Production Window where you step and jog the robot to the new position. A modified position value overwrites the original value. See *Modifying positions in the Program Editor or Production Window on page 187*.

CAUTION!

Changing programmed positions may significantly alter the robot’s movement pattern.

Always make sure any changes are safe for both equipment and personnel.

Positions in arrays

If a position is declared as an array, then the procedure for modifying or tuning may differ slightly depending on how the array is indexed in the move instruction.

See more information about arrays in *What is a data array? on page 334*.

Limitations

Note that jointtargets can only be modified using the Modify positions method in the Program Editor and the Production Window, i.e. not with HotEdit.

NOTE!

Your system can have restrictions on how positions can be modified. Restrictions can apply to distance using system parameters (topic *Controller*, type *ModPos Settings*) and which positions can be modified using UAS.

6.8.3. Modifying positions in the Program Editor or Production Window

Overview

When modifying positions by jogging the robot to the new position you can either single-step through the program to the position(s) you want to modify, or jog directly to the new position and change the corresponding position argument of the instruction.

The recommendation is to step through the program to the position, but if you know your robot program well and the new position is known, it is faster to use the jogging method. **Note!** Do not use this method to change orientation values.

Prerequisites

To modify positions using the Program Editor or Production Window, the system must be in manual mode. To modify positions in the Production Window, you must have started the program so that the motion pointer is set.

Applying modified positions

The modified position values will normally be used when you restart the program. If the robot cannot use the values directly at start, a warning is displayed. Then the modified position will be used the next time that the position is used in the program.

Modifying positions

This procedure describes how to modify positions, either by single-stepping to the positions or jogging. You can use the Program Editor or the Production Window, the functionality is the same.

	Action	Info
1.	On the **ABB** menu, tap **Program Editor**.	
2.	Stop the program, if running.	
3.	Do you want to single-step to the position or jog? If *single-stepping*, step through the program to the position you want to change. Make sure the correct argument is selected. If *jogging*, use the **Jogging** view to make sure that the same work object and tool that are used in the instruction are selected.	When single-stepping, if the instruction or procedure call has more than one position argument, continue to step to reach each argument.
4.	Jog to the new position.	
5.	When using the jogging method, tap to select the position argument you want to change.	
6.	In the Program Editor, tap **Modify Position**. In the Production Window, tap **Debug** and then **Modify Position**. A confirmation dialog appears.	When modifying a position in an array that is indexed with a variable you will have to select which element in the array to modify before the modification is executed.

Continues on next page

	Action	Info
7.	Tap **Modify** to use the new position, **Cancel** to keep the original.	If you select the check box **Don't show this dialog again** in the confirmation dialog, then you will not get any more confirmation dialogs when modifying positions. **Note!** This is only valid for the current Program Editor.
8.	Repeat step 3 through 7 for each position argument you want to change.	

Limitations

The Modify position button in the Program Editor is disabled until you select a position argument (that is possible to modify).

The Modify position button in the Production Window is disabled until the motion pointer is set and a position is selected. To set the motion pointer, the program must be started and then stopped.

The maximum movement or change in orientation, may be restricted by the system parameters (topic *Controller*, type *ModPos Settings*) in the system design. Please read your cell or plant documentation for details.

If the system parameters are setup to use absolute limits for position changes, then the original positions can only be restored or changed using the baseline menu in HotEdit. The baseline concept is described in section *Tuning positions with HotEdit on page 190*.

If a named position is modified, all other instructions using that position will be affected.

Differences between Program Editor and Production Window

The procedure for modifying positions is the same in the Program Editor and the Production Window. However, there are differences in how positions are selected.

Also, if your system uses MultiMove, then the result from the Program Editor and the Production Window will differ. See *Application manual - MultiMove*.

Program Editor selections

To select a position for modification in the Program Editor, tap the desired position.

Production Window selections

To select a position for modification in the Production Window you must step the program to the desired position.

Note!

If you have executed the program from another window and then switch back to the Production Window, the selected position will be changed to the position where the motion pointer now is. Make sure the correct position is selected before making the modification!

Continues on next page

Related information

For an overview on how to modify positions, see *Modifying and tuning positions on page 186*.

HotEdit and baseline are described in *Tuning positions with HotEdit on page 190*.

The HotEdit menu is also described in *HotEdit menu on page 75*.

Modifying positions in the Program Data window is described in *Editing data instances on page 140*.

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

Technical reference manual - System parameters.

Application manual - MultiMove

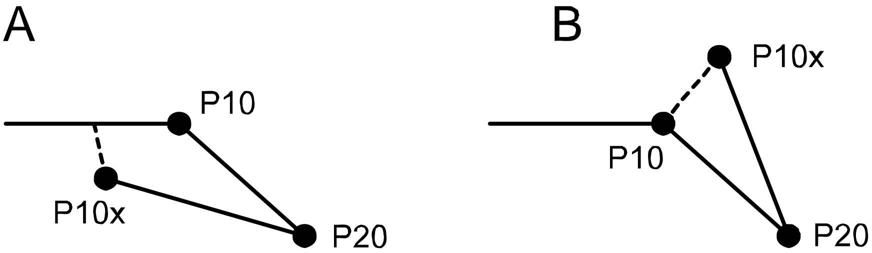
Examples planned path

The following examples show how the planned path will be effected when modifying positions.

Linear movement

In example A the robot is stopped on path before reaching the position P10. The robot is jogged off path to the new position (P10x) and the position P10 is modified.

In example B the robot is stopped on path in position P10. The robot is jogged off path to the new position (P10x) and the position P10 is modified.

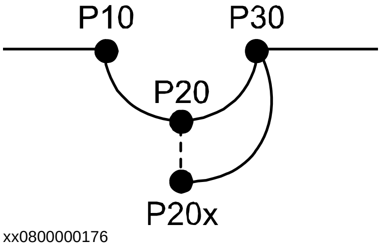


xx0800000175

In both examples, when restarting the program the robot continues from the new P10 (that is now the same as P10x) directly to P20 without returning to the previous planned path (via the old P10).

Circular movement

In this example the robot is stopped on path in position P20 and then jogged to the new position P20x. The position P20 is modified.



xx0800000176

When restarting the program the robot continues directly from the new P20 (that is now the same as P20x) to P30 without returning to the previous planned path (via the old P20). The new planned path from P20 (P20x) to P30 is calculated using these two positions and position P10.

6.8.4. Tuning positions with HotEdit

Overview

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

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

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

The HotEdit menu is described in the section *HotEdit menu on page 75*.

Applying tuned positions

Tuning values are used directly by an executing program when you tap **Apply**. If tuning is done close to the program or motion pointer it may be hard to predict exactly when it will take effect. It is therefore important that you know where in the program the robot is if applying offset values while the program is running.

However, the new values are not stored in the baseline until you use a **Commit** command.

How to tune positions

This is how you tune programmed positions using HotEdit:

	Action
1.	In the **Programmed targets** window, select the positions to be tuned and add them to **Selected targets** by tapping the arrow.
2.	Tap **Tune Targets** and select tuning mode (linear, reorient, or external axes) and then coordinate system (tool or work object).
3.	Tap + and - to specify the exact tuning of the position(s) in x, y and z direction. Select **Increment** to define the step size of these buttons.
4.	To activate the new values, tap **APPLY**. The offset will be used directly if the program is running.
5.	If you are satisfied with the result and want the tuned positions to become part of the baseline, tap **Baseline** and then **Commit Selection**.
6.	If, however, the selected targets need further tuning, you can tap **Baseline** and then **Restore Selection** and start all over again, or you can simply continue tuning until you are satisfied.

Working with selections

A selection of positions to be tuned later can be saved on the controller mass memory unit. If your system uses UAS, this may be the only way to select positions for tuning.

The commands for working with selections are located in the **File** menu:

Save Selection As	Make sure that the window **Selected targets** shows nothing but the positions to be saved. Tap **File** and **Save Selection As**. Enter the name and optionally a description of the file, then tap **OK**.
Open Selection	Tap **File** and **Open Selection**. Then tap the selection you want to use and tap **OK**.
Clear Selection	Clear the **Selected targets** area by tapping **File** and **Clear Selection**.

Continues on next page

Baseline concept

A baseline can be defined as a reference against which future changes are measured. The baseline concept makes it possible to undo tuning and revert to the position values stored in the latest baseline. To do this you use a **Restore** command.

When a **Commit** command is performed the baseline is updated with new offset values, and the old values no longer exist in program memory.

Use the baseline menu to apply or reject tuning.

- **Restore Selection** will discard all tuning of the currently selected positions and revert them to the values of the latest baseline, meaning that their offset values will be 0,0.
- **Restore Entire Program** will discard ALL tuning to programmed positions since the latest **Commit** command. This may include several HotEdit sessions for the same task. If the system uses *Absolute Limit ModPos* any **Modify Position** command from the Program Editor will also be undone.
- **Commit Selection** will apply the offset of the currently selected positions to the baseline.
- **Commit Entire Program** will apply ALL tuning to programmed positions. This may include several HotEdit sessions for the same task. If the system uses *Absolute Limit ModPos* it also includes **Modify Position** performed in the Program Editor.

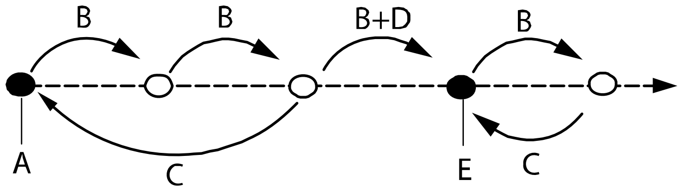
Baseline target criteria

Targets that fulfil all of the following criteria are part of the baseline:

- The data type must be robtarget or jointtarget
- It must **not** be declared locally in a routine
- It must **not** be declared as part of an array of targets

Illustration of baseline concept

The baseline concept is illustrated below, where a point is moved, restored and committed. Starting out from the original baseline (A), let us assume that you move the point (B) twice. If you regret the changes you perform a restore command (C). But if you instead continue moving the point and perform a commit command (B +D), you will have created a new baseline (E) and there is no way to revert to the original baseline. If you move the point one more time and then restore, the point is moved back to the latest baseline (E).



xx0600002620

A	Original baseline
B	Move selected point
C	Restore
D	Commit
E	New baseline

Continues on next page

Continued

Restore Selection or Restore Entire Program

The following example shows the difference between **Restore Selection** and **Restore Entire Program** to original. The same idea applies for **Commit Selection** and Commit Entire Program.

	Action
1.	The robtargets p10 and p30 are added to **Selected Targets** and tuned once.
2.	p10 is removed from **Selected Targets**
3.	p30 is tuned again.
4.	• **Restore Selection** sets the currently selected position, p30, to its value in the latest baseline. p10 is not affected, thus still tuned. • **Restore Entire Program** sets all tuned positions, that is both p10 and p30 to their baseline values.

HotEdit for external axes

External axes can be tuned with HotEdit if they are activated in at least one of the selected robtargets. Only axes with active values are tuned.

Limitations

HotEdit tuning is only possible for named (e.g. p10, p20) robtargets. (* robtargets are not visible in the treeview.)

Only a robtarget declared as an array and indexed with a number can be modified in HotEdit.

It is only possible to perform HotEdit tuning on targets that are part of the baseline. Targets that are NOT part of the baseline will not be shown in the HotEdit treeview, as they cannot be selected for tuning. This means that a target declared locally in a routine, for example, will not be displayed.

HotEdit tuning is possible for robtargets only. (Jointtargets can only be tuned by using Modify Position in the Program Editor.) If the system uses *Absolute limit ModPos* these jointtargets are however part of the baseline and will be affected when **Restore Entire Program** and **Commit Entire Program** are used.

NOTE!

For more information about *Absolute Limit ModPos*, see the *Technical reference manual - System parameters*, section Topic *Controller* - Type *ModPos Settings*.

Using UAS in HotEdit

The user authorization system can be used to restrict the Hot Edit functionality and only allow a user to edit pre-selected positions. These are loaded by tapping **File** and then **Open Selection**. The selected positions can then be tuned in the usual way.

Related information

Technical reference manual - System parameters.

6.8.5. Working with displacements and offsets

About displacements

Sometimes, the same path is to be performed at several places on the same object, or on several work pieces located next to each other. To avoid having to reprogram all positions each time a displacement coordinate system can be defined.

This coordinate system can also be used in conjunction with searches, to compensate for differences in the positions of the individual parts.

The displacement coordinate system is defined based on the object coordinate system.

The displacement coordinate system is described in section *What is a coordinate system? on page 323*.

Select displacement method

Depending on how, when, and how often you want to use displacements, the best method may vary.

Moving a work object

Moving a work object is suitable when you do not need to move or displace the work object very often.

See procedure in section *Defining the work object coordinate system on page 156*.

Displace a work object

A work object consists of a user frame and a object frame. You can move one or both of these frames. If you move both frames, then the whole work object is moved. It can be useful to displace the object frame from the user frame for instance when using one fixture for several work objects. Then you can keep the user frame and displace the object frame for the work objects.

See procedure *How to define object frame* in section *Defining the work object coordinate system on page 156*.

Displace and rotate a work object

You may want to displace and rotate the object frame from the user frame if the displacement is not in just x, y, and z.

To displace in x, y, and z, you can use the same method as above. To rotate the work object, follow the procedure in section *Editing the work object data on page 159*.

About offsets

Sometimes it is easier to define a position as an offset from a given position. If, for example, you know the exact dimensions of a work object, it will only be necessary to jog to one position.

The offset is programmed with the displacement distance in x, y, and z direction, in relation to the work object. For instance:

```
MoveL Offs(p10, 100, 50, 0), v50...
```

Continues on next page

Define the offset for the position with the following expressions:

1. Original position / starting point
2. Displacement in x direction
3. Displacement in y direction
4. Displacement in z direction

Examples

This example shows the move instructions with offsets to move the robot in a square (clockwise), starting at p10, with a 100 mm displacement in x and y.

```
MoveL p10, v50...
MoveL Offs(p10, 100, 0, 0), v50...
MoveL Offs(p10, 100, 100, 0), v50...
MoveL Offs(p10, 0, 100, 0), v50...
MoveL p10, v50...
```

How to create position offsets

This procedure details how to change a position to become an offset position.

	Action	Info
1.	In the Program Editor, tap to select the position argument to edit.	
2.	Tap **Edit** and then **Change Selected**.	
3.	Tap **Functions** and then **Offs**.	
4.	Tap to select each expression, **<EXP>**, and then tap any of the desired available data or functions. You can also tap **Edit** to access more functions. Tap **All** to open the soft keyboard and edit all expressions at the same time, or tap **Only Selected** to edit one at a time with the soft keyboard.	You can use the filter to narrow down the available data. You can also change data type of the available data. See more information about expressions in section *Editing instruction expressions and declarations on page 197*.
5.	Tap OK to save changes.	

Related information

There are a number of functions in RAPID that may be useful. See *Technical reference manual - RAPID Instructions, Functions and Data types*, and *Technical reference manual - RAPID overview*.

6.8.6. Moving the robot to a programmed position

Positions

A robot program usually contain programmed positions. The robot can move automatically to a programmed position using a function in the Jogging menu.

The robot will move at 250 mm/s.

⚠ **DANGER!**

When moving the robot automatically, the robot arm may move without warning. Make sure no personnel are in safeguarded space and that no objects are in the way between the current position and the programmed position.

Moving the robot to a programmed position

This procedure describes how to move a robot automatically to a programmed position.

	Action	Info
1.	On the **ABB** menu, tap **Jogging**.	
2.	Make sure the correct mechanical unit is selected and then tap **Go To...**.	
3.	Tap to select a programmed position.	If you have many programmed positions you can use a filter to narrow down the visible positions. See section *Filtering data on page 101*.
4.	Press and hold the enabling device and then tap and hold the **Go To** button. The robot now moves directly from the current position to the programmed position. Make sure no objects are in the way.	

6.8.7. Aligning tools

Overview

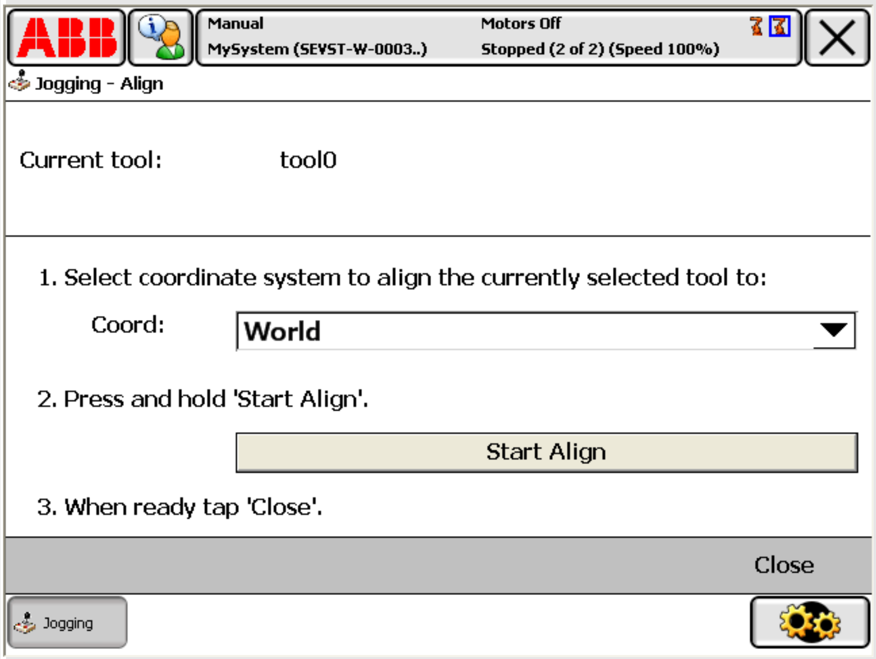
A tool can be aligned with another coordinate system.

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

Note that the tool’s data is not changed!

Aligning mechanical units

This procedure describes how to align tools.

	Action
1.	On the **ABB** menu, tap **Jogging**.
2.	Make sure that the right tool is active and then tap **Align...**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Jogging - Align Current tool: tool0 1. Select coordinate system to align the currently selected tool to: Coord: World 2. Press and hold 'Start Align'. Start Align 3. When ready tap 'Close'. Close Jogging en0500001548
3.	Select a coordinate system to align the selected tool to.
4.	Press and hold the enabling device and then tap and hold **Start Align** to start aligning the tool.
5.	Tap **Close** when completed.

6.8.8. Editing instruction expressions and declarations

Expressions

An expression specifies the evaluation of a value. It can be used, for example:

- as a condition in an IF instruction
- as an argument in an instruction
- as an argument in a function call

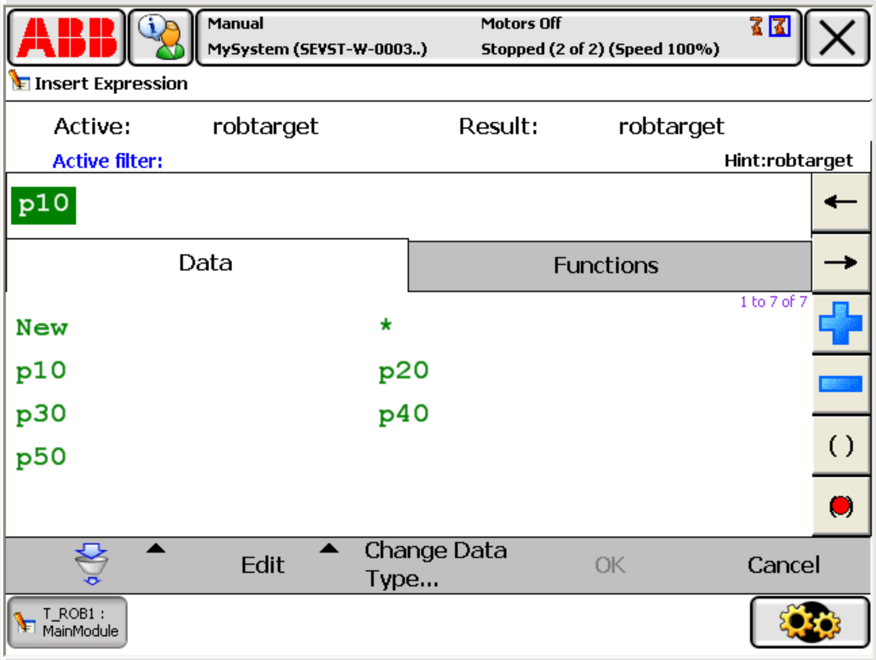
Read more in *Technical reference manual - RAPID overview* and *Technical reference manual - RAPID Instructions, Functions and Data types*.

Inserting expressions

This procedure describes how to insert and edit expressions in instructions.

Action

1. In the **Program Editor**, tap to select the instruction you want to edit and then tap **Edit**.
2. Tap **Change Selected** and tap to select the argument to change.
3. Tap **Expression**.



Active: robtarget Result: robtarget

Active filter: Hint:robtarget

p10

en0400000704

4. Edit the length of the expression by tapping the keys to the right:
   - Arrows: step backward and forward in the expression.
   - + to add expression. Tap the new expression to define it.
   - - to delete expression.
   - () to set a parenthesis around the highlighted expression.
   - (o) to delete a parenthesis.

Continues on next page

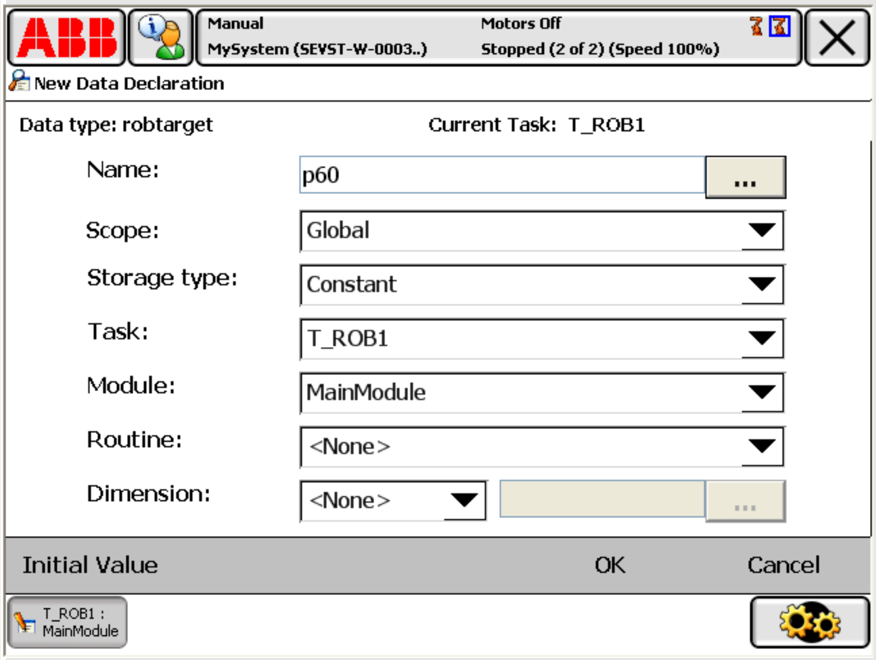
	Action
5.	Tap: - **New** to create a new data declaration, i.e. adding a data declaration not previously used. This is detailed in section *Creating new data declarations on page 198*. - **View** to change views or change data type. This is detailed in section *Changing data type on page 199*. - **ABC** displays the soft keyboard.
6.	Tap **OK** to save the expression.

Declarations and data types

When editing an expression new data can be declared with the button **New**. More information about data declarations and how to edit them can be found in section *Editing data instances on page 140*.

Creating new data declarations

This procedure describes how to create a new data declaration in an instruction expression.

	Action
1.	In the Insert Expression view, tap **New**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) New Data Declaration Data type: robtarget Current Task: T_ROB1 Name: p60 Scope: Global Storage type: Constant Task: T_ROB1 Module: MainModule Routine: <None> Dimension: <None> en0400000705

Continues on next page

Action

2. Tap and enter desired values:
   - **Initial value** to set the initial value.
   - **...** to display the soft keyboard and change the data type's name.
   - **Scope**
   - **Storage type**
   - **Module**
   - **Routine**
   - **Dimension** to set the size of an array if the data type should be an array.
   - If a value has been chosen for Dimension, tap **...** to set array size, see *What is a data array? on page 334*
3. After making all selections, tap **OK**.

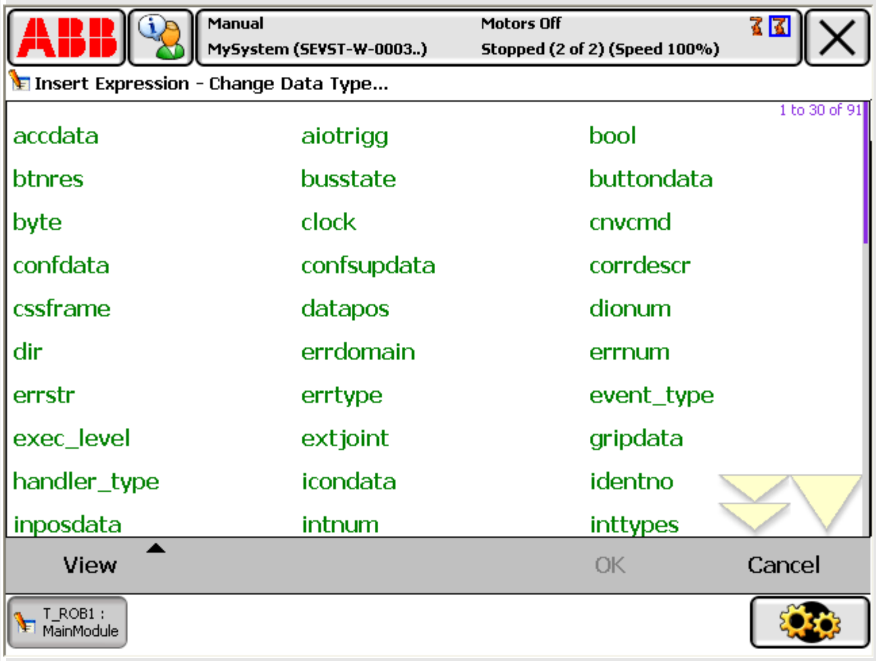
   A dialog box is displayed, prompting you to allow resetting of the program pointer and applying all changes:
   - Tap **Yes** to proceed.
   - Tap **No** to return to the data type view without resetting of the program pointer or applying changes.

Changing data type

This section describes how to change data type.

Action

1. In the Insert Expression view, tap **Change data type**, the following screen is displayed:



Manual
MySystem (SEVST-W-0003..)

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

Insert Expression - Change Data Type...

1 to 30 of 91

accdata	aiotrigg	bool
btnres	busstate	buttondata
byte	clock	cnvcmd
confdata	confsupdata	corrdescr
cssframe	datapos	dionum
dir	errdomain	errnum
errstr	errtype	event_type
exec_level	extjoint	gripdata
handler_type	icondata	identno
inposdata	intnum	inttypes

en0400000706

2. Tap to select the required data type and tap **OK**.

6.8.9. Hiding declarations in program code

Declarations

Program declarations can be hidden to make the program code easier to read.

Hiding declarations

This section describes how to hide or show declarations.

	Action
1.	In the **ABB** menu, tap **Program Editor** to view a program.
2.	Tap **Hide Declarations** to hide declarations. Tap **Show Declarations** to show declarations.

6.8.10. Deleting programs from memory

Overview

Deleting a program in a task does not erase the program from the controller mass memory but only from the program memory.

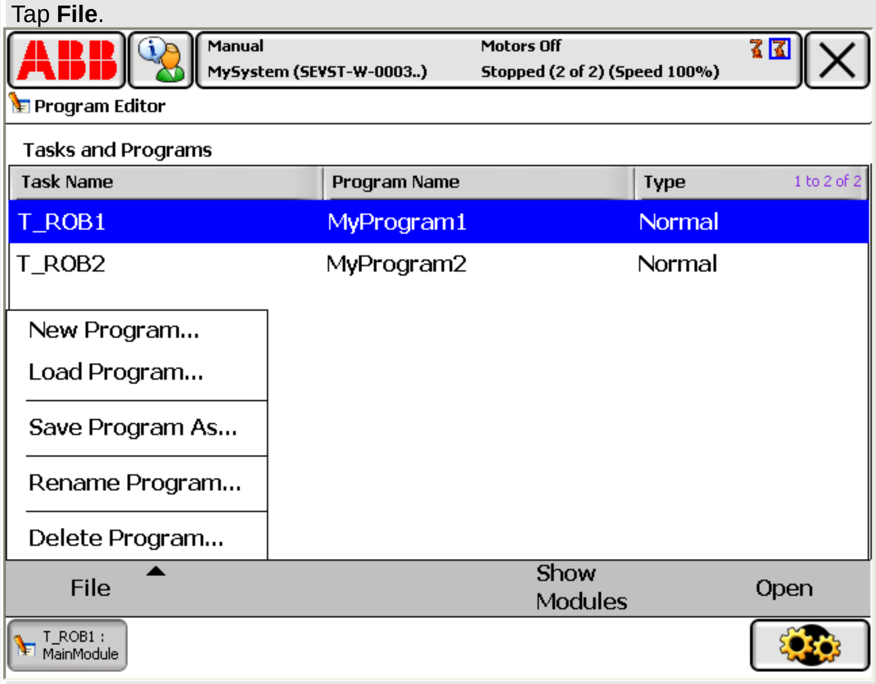
When you switch programs, the previously used program is deleted from the program memory, but not removed from the mass memory if it was saved there.

How to save your work is detailed in section *Handling of programs on page 166*.

The different memories are described in section *What is “the memory”? on page 260*.

Deleting programs from memory

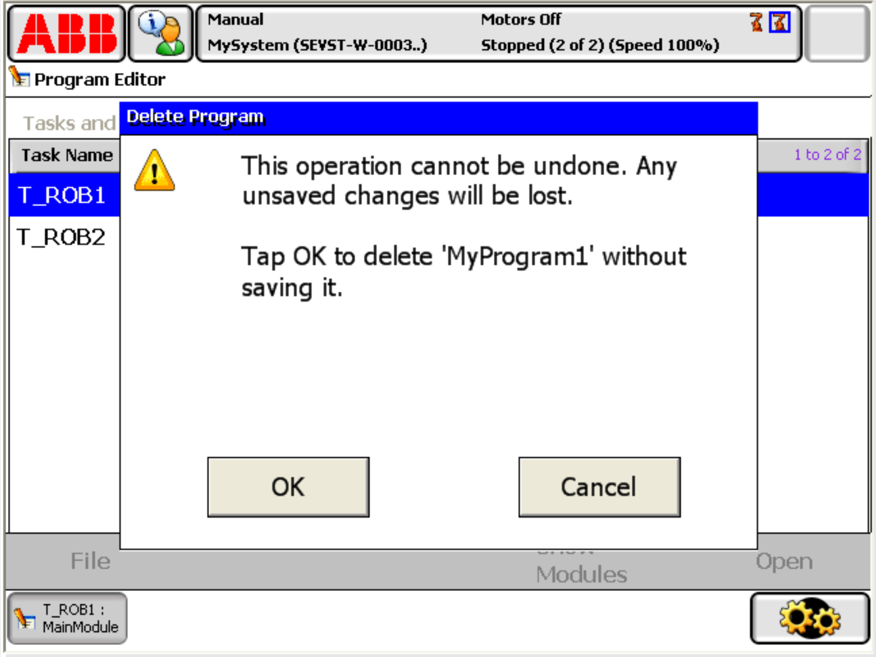
This section details how to delete programs from the program memory.

	Action
1.	On the ABB menu tap **Program Editor**.
2.	Tap **Tasks and Programs**.
3.	Tap **File**.  en0400000678

Tasks and Programs

Task Name	Program Name	Type
T_ROB1	MyProgram1	Normal
T_ROB2	MyProgram2	Normal

Continues on next page

	Action
4.	Tap **Delete Program...**. **WARNING!** Recent program changes will not be saved.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Delete Program This operation cannot be undone. Any unsaved changes will be lost. Tap OK to delete 'MyProgram1' without saving it. Program Editor Tasks and Task Name 1 to 2 of 2 T_ROB1 T_ROB2 OK Cancel File Show Modules Open T_ROB1 : MainModule en0400000853
5.	Tap **OK**. If you don't want to lose information about program changes then use **Save Program** before deleting the program. How to save your work is described in section *Handling of programs on page 166*.

6.8.11. Deleting programs from hard disk

Overview

Programs are deleted via FlexPendant Explorer or an FTP client. When deleting programs from the controller hard disk, the currently loaded program in the program memory is not affected.

The different memories are described in section *What is “the memory”? on page 260*.

Deleting programs with FlexPendant Explorer

Programs can be deleted using FlexPendant Explorer on the ABB menu. See section *FlexPendant Explorer on page 77*.

6.8.12. Activating mechanical units

Overview

A mechanical unit can be active or deactive. Only active units are run when executing a program. Deactivated units will not run. This may be useful when programming or testing a program.

A robot **cannot** be deactivated.

The Activate function does not affect jogging. To select mechanical unit for jogging, use the Mechanical unit property in the Jogging menu.

Activating mechanical units

This procedure describes how to activate a mechanical unit.

	Action	Info
1.	On the **ABB** menu, tap **Jogging**.	
2.	Make sure that the right mechanical unit is selected, then tap **Activate...**. To deactivate an active mechanical unit, tap **Deactivate**.	A robot cannot be deactivated.

Related information

Selecting mechanical unit for jogging on page 108.

Mechanical units can be active or deactive at startup depending on the system setup, see *Technical reference manual - System parameters*, topic *Motion*.

6.9 Testing

6.9.1. Using the hold-to-run function

When to use the hold-to-run function

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

In order to run a program in manual full speed mode it is necessary, for safety reasons, to keep pressing both the enabling device and the Start button. This hold-to-run function also applies when stepping through a program in manual full speed mode. When Start, Forward, and Backward buttons are used like this (press and hold) they are referred to as hold-to-run buttons. Some versions of the FlexPendant may also have separate hold-to-run buttons.

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

Using the hold-to-run function

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

	Action
1.	Press the enabling device on the FlexPendant.
2.	Choose execution mode by pressing and holding either: • **Start** (continuous program execution) • **Forward** (step-by step program execution forwards) • **Backward** (step-by step program execution backwards)
3.	If **Start** was pressed, then the program execution continues as long as the Start button is pressed. If **Forward** or **Backward** was pressed, the program is executed step-by-step by alternately releasing and pressing the Forward/Backward button. Note that the button must be pressed and held until the instruction is executed. If the button is released, program execution will stop immediately!
4.	If the hold-to-run button is released, program execution stops. If the hold-to-run button is pressed again after being released, program execution is resumed from the position in which it was released.
5.	It is possible to change execution mode when the hold-to-run button is released and then continue the program execution with the new execution mode, by just activating the hold-to-run button again.
6.	If the enabling device is released, intentionally or by accident, the complete procedure must be repeated to enable running.

6.9.2. Running the program from a specific instruction

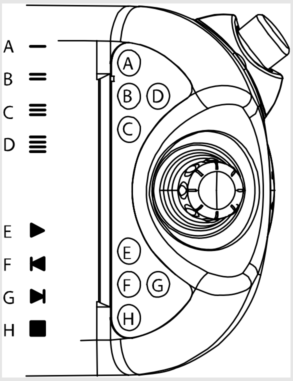
Overview

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

WARNING!

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

Running the program from a specific instruction

	Action
1.	On the **ABB** menu Tap **Program Editor.**
2.	Tap on the program step where you want to start, then tap **Debug** and then **PP to Cursor**.
3.	**DANGER!** **Make sure that no personnel are in the robot working area.** Before running the robot, observe the safety information in section *DANGER - Moving robots are potentially lethal! on page 18*.
4.	Press the **Start** button on the FlexPendant (see E in illustration below).  en0300000587

6.9.3. Running a specific routine

Overview

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

Prerequisites

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

Running a specific routine

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

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

Related information

How to run a service routine is described in *Running a service routine on page 215*. The same method can be used to run a specific routine in the task scope. See *Running a service routine on page 215* for detailed information.

6.9.4. Quickset menu, Run Mode

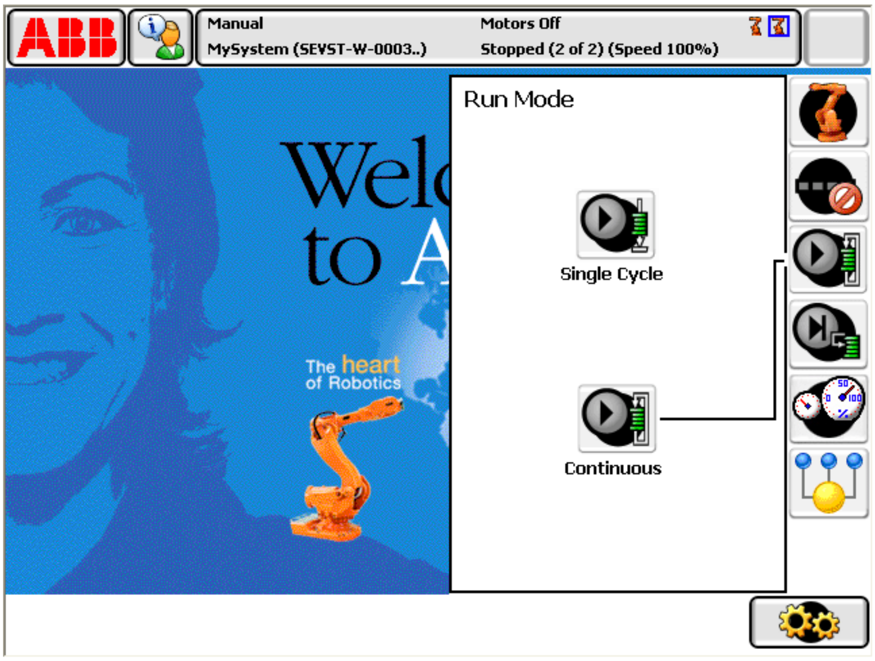
Run mode

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

For information about run mode in:

- *Multitasking*, see *Application manual - Engineering tools*, section *Multitasking*.
- *MultiMove*, see *Application manual - MultiMove*, section *User interface specific for Multimove*.

Illustration Run mode



en0300000472

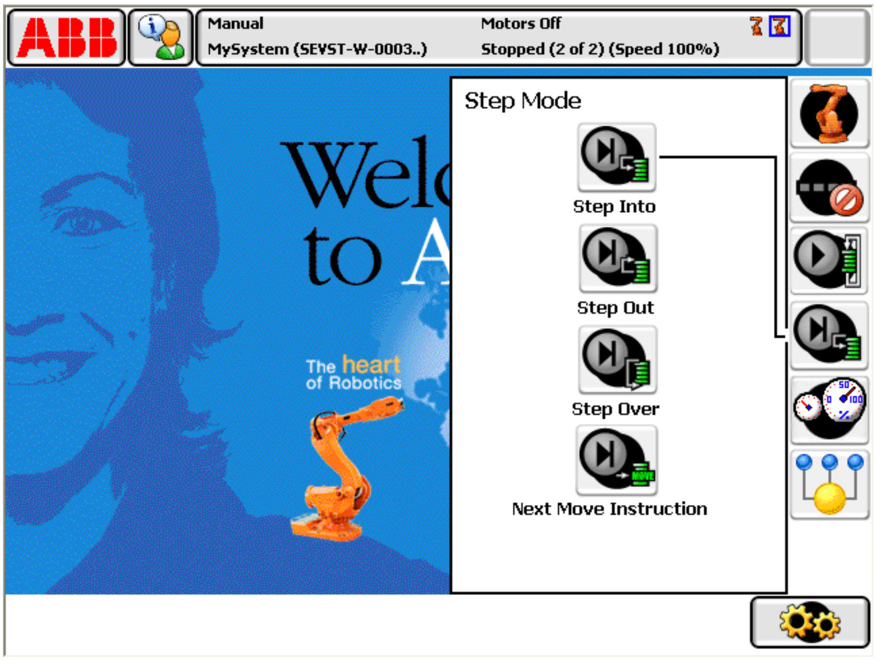
Single Cycle	Runs one cycle then stops execution.
Continuous	Runs continuously.

6.9.5. Quickset menu, Step Mode

Step mode

By setting step mode you define how the step-by-step program execution should function.

Illustration Step mode



en0300000543

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

6.9.6. Stepping instruction by instruction

Overview

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

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

Select step mode

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

	Action	Info
1.	Select step mode using the Quickset menu.	Described in *Quickset menu, Step Mode on page 209*.

Stepping

This section details how to step forwards and backwards.

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

Limitations of backward execution

There are some restrictions for the backward execution:

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

Backward execution behavior

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

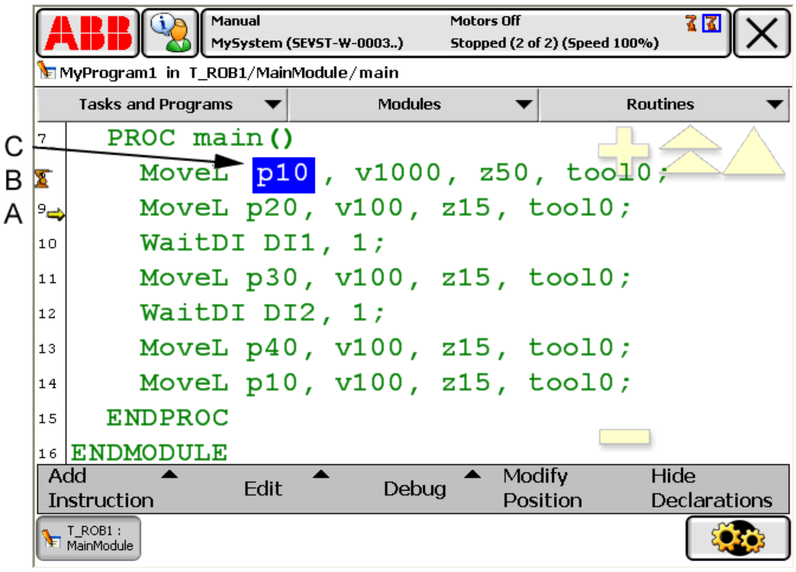
When stepping backward though the program code, the program pointer indicates the instruction above the motion pointer. When the program pointer indicates one move instruction and the motion pointer indicates another, the next backward movement will move to the target indicated by the program pointer,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

Continues on next page

Example of backward execution

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

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



```
  PROC main()
    MoveL p10, v1000, z50, tool0;
    MoveL p20, v100, z15, tool0;
    WaitDI DI1, 1;
    MoveL p30, v100, z15, tool0;
    WaitDI DI2, 1;
    MoveL p40, v100, z15, tool0;
    MoveL p10, v100, z15, tool0;
  ENDPROC
ENDMODULE
```

en0400001204

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

When...	then...
stepping forward until the robot is in p5	the motion pointer will indicate p5 and the program pointer will indicate the next move instruction (`MoveL p6`).
pressing the Backward button once	the robot will not move but the program pointer will move to the previous instruction (`MoveC p3, p4`). This indicates that this is the instruction that will be executed the next time Backward is pressed.
pressing the Backward button again	the robot will move to p4 linearly with the speed v300. The target for this movement (p4) is taken from the `MoveC` instruction. The type of movement (linear) and the speed are taken from the instruction below (`MoveL p5`). The motion pointer will indicate p4 and the program pointer will move up to `MoveL p2`.
pressing the Backward button again	the robot will move circularly, via p3, to p2 with the speed v100. The target p2 is taken from the instruction `MoveL p2`. The type of movement (circular), the circular point (p3) and the speed are taken from the `MoveC` instruction. The motion pointer will indicate p2 and the program pointer will move up to `MoveL p1`.

Continues on next page

Continued

When...	then...
pressing the Backward button again	the robot will move linearly to p1 with the speed v200. The motion pointer will indicate p1 and the program pointer will move up to `MoveJ p0`.
pressing the Forward button once	the robot will not move but the program pointer will move to the next instruction (`MoveL p2`).
pressing the Forward button again	the robot will move to p2 with the speed v200.

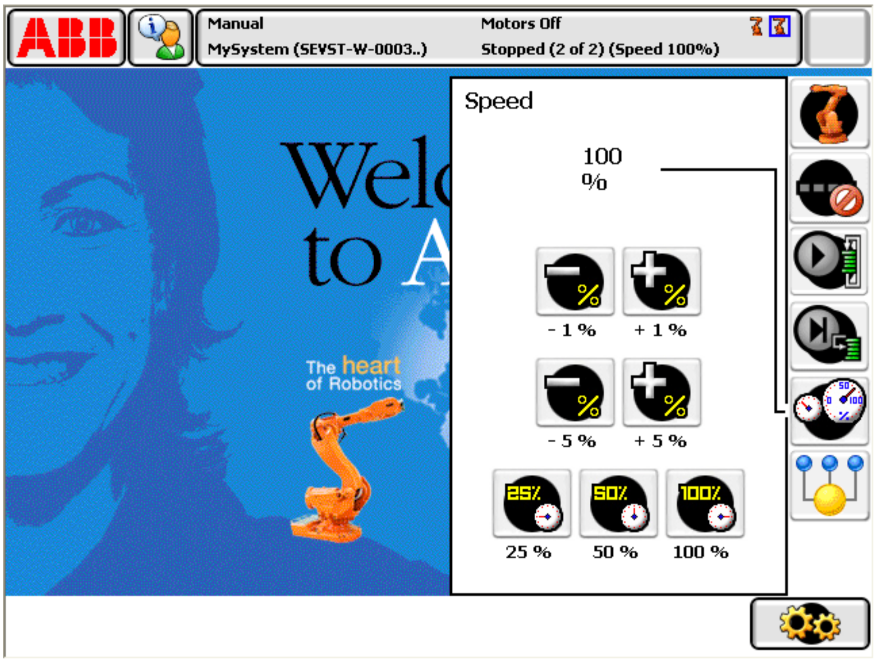
6.9.7. Quickset menu, Speed

Speed button

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

Illustration Speed

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


Manual
MySystem (SEVST-W-0003..)
Motors Off
Stopped (2 of 2) (Speed 100%)

en0300000470

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

6.9.8. Quickset menu, Tasks

Tasks button

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

By default, only normal tasks are possible to activate/deactivate in the Quickset menu. Using the Control Panel you can however change the settings so all tasks are possible to activate/ desactivate.

For static and semistatic tasks, only those with the system parameter *TrustLevel* set to `NoSafety` can be activated/deactivated. Activated tasks are started and stopped with the Start and Stop buttons on the FlexPendant.

The tasks settings are only valid for manual operating mode.

Related information

Application manual - Engineering tools, section *Multitasking*.

How to start and stop multitasking programs is described in section *Using multitasking programs on page 231*.

TrustLevel for tasks are set with system parameters, see *Configuring system parameters on page 283*, and section *Task* in *Technical reference manual - System parameters*.

You can define if all tasks or only normal tasks should be displayed. See section *Defining which tasks should be selectable in the tasks panel on page 307*.

6.10 Service routines

6.10.1. Running a service routine

Service routines

Service routines perform a number of common services. Which service routines are available depends on your system setup and available options. Please refer to your plant or cell documentation for more information.

Prerequisites

Service routines can only be started in manual mode. The program must be stopped and there has to be a program pointer.

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

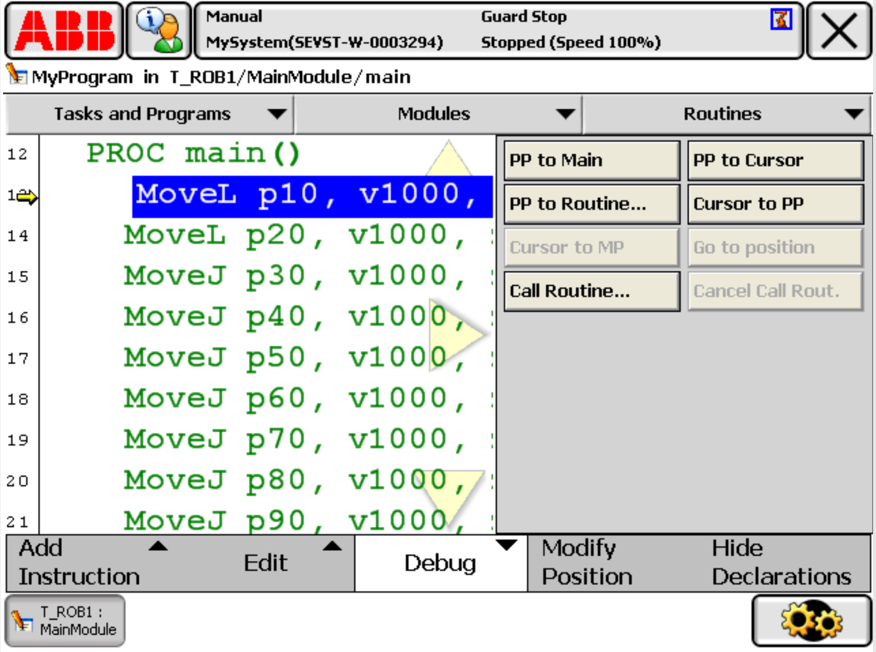
If the service routine contains parts that must be carried out in automatic mode, then the program pointer must not be moved manually before starting the service routine. The program pointer should be where the program flow was stopped.

CAUTION!

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

Running a service routine

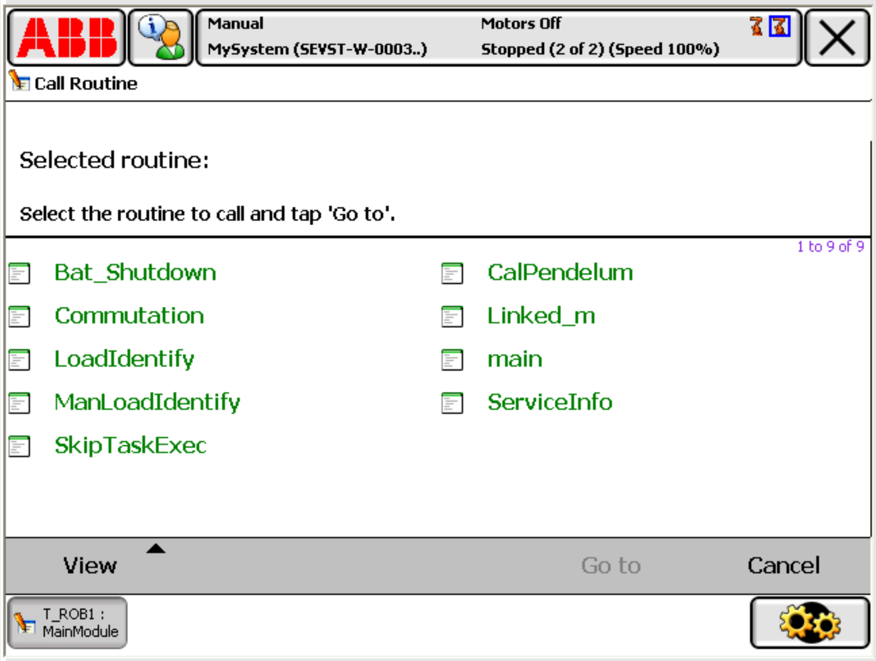
This section describes how to execute a service routine or another routine in the task scope using **Call Routine**.

	Action
1.	On the **ABB** menu tap **Program Editor**.
2.	On the **Debug** menu tap **Call Routine**. 

```
PROC main()
    MoveL p10, v1000,
    MoveL p20, v1000,
    MoveJ p30, v1000,
    MoveJ p40, v1000,
    MoveJ p50, v1000,
    MoveJ p60, v1000,
    MoveJ p70, v1000,
    MoveJ p80, v1000,
    MoveJ p90, v1000,
```

en0400000868

Continues on next page

	Action
3.	The **Call Service Routine** dialog lists all predefined service routines. The same dialog can however be used to run any routine in the task scope. Select **All Routines** on the **View** menu to see all available routines.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Call Routine Selected routine: Select the routine to call and tap 'Go to'. 1 to 9 of 9 Bat_Shutdown CalPendelum Commutation Linked_m LoadIdentify main ManLoadIdentify ServiceInfo SkipTaskExec View Go to Cancel T_ROB1 : MainModule en0400000885
4.	Tap a service routine and then tap **Go to**. The **Program Editor** will be displayed with the program pointer moved to the beginning of the selected routine.
5.	Press the **Start** button on the FlexPendant and follow the instructions displayed on the FlexPendant. After execution of the routine the task is stopped and the program pointer is returned to where it was before the service routine started.

CAUTION!

Press **Cancel Call Rout** if you need to interrupt the routine before it has finished executing. Before resuming normal program flow, however, you must see to it that the robot is correctly positioned. If the interrupted routine has moved it, you will need to take actions to return the robot to its position. See *Returning the robot to the path on page 238* for further information.

Limitations

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

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

Continues on next page

Related information

Battery shutdown service routine on page 218.

LoadIdentify, load identification service routine on page 221.

Service Information System, ServiceInfo service routine on page 220.

Calibration Pendulum, CalPendulum service routine on page 219.

6.10.2. Battery shutdown service routine

Bat_shutdown

It is possible to shutdown the battery backup of the Serial Measurement Board to save battery power during transportation or storage. The function is reset when the system is powered on again. The revolution counters will be lost and needs an update but the calibration values will remain.

Related information

How to start a service routine is described in *Running a service routine on page 215*.

The Serial Measurement Board is described in *Serial Measurement Board memory on page 294*.

How to update the revolution counters is described in *Updating revolution counters on page 288*.

6.10.3. Calibration Pendulum, CalPendulum service routine

CalPendulum

CalPendulum is a service routine used with *Calibration Pendulum*, the standard method for calibration of ABB robots. This is the most accurate method for the standard type of calibration, and it is also the recommended method in order to achieve proper performance.

The calibration equipment for *Calibration Pendulum* is delivered as a complete toolkit, including the manual *Operating manual - Calibration Pendulum*.

Related information

Running a service routine on page 215.

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

6.10.4. Service Information System, ServiceInfo service routine

ServiceInfo

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

Maintenance is scheduled by setting the system parameters of the type *SIS Parameters*. How to work with system parameters is described in section *Configuring system parameters on page 283*. All system parameters are described in *Technical reference manual - System parameters*. More details about SIS is described in *Operating manual - Service Information System*.

Supervised functions

The following counters are available:

- Calender time counter
- Operation time counter
- Gearbox operation time counters

Counters are reset when maintenance has been performed.

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

Related information

Running a service routine on page 215.

Configuring system parameters on page 283.

Operating manual - Service Information System.

The system parameters for SIS are described in *Technical reference manual - System parameters*, chapter *Motion*.

6.10.5. LoadIdentify, load identification service routine

Overview

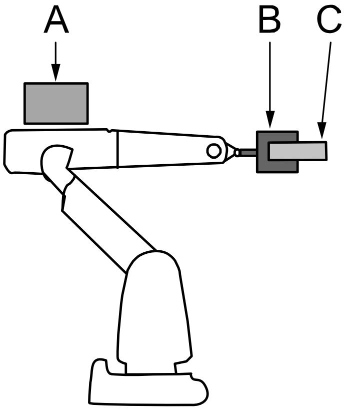
The service routine LoadIdentify is used to automatically calculate the data of loads mounted on the robot. You can also enter the data manually, but this requires information that may be difficult to calculate.

If you want to run load identification for the payload, make sure that the tool is correctly defined first, e.g. by running LoadIdentify for the tool.

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

LoadIdentify

LoadIdentify can identify the tool load and the payload. The data that can be identified are mass, center of gravity, and moments of inertia. Together with the identified data a measurement accuracy, indicating how well the identification went, is also given.



en0500001535

A	Upper arm load
B	Tool load
C	Payload

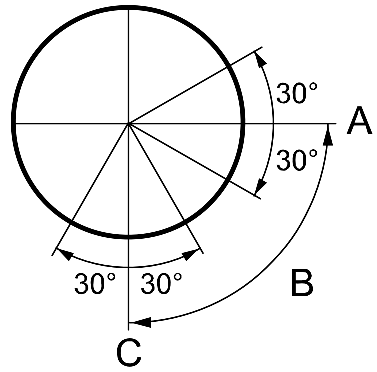
The movements of axis 3 will only be performed if the mass is to be identified. This means that to identify the mass, the upper arm load must be known and correctly defined first.

Continues on next page

Calibration angles

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

The optimum value for the configuration angle is 90 degrees.

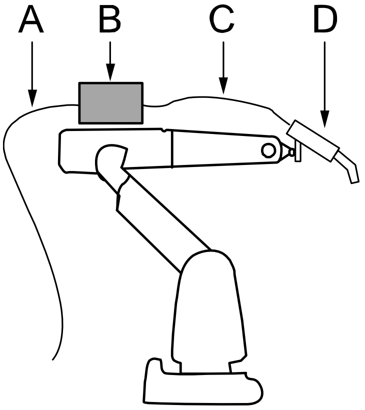


en0500001537

A	Configuration 2
B	Configuration angle
C	Configuration 1 (start position)

LoadIdentify with arm loads mounted

The best way to perform load identification is with a robot with no arm loads mounted. If this is not possible, good accuracy can still be achieved. Consider, for example, the robot in the figure below, which has arc welding equipment mounted on it.



en0500001536

A	Cable 1
B	Load 1
C	Cable 2
D	Load 2

If we want to use load identification to find the data of load 2, the most important thing to remember is to make sure that the upper arm load is correctly defined, in particular its mass and its center of gravity along the robot arm. The arm load includes everything that is mounted on the robot, except tool load and payload. In the figure above, cable 1, cable 2, and load 1 are included in the arm load.

Continues on next page

When performing the load identification, cable 2 should be disconnected since it will otherwise put an extra force on load 2. When identifying load 2 with such a force present, the result may differ considerably from the correct load. Ideally, cable 2 should be disconnected from load 2 and fastened on the upper arm. If this is not possible, the cable can also be disconnected at load 1 and fastened to the upper arm in such a way that the resulting force on load 2 is minimized.

Prerequisites for tool loads

Before running the LoadIdentify service routine for a tool load, make sure that:

- the tool is selected in the jogging menu
- the tool is correctly mounted
- axis 6 is close to horizontal
- the upper arm load is known, if the mass is to be identified
- the axes 3, 5, and 6 are not close to their corresponding working range limits
- the speed is set to 100%
- the system is in manual mode.

Note that LoadIdentify cannot be used for tool0.

Prerequisites for payloads

Before running the LoadIdentify service routine for a payload, make sure that:

- the tool and payload are correctly mounted
- axis 6 is close to horizontal
- the tool load is known (run LoadIdentify for the tool first)
- the upper arm load is known, if the mass is to be identified
- when using a moving TCP, the tool must be calibrated (TCP)
- when using a stationary TCP, the corresponding work object must be calibrated (user frame and object frame)
- the axes 3, 5, and 6 are not close to their corresponding working range limits
- the speed is set to 100%
- the system is in manual mode.

Running LoadIdentify

To start the load identification service routine you must have an active program in manual mode and the tool and payload that you want to identify must be defined and active in the jogging window.

	Action	Info
1.	Start LoadIdentify from the program editor. Press the enabling device and then the Start button on the FlexPendant.	How to start service routines is described in *Running a service routine on page 215*.
2.	Tap **OK** to confirm that current path will be cleared and that the program pointer will be lost.	Tap **Cancel** and then **Cancel Call Rout** to quit the service routine without loosing the program pointer.

Continues on next page

Continued

	Action	Info
3.	Tap **Tool** or **Payload**.	
4.	Tap **OK** to confirm that the correct tool and/or payload is active in the jogging menu and that the tool load/payload is correctly mounted.	If it is not correct, release the enabling device and select the correct tool/payload in the jogging menu. Then return to LoadIdentify, press the enabling device, and press Start. Tap **Retry** and confirm that the new tool/ payload is correct.
5.	When identifying tool loads, confirm that the tool is active. When identifying payloads, confirm that the payload's tool is active and calibrated.	See above.
6.	When identifying payloads with stationary TCP, confirm that the correct work object is active and (preferably) calibrated. If it is correct, tap **OK** to confirm.	See above.
7.	Select identification method. If you select the method where the mass is assumed to be known, remember that the tool/payload that you use must have the correct mass defined. Tap **OK** to confirm.	
8.	Select configuration angle. The optimum is +90 or -90 degrees. If this is impossible, tap **Other** and set the angle. The minimum is plus or minus 30 degrees.	
9.	If the robot is not in a correct position for load identification, you will be asked to jog one or more axes roughly to a specified position. When you have done this tap **OK** to confirm. If the robot is still not in a correct position for load identification, the robot will slowly move to the correct position. Press **Move** to start the movement.	Axis 1 to 3 must not be more than 10 degrees from proposed position.
10.	The robot can go through the load identification movements slowly before performing the load identification. Tap **Yes** if you want a slow test and **No** to proceed to the identification.	This is useful for ensuring that the robot will not hit anything during the identification. However, this will take a lot longer time.
11.	The setup for load identification is now complete. To start the motion, switch to Automatic mode and Motors On. Then tap **Move** to start the load identification movements.	
12.	When the identification is finished, switch back to manual mode, press the enabling device and the Start button. Tap **OK** to confirm.	
13.	The result of the load identification is now presented on the FlexPendant. Tap **Yes** if you want to update the selected tool or payload with the identified parameters or **No** otherwise.	

Continues on next page

Error handling

If the enabling device is released during the load identification (before the movements start), the routine can always be restarted by pressing the enabling device again and then pressing the Start button.

If an error should occur during the load identification movements, the routine must be restarted from the beginning. This is done automatically by pressing Start after confirming the error. To interrupt and leave the load identification procedure, tap **Cancel Call Routine** in the program editor’s debug menu.

Limitations for LoadIdentify

Only tool loads and payloads can be identified with LoadIdentify. Thus arm loads cannot be identified using this procedure.

If the load identification movements are interrupted by any kind of stop (program stop, emergency stop, etc.), the load identification must be restarted from the beginning. This is done automatically if you press Start after confirming the error.

If the robot is stopped on a path with program stop and load identification is performed at the stop point, the path will be cleared. This means that no regain movement will be performed to return the robot back to the path.

The load identification ends with an EXIT instruction. That means that the program pointer is lost and must be set to main before starting any program execution.

If the measurement accuracy is lower than 80%, the result of the load identification may have significant errors. In this case, a higher accuracy may be achieved by repeating the LoadIdentify routine. If repeating the routine does not give a higher accuracy, then the torques measured in the identification are probably too small and the tool and/or payload data must be set manually. This is typically the case if the mass of the load is small (10% or less of the maximum load). It can also happen if the load has a particular symmetry property, for instance if the tool load is symmetrical around axis 6. However, even if the measurement accuracy is low some of the identified data may still be correct.

Related information

It is also possible to include LoadIdentify in a program by using RAPID instructions. See *Technical reference manual - RAPID Instructions, Functions and Data types*.

How to enter the data manually is described in *Editing the tool data on page 149*, and *Editing the payload data on page 163*.

The product manual for the robot contain information on how and where to mount the loads.

Load identification for positioners is done with the service routine ManLoadIdentify. This is described in the manual *System settings* for the positioner.

7 Running in production

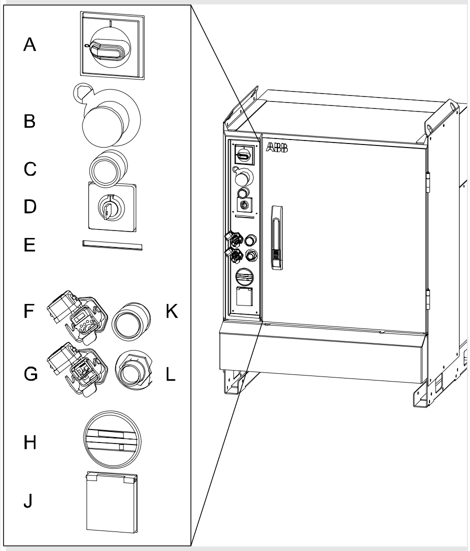
7.1 Basic procedures

7.1.1. Starting programs

Starting programs

Use this procedure to start a program for the first time or to continue running a program that has been stopped.

If your robot system has the option *Multitasking* installed, also see *Using multitasking programs on page 231*.

	Action	Info
1.	Check that all necessary preparations are done to the robot and in the robot cell and that no obstacles exist within the robot work area.	
2.	Make sure no personnel are inside the robot cell.	
3.	Select operating mode on the controller with the mode switch.	 xx0600002782 C: Motors on button D: Mode switch
4.	Press the Motors on button on the controller to activate the robot.	
5.	Is a program loaded? If yes, proceed to the next step. If no, load a program.	How to load programs is described in section *Handling of programs on page 166*.

Continues on next page

	Action	Info
6.	In Auto mode: 1. Press the Start button on the FlexPendant to start the program. In manual mode: 1. Select start mode. 2. Press and hold the enabling device. 3. Press the Start button on the FlexPendant to start the program.	The button is shown in section *Hard buttons on page 41*. How to select start mode is detailed insection *Using the hold-to-run function on page 205*.
7.	Is the **Return to path** dialog box displayed? If yes, return the robot to the path using a suitable method. If no, proceed.	Returning the robot to the path is described in section *Returning the robot to the path on page 238*.
8.	If the **Cursor does not coincide with PP** dialog box is displayed then tap **PP** or **Cursor** to select from where the program should start. Then press the Start button again.	This dialog box is only displayed if the system parameters of type *Warning at start* are defined. See *Technical reference manual - System parameters*.

Continue running after the program is changed

You can always continue a program even if it has been changed.

In automatic mode, a warning dialog may appear to avoid restarting the program if the consequences are unknown.

If you...	then tap...
Are sure the changes you have made are not in conflict with the current robot position and that the program can continue without danger to equipment or personnel	Yes
Are unsure of the consequences your changes might have and want to investigate further	No

Restart from the beginning

A program can be restarted from the Production Window or the Program Editor.

PP to Main from the Production Window will reset the program pointer to the production entry in all normal tasks, including tasks deactivated in the task selection panel.

PP to Main from the Program Editor will reset the program pointer to the production entry in the specified task only, even if the task is deactivated in the task selection panel.

Use this procedure to restart a program from the Production Window.

	Action
1.	On the **ABB** menu, tap **Production Window**.
2.	Tap **PP to Main**.
2.	Start the program by pressing the Start button on the FlexPendant.

Continues on next page

Continued

Use this procedure to restart a program from the Program Editor.

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Debug**.
3.	Tap **PP to Main**.
4.	Start the program by pressing the Start button on the FlexPendant.

Limitations

Only one program at a time can be executed, unless your system has the *Multitasking* option. If so several programs can be executed simultaneously.

If the robot system encounters program code errors while the program is running, it will stop the program and the error is logged in the event log.

7.1.2. Stopping programs

Stopping programs

If your robot system has the *Multitasking* option installed, see *Using multitasking programs on page 231*.

	Action
1.	Check that the ongoing operation is in such a state that it can be interrupted.
2.	Make sure it is safe to stop the program.
3.	Press the **Stop** button on the FlexPendant hardware button set. The button is shown in section *Hard buttons on page 41*.

DANGER!

Do not use the Stop button in an emergency. Use the emergency stop button.

Stopping a program with the stop button does not mean that the robot will stop moving immediately.

Stopping execution when using hold-to-run or step-by-step execution

When using hold-to-run or step-by-step execution, execution can be stopped according to the following.

Mode	Action	Info
Operation *with* hold-to-run	Release the Start button	The hold-to-run function is described in section *What is a FlexPendant? on page 40*.
Step-by-step mode	The robot will stop after executing each instruction. Execute the next instruction by pressing the Forward button again.	The STOP and Forward button are described in section *What is a FlexPendant? on page 40*. If you press the STOP button while executing a move instruction, the robot will stop without completing the move.

7.1.3. Using multitasking programs

Overview

In a system with the option *Multitasking* installed, you may have one or several programs running in parallel, for instance in a *MultiMove* cell with more than one robot where each robot has its own task and program (multitasking).

For general information on program handling, see *Handling of programs on page 166*. Multitasking is described in *Application manual - Engineering tools*.

TIP!

Need to know more about tasks and programs? These concepts are described in *The structure of a RAPID application on page 134*.

Manually set up tasks

Tasks need to be set up in order to run as planned. Normally, all tasks are set up on delivery. Setting up tasks is done by defining system parameters of the type *Controller*. See section *Configuring system parameters on page 283* on how to configure system parameters, or *Technical reference manual - System parameters* for information about the parameters.

You need detailed information to set up tasks manually. Please read your plant or cell documentation for details.

How tasks are run

Tasks may be defined as Normal, Static, or Semistatic. Static and Semistatic tasks are automatically started as soon as a program is loaded into that task.

Normal tasks are started when you press the Start button of the FlexPendant, and stopped when you press the Stop button.

To be able to step, start and stop a Static or Semistatic task: set *TrustLevel* to NoSafety, set Task Panel Settings to All tasks and activate the task using the Quickset menu. See *Application manual - Engineering tools*, section *Multitasking*.

The concepts of Static, Semistatic, and Normal are described in *Technical reference manual - System parameters*, type *Tasks*.

Load, run, and stop multitasking programs

This section describes how to load, run, and stop multitasking programs.

	Action
1.	Make sure there is more than one task set up. This is done using system parameters, see *Technical reference manual - System parameters*.
2.	Load programs to respective task using the Program Editor or the Production Window, this is described in section *Loading an existing program on page 167*.
3.	If one or more task should be disabled, go to the Quickset menu to do this. See section *Quickset menu, Tasks on page 214*. Deselecting tasks can only be done in manual mode. When switching to automatic mode, an alert box will appear warning that not all tasks are selected to run.
4.	Start program execution by pressing the start button. All active tasks are started.
5.	Stop program execution by pressing the stop button. All active tasks are stopped.

Continues on next page

How to load a program to a task

This section describes how to load a program to a task in a multitasking system. It is assumed that the tasks have been configured.

Load a program from the Production Window

	Action
1.	On the **ABB** menu, tap **Production Window**.
2.	Tap the task into which you want to load a program.
3.	Tap **Load Program...**. If you want to open a program in another folder, locate and open that folder. See description in *FlexPendant Explorer on page 77*. The file dialog box appears.
4.	Tap the program you want to load followed by **OK**.

Load a program from the Program Editor

	Action
1.	On the **ABB** menu, tap **Program Editor**.
2.	Tap **Tasks and Programs**.
3.	Tap the task into which you want to load a program.
4.	On the File menu, tap **Load Program...**. If you want to open a program in another folder, locate and open that folder. See description in *FlexPendant Explorer on page 77*. The file dialog box appears.
5.	Tap the program you want to load followed by **OK**.
6.	Tap **Close** to close the Program Editor.

Viewing multitasking programs

In the Production Window, there is one tab for each task. To switch between viewing the different tasks, tap on the tabs.

To edit several tasks in parallel, open one Program Editor for each task. To edit static and semistatic tasks, see *Application manual - Engineering tools*, section *Multitasking*.

7.1.4. Using motion supervision and non motion execution

Motion supervision

The controller software has functionality aiming at reducing collision impact forces on the robot. This helps protecting the robot and external equipment from severe damage if a collision occurs.

Motion supervision during program execution is by default always active, regardless which options are installed in the controller. When a collision is detected, the robot will immediately stop and relieve the residual forces by moving in reversed direction a short distance along its path. The program execution will stop with an error message. The robot remains in the state Motors on so that program execution can be resumed after the collision error message has been acknowledged.

Moreover, there is a software option called *Collision Detection*, which has extra features such as supervision during jogging. To find out if your system has this option installed, tap **System Info** on the ABB menu. Expand the node *System Properties* and tap *Options* under *Control Module*.

Functions in RobotWare base

Description of functions in RobotWare base:

- *Path Supervision* in automatic and manual full speed mode used to prevent mechanical damage due to the robot running into an obstacle during program execution.
- *Non motion execution* used to run a program without robot motion.

Functions in Collision Detection

A RobotWare system with *Collision Detection* has additional functionality:

- *Path Supervision* in manual mode and the possibility to tune supervision in all modes.
- *Jog Supervision* used to prevent mechanical damage to the robot during jogging.
- RAPID instruction `MotionSup` used to activate/deactivate collision detection and to tune sensitivity during program execution.

NOTE!

All motion supervision must be set for each task separately.

Editing motion supervision settings

This section describes how to modify settings for motion supervision.

	Action	Info
1.	On the **ABB** menu tap **Control Panel** and then **Supervision**.	
2.	Tap the **Task** list and select a task.	If you have more than one task, you need to set the desired values for each task separately.

Continues on next page

Continued

	Action	Info
3.	Tap **OFF/ON** to remove or activate path supervision. Tap -/+ to adjust sensitivity. **Note**: Unless you have the option *Collision Detection* installed, path supervision only affects the robot in auto and manual full speed mode. **Note**: The sensitivity setting has no effect, unless you have *Collision Detection* installed.	**TIP!** Sensitivity can be set between 0 and 300. If it is set lower than 80, however, the robot may stop due to internal drag.
4.	Tap **OFF/ON** to remove or activate jog supervision. **Note**: This setting has no effect, unless you have *Collision Detection* installed. Tap the -/+ to adjust sensitivity. **Note**: This setting has no effect, unless you have the option *Collision Detection* installed.	**TIP!** Sensitivity can be set between 0 and 300. If it is set lower than 80, however, the robot may stop due to internal drag.
5.	Under **Execution Settings**, tap **OFF/ON** to deactivate or activate non motion execution. This is a separate function, not a part of motion supervision.	See *Non motion execution on page 234* for information about this function.

Non motion execution

Non motion execution enables you to run a RAPID program without robot motion. All other functions work normally; current cycle times, I/O, TCP speed calculation etcetera.

Non motion execution can be used for program debugging or cycle time evaluation. It also represents a solution if you need to measure for example glue or paint consumption during a cycle.

When non motion execution is activated it can be executed in:

- manual mode
- manual full speed mode
- auto mode

Cycle times will be simulated according to the selected mode.

NOTE!

Non motion execution can only be activated when the system is in Motors Off state.

CAUTION!

Non motion execution is reset after a reboot. If you intend to run the program in non motion mode, do not restart without checking the status of **Non motion execution**. Starting the program incorrectly may cause serious injury or death, or damage the robot or other equipment.

Related information

For more information on *Collision Detection*, see *Application manual - Motion coordination and supervision*.

7.1.5. Using the hot plug option

Hot plug option

The hot plug option makes it possible to:

- Disconnect the FlexPendant from a system in automatic mode and thereby run the system without a FlexPendant connected.
- Temporarily connect and operate a FlexPendant without interrupting the application running on the system.

WARNING!

Pressing the hot plug button disables the emergency stop button on the FlexPendant. Only press the hot plug button while connecting or disconnecting the FlexPendant.

WARNING!

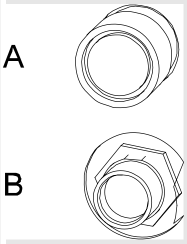
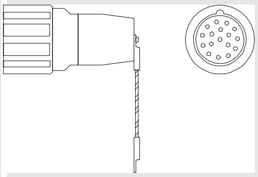
A disconnected FlexPendant must always be stored separated from the IRC5 controller!

Connect and disconnect the FlexPendant using the hot plug button

The following procedure describes how to connect or disconnect the FlexPendant on a system in automatic mode using the hot plug button option.

NOTE!

Do not switch to manual mode (or manual full speed mode) while the system is running without the FlexPendant. The FlexPendant must be connected when you switch to automatic mode otherwise you cannot confirm the mode change.

	Action	Info
1.	Make sure that the system is in automatic mode.	
2.	Press and hold the hot plug button.	A red lamp inside the button indicates when pressed.
3.	Keep pressing the hot plug button and at the same time, switch the jumper plug with the FlexPendant plug.	 xx0600002784 A: Hot plug button B: FlexPendant connector  xx0600002796 Jumper plug

Continues on next page

	Action	Info
4.	Release the hot plug button.	Make sure that the button is not stuck in the actuated position since this disables the FlexPendant emergency stop button.

NOTE!

When the FlexPendant is disconnected, the jumper plug must be connected in its place.

NOTE!

If the hot plug button is released while neither the jumper plug, nor the FlexPendant is connected, the robot movements will be stopped since the emergency stop chains are opened.

Limitations for messages on the FlexPendant

When using the hot plug option, the following limitations apply to messages on the FlexPendant:

Operator messages

Some applications may require input from the operator by using the FlexPendant (e.g. applications using RAPID instructions `TPReadNum`, `UIMsgBox`, etc.). If the application encounters such an operator message, program execution will wait. After connecting the FlexPendant you must then stop and start the program execution to be able to see and respond to these messages. They are not displayed automatically by just connecting the FlexPendant.

If possible, avoid using these types of instructions when programming systems that are using the hot plug button option.

Event log messages

When connecting the FlexPendant, event log messages can be viewed also for the period when the FlexPendant was disconnected, since these are stored on the controller.

7.2 Trouble shooting and error recovery

7.2.1. General procedure when trouble shooting

Types of faults

Faults occurring in the robot system may be of two categories:

- Faults detected by the built-in diagnostics system. These faults are described in section *Event log messages* in *Operating manual - Trouble shooting*.
- Faults NOT detected by the built-in diagnostics system. These faults are described in section *Other types of faults* in *Operating manual - Trouble shooting*.

Faults causing error message on the FlexPendant

The control system is supplied with diagnostic software to facilitate trouble shooting and to reduce downtime. Any errors detected by the diagnostics are displayed in plain language with a code number on the FlexPendant.

All system and error messages are logged in a common log in which the last 150 messages are saved. The log can be accessed from the Status bar on the FlexPendant.

To facilitate trouble shooting, it is important that some basic principles are followed. These are specified in *Trouble shooting principles* in *Operating manual - Trouble shooting*.

	Action	Info
1.	Read the error message displayed on the FlexPendant and follow any instructions given.	How to interpret the messages is detailed in *Event Log on page 90*, and chapter *Handling the event log on page 255*.
2.	Was the information given on the FlexPendant enough to solve the problem? If yes; resume operation. If no; proceed below.	
3.	If relevant, check the LEDs on the units.	Each unit is thoroughly described in section *Unit LEDs* in *Operating manual - Trouble shooting*, including a description of its LEDs.
4.	If relevant, check the cables, etc., with help of the circuit diagram.	Circuit diagrams are found in the *Product manual* for the robot or controller.
5.	Replace, adjust or fix as detailed in the Repairs instruction if required.	See the *Product manual* for the robot or controller.

Faults NOT causing error messages on the FlexPendant

These faults are not detected by the diagnostic system and are handled in other ways. The way the symptom of the fault is observed greatly influences the type of fault. Instructions are given in section *Other types of faults* in *Operating manual - Trouble shooting*.

To trouble shoot faults NOT causing error messages on the FlexPendant, follow steps 3 and 4 in the procedure above.

Other possible actions

Some errors may require running a service routine. See section *Service routines on page 215*.

7.2.2. Returning the robot to the path

About paths and return regions

While a program is running, the robot or additional axis is considered to be *on path*, which means that it follows the desired sequence of positions.

If you stop the program the robot is still on path, unless you change its position. It is then considered to be *off path*. If the robot is stopped by an emergency or safety stop it may also be off path.

If the stopped robot is within the *path return region* you can start the program again, and the robot will return to the path and continue program execution.

Note that there is no way to predict the exact return movement for the robot.

TIP!

The path return region is set with system parameters. This is described in the *Technical reference manual - System parameters*, type *Path Return Region*.

Returning to path

	Action
1.	Make sure there are no obstacles blocking the way and that payload and work objects are properly placed.
2.	If necessary, put the system in automatic mode and press the Motors on button on the controller to activate the robot motors.
3.	Press the Start button on the FlexPendant to continue execution from where it stopped. One of these things will happen: • The robot or axis will slowly return to the path and execution will continue. • The Return to path dialog will be displayed.
4.	If the return to path dialog is displayed, select the proper action.

Select action

If you...	then tap...
want to return to the path and continue the program	**Yes**
want to return to the next target position and continue the program	**No**
don't want to continue the program	**Cancel**

7.2.3. Running RAPID program with uncalibrated mechanical unit

When is this useful?

If a servo gun is damaged or uncalibrated, you may want to run a service routine. In order to run the service routine (or any RAPID code), even though an additional axis is uncalibrated, the steps in this description must be followed.

How to get the program started

	Action
1.	Set the system parameter *Active at Start Up* (in type *Mechanical Unit*, topic *Motion*) to No. Set the system parameter *Disconnect at Deactivate* (in type *Measurement Channel*, topic *Motion*) to Yes. How to set system parameters is described in section*Configuring system parameters on page 283*.
2.	If any of the system parameter values where changed, restart the controller (warm start).
3.	Deactivate the uncalibrated mechanical unit. How to do this is described in section *Activating mechanical units on page 204*.
4.	Move the program pointer to Main (otherwise the mechanical unit will be automatically activated).
5.	Run the service routine or other RAPID code.

7.3 Operating modes

7.3.1. Present operating mode

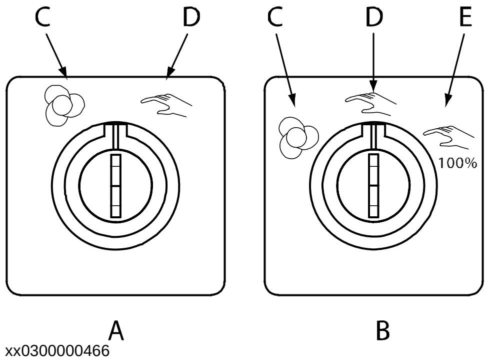
Overview

Check the position of the controller's mode switch or the status bar of the FlexPendant.

Operational mode changes are also logged in the event log.

The mode switch

The mode switch should be in the position as illustrated:



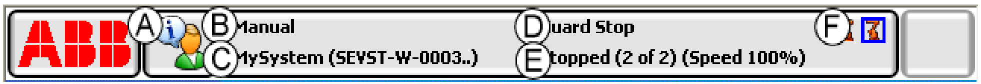
xx0300000466

A	Two position mode switch
B	Three position mode switch
C	Automatic mode
D	Manual reduced speed mode
E	Manual full speed mode

	Action	Info
1.	To switch from manual to automatic mode	detailed in *Switching from manual to automatic mode on page 243*.
2.	To switch from automatic to manual mode	detailed in *Switching from automatic to manual mode on page 245*.

Viewing present mode on the FlexPendant

On the FlexPendant, you can view the present operating mode in the status bar. An example of the status bar is shown below:



Manual
MySystem (SEVST-W-0003..)
Guard Stop
Stopped (2 of 2) (Speed 100%)

en0300000490

A	Operator window
B	Operating mode
C	Active system
D	Controller state
E	Program state
F	Mechanical units, active is highlighted

7.3.2. About the automatic mode

What is the automatic mode?

In automatic mode the enabling device is disconnected so that the robot can move without human intervention.

A robot system in production normally runs in the automatic mode. This mode enables controlling the robot system remotely, for instance by using the controller's I/O signals. An input signal may be used to start and stop a RAPID program, another to activate the robot's motors.

There are also additional safeguarding mechanisms active in automatic mode, not used in manual mode, to increase safety.

Tasks you normally perform in the automatic mode

In automatic mode you normally:

- Start and stop processes.
- Load, start and stop RAPID programs.
- Return the robot to its path when you return to operation after an emergency stop.
- Backup the system.
- Restore backups.
- Tune paths.
- Clean tools.
- Prepare or replace work objects.
- Perform other process oriented tasks.

A well designed system allows you to perform tasks safely and without affecting the running process. In such a system you can at any time enter safeguarded space temporarily having the process stopped by safeguarding mechanisms while you perform the tasks necessary. When you leave safeguarded space the process is resumed.

Please consult your plant or cell documentation for details on process oriented tasks.

CAUTION!

If the robot system is under remote control actions such as starting or stopping process applications and RAPID programs may be overridden. Path tuning may also be disturbed.

In such case perform the mentioned tasks in manual mode.

Limitations in automatic mode

Jogging is not possible in automatic mode. There may also be other specific tasks that you should perform in manual mode to make sure only you are in control of the robot and its movements.

Please consult your plant or system documentation to find out which specific tasks should not be performed in manual mode.

7.3.3. About the manual mode

What is the manual mode?

In manual mode the robot can only move in a reduced and safe speed, and only under manual control.

You need to press the enabling device to activate the robot’s motors. The manual mode is most often used when creating programs and when commissioning a robot system.

In some robot systems, there are two manual modes, the normal manual mode, sometimes referred to as Manual Reduced Speed Mode, and then there is a Manual Full Speed Mode.

What is the manual full speed mode?

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

You need to press the enabling device and the hold-to-run button to activate the robot’s motors. The manual full speed mode is most often used when testing programs and commissioning a robot system.

Note that the manual full speed mode is not available in all robot systems.

Tasks you normally perform in manual mode

In manual mode you normally:

- Jog the robot back on its path when you return to operation after an emergency stop.
- Correct the value of I/O signals after error conditions.
- Create and edit RAPID programs.
- Tune programmed positions.

Safety in manual mode

When in manual mode some safeguarding mechanisms are disabled since the robot in this mode often is operated with personnel in close proximity. Maneuvering an industrial robot is potentially dangerous and therefore maneuvers should be performed in a controlled fashion, in manual mode the robot is operated in reduced speed, normally 250 mm/s.

7.3.4. Switching from manual to automatic mode

When should I put the system in automatic mode?

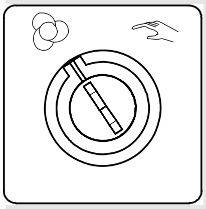
Put the system in automatic mode when you have a process application or a RAPID program that is ready to be run in production.

DANGER!

When put in automatic mode the robot may move without warning.

Make sure no personnel are in safeguarded space before you change operating mode.

Switching from manual to automatic mode

	Action	Info
1.	Set the mode switch in the automatic position. A mode change dialog is displayed.	 xx0300000467
2.	If any debug settings have been changed, a dialog informs about the changes and if these values will be reset. Tap **Acknowledge**.	If these values are reset or not is defined by system parameters in the type *Auto Condition Reset* in the topic *Controller*.
3.	Tap **OK** to close the dialog. If you change the switch back to manual mode the dialog will be closed automatically.	
4.	Did the system change mode without errors? If yes, then resume or start the process application or RAPID program. If no, stop and troubleshoot the problem.	How to start programs is described in *Starting programs on page 227*.

NOTE!

If your specific system uses a distributed operator’s panel, controls and indicators may not be placed exactly as described in this manual. Please consult your plant or cell documentation for details.

Controls and indicators do however look and function the same way.

When can I start using the robot system?

As long as the mode change dialog is displayed programs cannot be started and the robot’s motors cannot be activated either manually or remotely.

Exceptions

In automatic mode it is possible to start a RAPID program and turn motors on remotely. This means that the system will never enter a safe standby state and the robot may move at any time.

Please consult your plant or cell documentation for details on how your system is configured.

Continues on next page

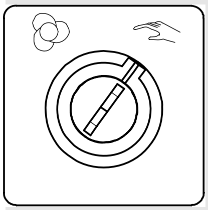
Continued

Related information

A number of conditions can be set or reset when switching to automatic mode, see *Technical reference manual - System parameters*, sections *Auto Condition Reset* and *Run Mode Settings*.

7.3.5. Switching from automatic to manual mode

Switching from automatic to manual mode

	Action	Info
1.	Set the mode switch in the manual position.	 xx0300000468
2.	Did the system change mode without errors? If yes, then this procedure is completed. If no, try to locate the error.	Error handling is detailed in *Operating manual - Trouble shooting*.

NOTE!

If your specific system uses a distributed operator’s panel, controls and indicators may not be placed exactly as described in this manual. Please consult your plant or cell documentation for details.

Controls and indicators do however look and function the same way.

7.3.6. Switching to manual full speed mode

When should I use the manual full speed mode?

Use full speed manual mode when the program is to be tested at full speed.

The manual full speed mode allows you to run the program at full speed while still having access to all the available debugging functions of the program editor.

⚠ **DANGER!**

Testing at full speed is dangerous.

Make sure no personnel are in safeguarded space when starting the program.

Switching to manual full speed mode

	Action	Info
1.	Set the mode switch to the manual full speed position.	
2.	Did the system change mode without errors? If yes, then this procedure is completed. If no, try to locate the error.	Error handling is detailed in *Operating manual - Trouble shooting*.

FlexPendant alert

When changing mode a dialog is displayed on the FlexPendant to alert you about the change of mode. Tap **OK** to close the dialog.

If you change the switch back to the previous mode the dialog will be closed automatically and there will be no change in mode.

8 Handling inputs and outputs, I/O

8.1 Basic procedures

8.1.1. Viewing signal lists

Overview

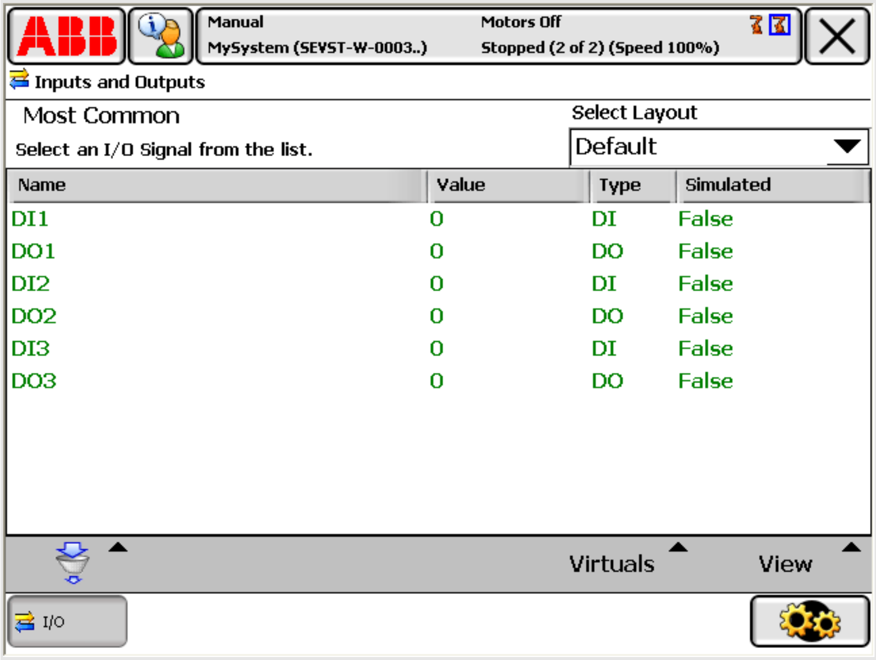
I/O signal properties is used to view the input and output signals and their values. Signals are configured with system parameters, see section *Configuring system parameters on page 283*.

How to view signal lists

This section details how to view a list of signals.

Action

1. On the **ABB** menu tap **Inputs and Outputs**.
   The list of Most Common I/O signals is displayed.



Manual
MySystem (SEVST-W-0003..)
Motors Off
Stopped (2 of 2) (Speed 100%)

Inputs and Outputs

Most Common

Select an I/O Signal from the list.

Select Layout: Default

Name	Value	Type	Simulated
DI1	0	DI	False
DO1	0	DO	False
DI2	0	DI	False
DO2	0	DO	False
DI3	0	DI	False
DO3	0	DO	False

en0400000770

2. Tap **View** to change the selection of signals in the list.

TIP!

Tap the **Select Layout** menu if you want to view signal labels in the list.

Related information

Simulating and changing signal values on page 248.

Filtering data on page 101.

Creating I/O categories on page 250.

Configuring Most Common I/O on page 312.

Configuring system parameters on page 283.

8.1.2. Simulating and changing signal values

Simulating and changing signal values

A signal can be changed into a simulated signal and the value of the signal can be changed. More information on how to change the signal's properties is described in the section Control Panel, *Configuring Most Common I/O on page 312*.

	Action
1.	On the **ABB** menu, tap **I/O**. A list of most common signals is displayed. See section *Configuring Most Common I/O on page 312*.
2.	Tap on a signal.
3.	Tap on **Simulate** to change the signal into a simulated signal. Tap on **Remove Simulation** to remove the simulation from the signal.
4.	For a digital signal, tap **0** or **1** to change the signal's value. For analog signals and groups, tap on **123...** to change the signal's value. The soft numeric keyboard is displayed. Enter the new value and tap **OK**.

8.1.3. Viewing signal group

Viewing signal group

This section details how to view signal groups.

	Action
1.	On the **ABB** menu, tap **I/O**. A list of most common signals is displayed. See section *Configuring Most Common I/O on page 312*.
2.	In the **View** menu, tap **Groups**.
3.	Tap on the signal group's name in the list and then tap **Properties**. Or tap twice on the signal group's name. The signal group's properties is displayed.

8.1.4. Creating I/O categories

I/O categories

I/O categories can be useful to filter out selections of signals. You can create your own categories. Each signal can only belong to one category.

Creating I/O categories

This section describes how to create I/O categories.

	Action
1.	On the **ABB** menu, tap **Control Panel** and then tap **Configuration**.
2.	Select topic **I/O** and then type **Signal**.
3.	The category can be set when: • creating a new signal, tap **Add** to create the signal • editing a signal, tap to select a signal and then tap **Edit** Use the parameter Category to create the category for the signal.
4.	Tap **OK** to save the signal. When a category has been set to a signal, the system must be restarted with a warm start for the changes to take effect.

Related information

Categories are used for filtering, see section *Filtering data on page 101*.

Signals are part of the configuration and edited with system parameters, see *Technical reference manual - System parameters*.

8.1.5. Deactivating and activating I/O units

Overview

Deactivating an I/O unit makes the controller ignore the unit. This can be useful during commissioning, for avoiding errors if the I/O unit is not connected to the controller yet. The signals configured on the unit will still be visible when it is deactivated, but the signal values will not be available. The controller will not attempt to send or receive any signals on a deactivated unit.

Activating the unit again will take it back to normal operation.

Deactivating and activating I/O units

This section describes how to activate I/O units.

	Action
1.	On the **ABB** menu tap **Inputs and outputs**. The list of Most common I/O signals is displayed.
2.	Tap **View** to change the selection of signals in the list. Select **I/O Units**.
3.	Tap an I/O unit in the list.
4.	Tap **Activate** or **Deactivate**.

NOTE!

All signals on the I/O unit must have an access level that allows local clients (for instance the FlexPendant) to have write access. If not, then the unit cannot be activated or deactivated from local clients. The access level is set with system parameters for each signal, see the types *Signal* and *Access Level* in the topic *I/O*.

NOTE!

The unit cannot be deactivated if the system parameter *Unit Trustlevel* is set to *0 (Required)*. *Unit Trustlevel* belongs to the type *Unit* in the topic *I/O*.

Related information

For information on how to configure an I/O unit (for instance to add and remove signals or to set the limits of the signal), see *Configuring system parameters on page 283*.

Technical reference manual - System parameters.

8.2 Safety signals

8.2.1. Safety I/O signals

General

In the controller's basic and standard form, certain I/O signals are dedicated to specific safety functions. These are listed below with a brief description of each.

All signals can be viewed in the I/O menu on the FlexPendant.

Safety I/O signals

The list below contains the safety I/O signals as used by the standard system.

Signal name	Description	Bit value condition	From - To
ES1	Emergency stop, chain 1	1 = Chain closed	From panel board to main computer
ES2	Emergency stop, chain 2	1 = Chain closed	From panel board to main computer
SOFTESI	Soft Emergency stop	1 = Soft stop enabled	From panel board to main computer
EN1	Enabling device1&2, chain 1	1 = Enabled	From panel board to main computer
EN2	Enabling device1&2, chain 2	1 = Enabled	From panel board to main computer
AUTO1	Op mode selector, chain 1	1 = Auto selected	From panel board to main computer
AUTO2	Op mode selector, chain 2	1 = Auto selected	From panel board to main computer
MAN1	Op mode selector, chain 1	1 = MAN selected	From panel board to main computer
MANFS1	Op mode selector, chain 1	1 = Man. full speed selected	From panel board to main computer
MAN2	Op mode selector, chain 2	1 = MAN selected	From panel board to main computer
MANFS2	Op mode selector, chain 2	1 = Man. full speed selected	From panel board to main computer
USERDOOVLD	Over load, user DO	1 = Error, 0 = OK	From panel board to main computer
MONPB	Motors-on pushbutton	1 = Pushbutton pressed	From panel board to main computer
AS1	Auto stop, chain 1	1 = Chain closed	From panel board to main computer
AS2	Auto stop, chain 2	1 = Chain closed	From panel board to main computer
SOFTASI	Soft Auto stop	1 = Soft stop enabled	From panel board to main computer
GS1	General stop, chain 1	1 = Chain closed	From panel board to main computer

Continues on next page

Continued

Signal name	Description	Bit value condition	From - To
GS2	General stop, chain 2	1 = Chain closed	From panel board to main computer
SOFTGSI	Soft General stop	1 = Soft stop enabled	From panel board to main computer
SS1	Superior stop, chain1	1 = Chain closed	From panel board to main computer
SS2	Superior stop, chain2	1 = Chain closed	From panel board to main computer
SOFTSSI	Soft Superior stop	1 = Soft stop enabled	From panel board to main computer
CH1	All switches in run chain 1 closed	1 = Chain closed	From panel board to main computer
CH2	All switches in run chain 2 closed	1 = Chain closed	From panel board to main computer
ENABLE1	Enable from MC (read back)	1 = Enable, 0 = break chain 1	From panel board to main computer
ENABLE2_1	Enable from AXC1	1 = Enable, 0 = break chain 2	From panel board to main computer
ENABLE2_2	Enable from AXC2	1 = Enable, 0 = break chain 2	From panel board to main computer
ENABLE2_3	Enable from AXC3	1 = Enable, 0 = break chain 2	From panel board to main computer
ENABLE2_4	Enable from AXC4	1 = Enable, 0 = break chain 2	From panel board to main computer
PANEL24OVLD	Overload, panel 24V	1 = Error, 0 = OK	From panel board to main computer
DRVOVLD	Overload, drive modules	1 = Error, 0 = OK	From panel board to main computer
DRV1LIM1	Read back of chain 1 after limit switches	1 = Chain 1 closed	From axis computer to main computer
DRV1LIM2	Read back of chain 2 after limit switches	1 = Chain 2 closed	From axis computer to main computer
DRV1K1	Read back of contactor K1, chain 1	1 = K1 closed	From axis computer to main computer
DRV1K2	Read back of contactor K2, chain 2	1 = K2 closed	From axis computer to main computer
DRV1EXTCONT	External contactors closed	1 = Contactors closed	From axis computer to main computer
DRV1PANCH1	Drive voltage for contactor-coil 1	1 = Voltage applied	From axis computer to main computer
DRV1PANCH2	Drive voltage for contactor-coil 2	1 = Voltage applied	From axis computer to main computer
DRV1SPEED	Read back of op. mode selected	0 = Man. mode low speed	From axis computer to main computer
DRV1TEST1	A dip in run chain 1 has been detected	Toggled	From axis computer to main computer
DRV1TEST2	A dip in run chain 2 has been detected	Toggled	From axis computer to main computer

Continues on next page

Continued

Signal name	Description	Bit value condition	From - To
SOFTESO	Soft Emergency stop	1 = Set soft E-stop	From main computer to panel board
SOFTASO	Soft Auto stop	1 = Set soft Auto stop	From main computer to panel board
SOFTGSO	Soft General stop	1 = Set soft General stop	From main computer to panel board
SOFTSSO	Soft Superior stop	1 = Set soft Sup. E-stop	From main computer to panel board
MOTLMP	Motors-on lamp	1 = Lamp on	From main computer to panel board
TESTEN1	Test of Enable1	1 = Start test	From main computer to panel board
DRV1CHAIN 1	Signal to interlocking circuit	1 = Close chain 1	From main computer to axis computer 1
DRV1CHAIN 2	Signal to interlocking circuit	1 = Close chain 2	From main computer to axis computer 1
DRV1BRAKE	Signal to brake-release coil	1 = Release brake	From main computer to axis computer 1

9 Handling the event log

9.1 Basic procedures

9.1.1. Accessing the event log

Event log

Open the event log to:

- view all present entries.
- study specific entries in detail.
- handle the log entries, such as saving or deleting.

The log can be printed by using RobotStudio.

Open and close the event log

This section details how to open the event log.

	Action
1.	Tap the status bar. The status window is displayed.
2.	Tap **Event Log**. The event log is displayed.
3.	If the log contents do not fit into a single screen, it can be scrolled.
4.	Tap a log entry to view the event message.
5.	Tap the status bar again to close the log.

Related information

Operating manual - RobotStudio.

9.1.2. Deleting log entries

Why should I delete log entries?

Logs can be deleted to increase available disk space. Deleting log entries is often a good way to trace faults since you remove old and insignificant log entries not related to the problem you are trying to solve.

Delete all log entries

	Action
1.	Tap the status bar, then the **Event Log** tab to open the event log.
2.	On the **View** menu, tap **Common**.
3.	Tap **Delete** and then **Delete all logs**. A confirmation dialog is displayed.
4.	Tap **Yes** to delete, or **No** to keep the log intact.

Delete log entries of a specific category

	Action
1.	Tap the status bar, then the **Event Log** tab to open the event log.
2.	On the **View** menu, tap the category of choice.
3.	Tap **Delete** and then **Delete log**. A confirmation dialog is displayed.
4.	Tap **Yes** to delete, or **No** to keep the log intact.

9.1.3. Saving log entries

Why should I save log entries?

You should save log entries when:

- you need to clear the log but want to keep the current entries to be viewed later.
- you want to send log entries to support to solve a problem.
- you want to keep log entries for future reference.

NOTE!

The log can keep up to 20 entries per category and up to 150 entries in the all events list. When the buffer is full the oldest entries will be overwritten and lost.

There is no way to retrieve these lost log entries.

Save all log entries

This section details how to save all log entries.

	Action
1.	Tap the status bar to open the event log.
2.	Tap **Save all logs as**. The file dialog is displayed.
3.	If you want to save the log in a different folder, locate and open the folder.
4.	In the **File name** box, type a name for the file.
5.	Tap **Save**.

10 Systems

10.1. What is a system?

The system

A system is the software that runs on a controller. It consists of the specific RobotWare parts for the robots connected to the controller, configuration files, and RAPID programs.

The RobotWare license key

What parts of RobotWare (supported robot models, options, etc.) that is included in the system is determined by the RobotWare license key.

When running a system on a real controller it has to be built with the license key that was delivered with the robot.

For running a system on a virtual controller (e.g. for simulations in RobotStudio) either a license key from a real robot or a virtual license key can be used. Using a license key from a real robot is a quick way to ensure that the system matches that robot. Using a virtual key provides possibility to simulate and evaluate any robot model with any configuration. A system built with a virtual key can however never be run on a real controller.

Empty system

A new system that only contains the RobotWare parts and the default configurations is called an empty system. When robot or process specific configurations are made, I/O signals are defined or RAPID programs are created, the system is no longer considered empty.

Loaded system and stored systems

The loaded system is the system that will run on the controller when it is started. A controller can only have one system loaded, but additional systems can be stored on the controller’s disk or any disk on the PC network.

It is when a system is loaded, either in a real controller or a virtual one, you normally edit its content, like RAPID programs and configurations. For stored systems, you can make some changes with the System Builder in RobotStudio, like adding and removing options and replacing whole configuration files.

10.2 Memory and file handling

10.2.1. What is “the memory”?

Overview

When using the term “memory”, a number of things may be implied:

- The main computer RAM memory
- The controller mass memory unit (hard disk, flash disk, or other drive)
- The hard disk of some other unit connected to the same LAN as the robot system, serving as a storage for software.

Main computer RAM memory

The RAM memory is the main computer primary memory located on the computer motherboard. The memory is used by the processor during all program execution.

The contents of the RAM memory during operation is described in section *The structure of the main computer RAM memory contents on page 261*.

Controller mass memory unit

This is the main mass storage unit of the controller, i.e. the controller mass memory. Depending on controller version, it may be a flash disk, hard drive, or other type and it may vary in size. It contains all necessary software for operating the robot, and is the unit on which RobotWare is installed.

When starting up, data is loaded into the RAM memory from the mass memory.

When powering down, the image.bin is saved here. The contents of the image.bin is described in section *The structure of the main computer RAM memory contents on page 261*.

LAN unit

This may be used as extra mass storage device if the one in the controller is not sufficient. It is not normally considered a part of the robot system.

10.2.2. The structure of the main computer RAM memory contents

General

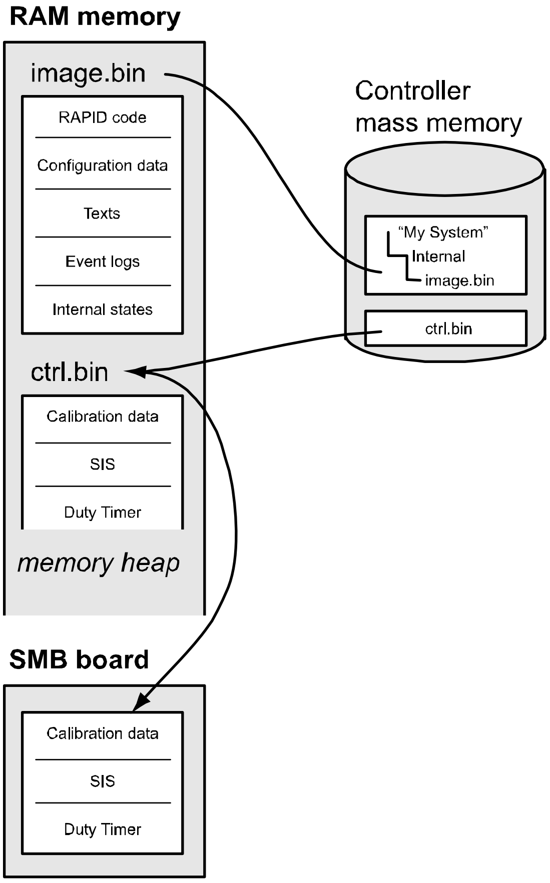
This section describes what the main computer RAM memory contains during normal operation.

The term “RAM memory” means the main computer primary memory, i.e. the memory modules with which the main computer processor works during normal operation.

The generic term “memory” is described in section *What is “the memory”? on page 260*.

Illustration of the RAM memory

Each part of the illustration is described in the table below.



en0500001446

Parts

Part	Function
RAM memory	The main computer memory modules, located on the computer motherboard. The processor reads and writes to this memory during program execution. The size of the RAM memory may vary, but increasing the size *will not improve* computer performance unless a number of hard- and software changes are made to the robot system.

Continues on next page

Part	Function
image.bin	When the system is powered OFF, intentionally or due to power failure, the image.bin file is saved to the controller mass memory. It is an internal file, created by the system during operation, usually invisible to the user. When performing a “warm start” of the system, the complete image.bin file is reloaded into the RAM memory. Other types of restarts may start with another system, etc., which is described in the *Operating manual - IRC5 with FlexPendant*.
ctrl.bin	This file contains, among other things: • robot identity data • calibration data • SIS data • duty timer data The file is stored on the SMB board on robot delivery. Data may then be transferred to the controller as detailed in the *Operating manual - IRC5 with FlexPendant*. NOTE that the ctrl.bin file is *not stored* in the system specific folder on the mass memory unit. This means that all data in the file will be retained even if the system software is updated or in any other way replaced.
SMB board	The SMB board (serial measurement board) is normally fitted on the mechanical unit, and contains among other things, data from the ctrl.bin file. How to handle the data on the SMB board, moving data between SMB and controller, etc. is detailed in the *Operating manual - IRC5 with FlexPendant*.
Controller mass memory unit	The main mass storage unit of the controller, located in the Computer Unit. Depending on controller version, it may be a flash disk, hard drive, or other type and it may vary in size. It contains all necessary software for operating the robot, and is the unit on which RobotWare is installed. When starting up, data is loaded into the RAM memory from the mass memory. When powering down, the image.bin file is automatically saved here.
RAPID code	This section contains all executable RAPID code, whether written by ABB or the customer.
Configuration data	This data is basically the contents of the configuration files: • proc.cfg • moc.cfg • sio.cfg • mmc.cfg • sys.cfg Each file contains the settings made when creating and defining the system, options etc. The configuration files may not be changed after creation, but their contents may be checked as detailed in the *Operating manual - Trouble shooting*. When changing the contents of the configuration files, ABB strongly recommends using RobotStudio to reduce the risk of introducing errors. See *Operating manual - RobotStudio*.
Texts	Some of the texts used by the system during operation, in all languages selected when creating the system.
Event logs	All events logged in all event logs. This means that the logs will be saved even if a power failure occurs, which in turn, simplifies finding the fault causing the power failure.

Continues on next page

Part	Function
Internal states	This is data recording the state and position of all robot axes, all I/O, the state of each manipulator connected to a MultiMove system, etc. This data is constantly updated during operation. This enables the system to instantly return to its previous state if the system for any reason stops, there is a power failure or the robot collides with an obstacle etc.
Calibration data	This is calibration data for one robot, i.e. all data describing the calibration position for all six axes of one robot.
SIS	This is service data related to the SIS system (Service Information System). This means that SIS data will be kept by the robot even if its controller is replaced.
Duty timer	This is the Duty timer data. This means that duty timer count will be kept by the robot even if its controller is replaced.
“My system”	This is the directory in which the RobotWare is stored after installation. The image file is stored in the directory “Internal”. NOTE that the ctrl.bin file is *not stored* here, which means that the contents of the image.bin file will be retained even if updating the system software during operation.

10.2.3. File handling

File handling and storing

Backups, programs, and configurations etc. are saved as files in the robot system. These files are handled either in a specific FlexPendant application, such as the Program editor, or using the FlexPendant Explorer.

Files can be stored on a number of different drives, or memory devices, such as:

- Controller mass memory unit
- Portable PC
- USB device
- Other network drives

These drives are used the same way and available in the FlexPendant Explorer or when saving or opening files using an application on the FlexPendant.

USB memory information

IRC5 is equipped with a USB port on the controller, see chapter *Buttons and ports on the controller on page 50*.

A USB memory is normally detected by the system and ready to use within a few seconds from plugging in the hardware. A plugged in USB memory is automatically detected during system start up.

It is possible to plug in and unplug a USB memory while the system is running. However, observing the following precautions will avoid problems:

- Do not unplug a USB memory immediately after plugging in. Wait at least five seconds, or until the memory has been detected by the system.
- Do not unplug a USB memory during file operations, such as saving or copying files. Many USB memories indicates ongoing operations with a flashing LED.
- Do not unplug a USB memory while the system is shutting down. Wait until shutdown is completed.

Please also note the following limitations with USB memories:

- There is no guarantee that all USB memories are supported.
- Some USB memories have a write protection switch. The system is not able to detect if a file operation failed due to the write protection switch.

Related information

Operating manual - Trouble shooting

What is “the memory”? on page 260.

10.3 Restart procedures

10.3.1. Restart overview

When do I need to restart a running controller?

ABB robot systems are designed to operate unattended for long times. There is no need to periodically restart functioning systems.

Restart the robot system when:

- new hardware has been installed.
- the robot system configuration files have been changed.
- a new system has been added and is to be used.
- a system failure (SYSFAIL) has occurred.

Restart types

A number of restart types are available:

Situation:	Restart type:	Detailed in section:
You want to restart and use the current system. All programs and configurations will be saved.	W-start (Warm restart)	*Restart and use the current system (warm start) on page 269*.
You want to restart and select another system. The Boot Application will be launched at startup.	X-start (Xtra restart)	*Restart and select another system (X-start) on page 270*.
You want to switch to another installed system **or** install a new system **and**, at the same time, remove the current system from the controller. **Warning!** *This can not be undone. The system and the RobotWare system package will be deleted.*	C-start (Cold restart)	*Restart and delete the current system (C-start) on page 271*.
You want to delete all user loaded RAPID programs. **Warning!** *This can not be undone.*	P-start	*Restart and delete programs and modules (P-start) on page 272*.
You want to return to the default system settings. **Warning!** *This will remove all user defined programs and configurations from memory and restart with default factory settings.*	I-start (Installation restart)	*Restart and return to default settings (I-start) on page 273*.
The system has been restarted and you want to restart the current system using the image file (system data) from the most recent successful shut down.	B-start	*Restart from previously stored system (B-start) on page 274*.
You want to shut down and save the current system and shut down the main computer.	Shutdown	*Shutting down on page 71*.

Related information

Operating manual - Trouble shooting.

10.3.2. Using the Boot Application

Boot Application

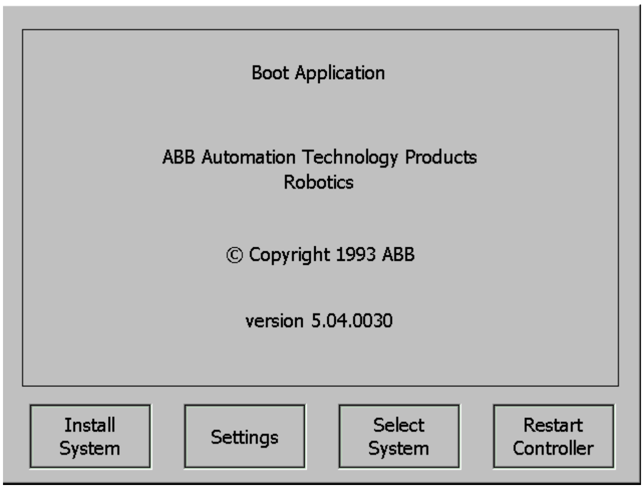
The Boot Application is primarily used to start up the system when no RobotWare is installed, but may also be used for other purposes, such as changing the system to start. You can also use RobotStudio, see *Operating manual - RobotStudio*.

Purpose of the Boot Application

The Boot Application is installed at delivery and can be used to:

- Install systems.
- Set or check network settings.
- Select a system/switch between systems from the mass storage memory.
- Load the system from USB memory units or network connections.

The illustration shows the Boot Application main screen. The buttons and functions available are described below.



Boot Application

ABB Automation Technology Products
Robotics

© Copyright 1993 ABB

version 5.04.0030

en0400000894

Installing a system

This procedure may take several minutes.

	Action	Info
1.	Perform an X-start to start the Boot Application.	X-start is detailed in section *Restart and select another system (X-start) on page 270*.
2.	Tap **Install System**.	
3.	Connect a USB memory containing a system to the computer unit USB port and tap **Continue**. If you do not have a memory stick containing a system then create a new system using the System Builder in RobotStudio.	How to load a system to the USB memory is detailed in *Operating manual - RobotStudio*. The USB port is shown in section *Buttons and ports on the controller on page 50*.

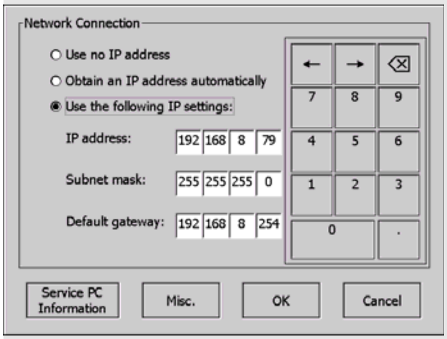
Continues on next page

Continued

	Action	Info
4.	Tap **...** to the right of the Path text box to locate the system folder on the USB memory. Select a system folder and then tap **OK**.	
5.	Tap **Continue** to start the installation. The system is read from the USB memory, and a dialog box is displayed, urging you to restart.	
6.	Tap **OK**.	The USB memory can be disconnected at this point.
7.	Tap **Restart Controller** and then tap **OK**. The controller is now restarted with the system. The restart may take several minutes.	

Boot Application settings

The Boot Application settings contain IP and network settings.

	Action	Info
1.	Perform an X-start to start the Boot Application.	X-start is detailed in section *Restart and select another system (X-start) on page 270*.
2.	Tap **Settings**.  Network Connection ○ Use no IP address ○ Obtain an IP address automatically ● Use the following IP settings: IP address: 192 168 8 79 Subnet mask: 255 255 255 0 Default gateway: 192 168 8 254 ← → ⌫ 7 8 9 4 5 6 1 2 3 0 . Service PC Information Misc. OK Cancel en0400000902	
3.	Enter your settings: • Use no IP address • Obtain IP address automatically • Use the following settings Use the numerical keyboard to enter the desired values.	These settings are detailed in section *Set up the network connection on page 56*.
4.	Tap **Service PC information** to display network settings to be used when connection a service PC to the controller service port.	
5.	Tap **Misc.** to display FlexPendant hardware and software versions. Tap **Advanced** to display the boot loader version.	

Continues on next page

	Action	Info
6.	Tap **Calibrate** to calibrate the touch screen and joystick.	Follow the instructions on the screen.
7.	Tap **Upgrade Bootloader** to load a new boot loader version	

Selecting system

	Action	Info
1.	Perform an X-start to start the Boot Application.	X-start is detailed in section *Restart and select another system (X-start) on page 270*.
2.	Tpp **Select System**. A dialog box is displayed showing the available installed systems.	
3.	Tap a system and then tap **Select**. The selected system is displayed in the box Selected System.	
4.	Tap **Close**. A dialog box is shown urging you to restart to be able to use the selected system.	

Restarting controller

	Action	Info
1.	Perform an X-start to start the Boot Application.	X-start is detailed in section *Restart and select another system (X-start) on page 270*.
2.	Tap **Restart System**. A dialog box is displayed specifying the selected system.	
3.	Tap **OK** to restart using the selected system.	

Related information

Operating manual - RobotStudio.

10.3.3. Restart and use the current system (warm start)

What happens with my current system?

The current system will be stopped.

All system parameters and programs will be saved to an image file.

During the restart process the system's state will be resumed. Static and semistatic tasks will be started. Programs can be started from the point they where stopped.

Restarting this way will activate any configuration changes entered using RobotStudio.

Restart and use the current system

This section describes how to restart and use the current system.

	Action	Info
1.	On the **ABB** menu, tap **Restart**. The restart dialog is displayed.	
2.	Tap **Warm Start** to restart the controller using the current system.	To select another type of start, tap **Advanced**. Detailed information about advanced starts is given in *Restart overview on page 265*.

10.3.4. Restart and select another system (X-start)

What happens with my current system?

The current system will be stopped.

All system parameters and programs will be saved to an image file, so that the system state can be resumed later.

Restart and select another system

This section describes how to restart and select another system.

	Action	Info
1.	Make sure the power to the controller cabinet is switched on.	
2.	On the **ABB** menu, tap **Restart**. The restart dialog is displayed.	
3.	Tap **Advanced...** to select restart method. The select restart method dialog is displayed.	
4.	Tap **X-start**, then tap **OK**. A confirmation dialog is displayed.	
5.	Tap **X-Start** to restart the controller. The controller is restarted. After the startup procedure the Boot Application is started.	
6.	Use the Boot Application to select system.	How to use the Boot Application is detailed in *Using the Boot Application on page 266*.
7.	Tap **Close**, then **OK** to return to the Boot Application.	
8.	Tap **Restart** to restart the controller using the selected system.	

10.3.5. Restart and delete the current system (C-start)

What happens with my current system?

Your current system will be stopped.

All contents, backups and programs, in the system directory **will be deleted**. This means it will be **impossible to resume** this system's state in any way. A new system must be installed using RobotStudio.

Restart and delete the current system

This section describes how to restart and delete the current system.

	Action	Info
1.	On the **ABB** menu, tap **Restart**. The restart dialog is displayed.	
2.	Tap **Advanced...** to select restart method. The select restart method dialog is displayed.	
3.	Tap **C-start**, then tap **OK**. A dialog letting you confirm that you want to restart is displayed.	
4.	Tap **C-start** to restart the controller. A dialog letting you confirm that you want to restart is displayed.	
5.	Perform any of the following procedures: • Select an already installed system and restart. • Install another system from RobotStudio or from a USB memory.	How to restart and select another system is described in section *Restart and select another system (X-start) on page 270*.

Related information

Operating manual - RobotStudio.

10.3.6. Restart and delete programs and modules (P-start)

What happens with my current system?

After restart the system’s state will be resumed except for manually loaded programs and modules. Static and semistatic tasks are started from the beginning, not from the state they had when the system was stopped.

Modules will be installed and loaded in accordance with the set configuration. System parameters will not be affected.

Restart and delete programs and modules

This section describes how to restart and delete user loaded programs and modules.

	Action
1.	On the **ABB** menu, tap **Restart**. The restart dialog is displayed.
2.	Tap **Advanced...** to select restart method. The select restart method dialog is displayed.
3.	Tap **P-start**, then tap **OK**. A dialog letting you confirm that you really want to restart is displayed.
4.	Tap **P-start** to restart the controller. The controller is restarted using the current system. After the startup procedure no programs or modules are open.

10.3.7. Restart and return to default settings (I-start)

What happens to my current system?

After restart, the system's state will be resumed but any changes done to system parameters and other settings will be lost. Instead, system parameters and other settings are read from the originally installed system on delivery.

For example, this returns the system to the original factory system state.

Restart and return to default settings

This section describes how to restart and return to default settings.

	Action
1.	On the **ABB** menu, tap **Restart**. The restart dialog is displayed.
2.	Tap **Advanced...** to select restart method. The select restart method dialog is displayed.
3.	Tap **I-start**, then tap **OK**. A dialog letting you confirm that you really want to restart is displayed.
4.	Tap **I-start** to restart the controller. The controller is restarted using the current system. Changes to system parameters and other settings are lost.

10.3.8. Restart from previously stored system (B-start)

What happens with my current system?

After restart the system uses the backup of the image file from the last successful shut down. This means that all changes made to the system after that successful shut down are lost.

When to use B-start

B-start should be used if the controller was shut down without successfully saving the image file and you want to restart the same system again. However, all changes made to the system since the last successful shut down are lost, for instance new programs, modifyed positions, or changes to system parameters.

If the system starts up with a corrupt or missing image file then the system is in system failure mode and an error message is displayed in the event log. The system must be restarted.

To the current system from the last successful shut down, then use B-start. An alternative is to use I-start (resume the originally installed system at delivery).

Using B-start when the controller is not in system failure mode due to a corrupt image file will be the same as a normal warmstart.

Restart from previously stored system data

This section describes how to restart from previously stored image file.

CAUTION!

When restarting using B-start all changes made to the system since the last successful shut down are lost and cannot be resumed.

	Action
1.	On the **ABB** menu, tap **Restart**. The restart dialog is displayed.
2.	Tap **Advanced...** to select restart method. The select restart method dialog is displayed.
3.	Tap **B-start**, then tap **OK**.
4.	Tap **B-start** to restart the controller. The controller is restarted using the image file from the most recent successful shut down.

NOTE!

After loading a backup the program pointer will most likely not agree with the actual position of the robot.

Related information

Restart and return to default settings (I-start) on page 273.

10.3.9. Reflashing firmware and FlexPendant

Overview of reflashing

After replacing hardware units, such as axis computer, buses, etc., or installing newer versions of RobotWare, the system will automatically attempt reflashing the unit in order to maintain hardware/software compatibility.

Reflashing is loading appropriate firmware (hardware specific software) onto a specific unit running this software during operation.

If RobotWare is upgraded on the controller, then the FlexPendant will reflash, i.e. update to the new version, when connected.

Note that performing any such replacements/updates might require running firmware versions not available! To avoid jeopardizing the function of the system, ABB recommends using the same versions as earlier, unless these are unavailable.

The units currently using the reflash function are:

- Contactor interface board
- Drive units
- FlexPendant
- Profibus master
- Axis computer
- Panel board

Reflashing process

The automatic reflashing process, described below, must not be disturbed by switching off the controller while running:

	Event	Info
1.	When the system is restarted, the system checks the versions of the firmware used. These are checked against the hardware versions used.	
2.	If the hardware and firmware versions do not match, the system restarts itself automatically while going to a specific *Update Mode*.	During the Update Mode, an attempt is made to download appropriate firmware onto the hardware while a message is very briefly displayed on the FlexPendant.
3.	Was an appropriate firmware version found? If YES, the reflash will proceed. If NO, the system will stop.	In either case, a message is very briefly displayed on the FlexPendant and stored in the event log. The actual reflashing may take a few seconds or up to a few minutes, depending on the hardware to be reflashed.
4.	After performing a successful reflash, the system restarts.	
5.	Another check is made for any additional hardware/firmware mismatches.	

Continues on next page

Continued

	Event	Info
6.	Was any additional mismatches found? If YES, the process is repeated until none are found. If NO, the process is complete.	

10.4 Back up and restore systems

10.4.1. What is saved on backup?

General

When performing a backup, or restoring a previously made backup, only certain data is dealt with. This section is a specification and description of these.

What *is* saved?

The backup function saves all system parameters, system modules, and program modules in a context.

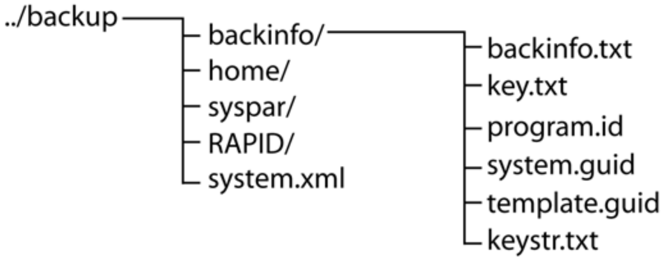
The data is saved in a directory specified by the user. A default path may be set as detailed in section *Setting default paths on page 299*.

The directory is divided into four subdirectories, Backinfo, Home, Rapid, and Syspar. System.xml is also saved in the ../backup (root directory) it contains user settings.

Backinfo

consists of the files *backinfo.txt*, *key.id*, *program.id* and *system.guid*, *template.guid*, *keystr.txt*.

The restore part uses *backinfo.txt* when the system is restored. This file must **never** be edited by the user! The files *key.id* and *program.id* may be used to recreate a system, using RobotStudio, with the same options as the backed up system. The *system.guid* is used to identify the unique system the backup was taken from. The *system.guid* and/or *template.guid* is used in the restore to check that the backup is loaded to the correct system. If the *system.guid* and/or *template.guid* do not match, the user will be informed.



en0400000916

Home

a copy of the files in the HOME directory.

Rapid

consists of a subdirectory for each task configured. Every task has one directory for program modules and one for system modules.

The first directory will keep all installed modules. More information on loading modules and programs is given in the *Technical reference manual - System parameters*.

SysPar

contains the configuration files.

Continues on next page

What *is not* saved?

A few things are not saved during backup, and it may be vital to be aware of this, in order to save these separately:

- The environment variable RELEASE: points out the current system pack. System modules loaded with RELEASE: as its path, are not stored in the backup.
- The current value of a PERS object in a installed module is not stored in a backup.

Related information

Technical reference manual - System parameters.

Operating manual - RobotStudio.

10.4.2. Back up the system

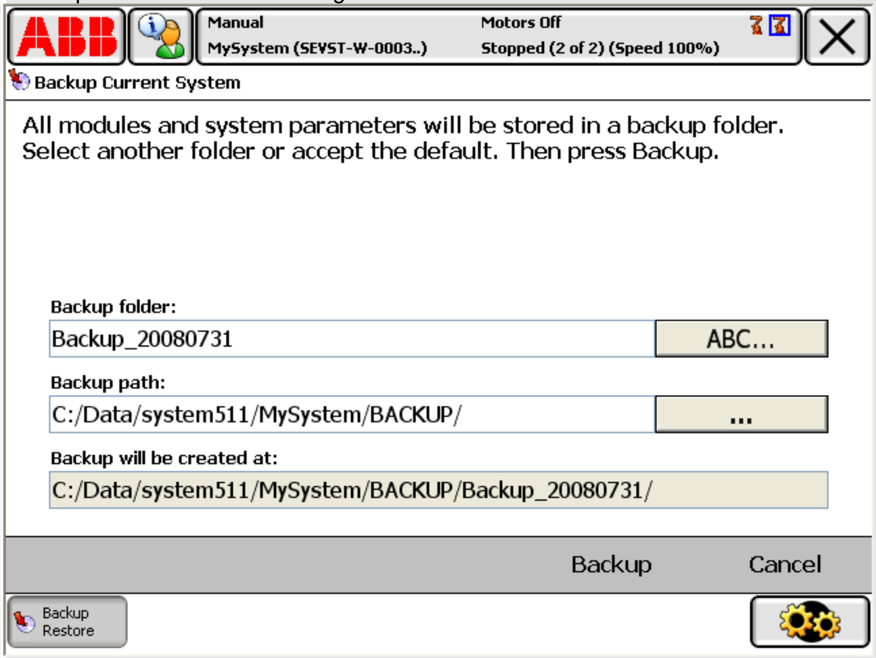
When do I need this?

ABB recommends performing a backup:

- *before* installing new RobotWare.
- *before* making any major changes to instructions and/or parameters to make it possible to return to the previous setting.
- *after* making any major changes to instructions and/or parameters and testing the new settings to retain the new successful setting.

Back up the system

This section describes how to back up the system.

	Action
1.	Tap the **ABB** menu and then tap **Backup and Restore**.
2.	Tap **Backup Current System**. A display showing the selected path is shown. If a default path has been defined as detailed in section *Back up the system on page 279*, this is shown.
3.	Is the displayed backup path the correct one? If YES: Tap **Backup** to perform the backup to the selected directory. A backup file named according to the current date is created. If NO: Tap **...** to the right of the backup path and select directory. Then tap **Backup**. A backup folder named according to the current date is created.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Backup Current System All modules and system parameters will be stored in a backup folder. Select another folder or accept the default. Then press Backup. Backup folder: Backup_20080731 Backup path: C:/Data/system511/MySystem/BACKUP/ Backup will be created at: C:/Data/system511/MySystem/BACKUP/Backup_20080731/ xx0300000441

10.4.3. Restore the system

When do I need this?

ABB recommends performing a restore:

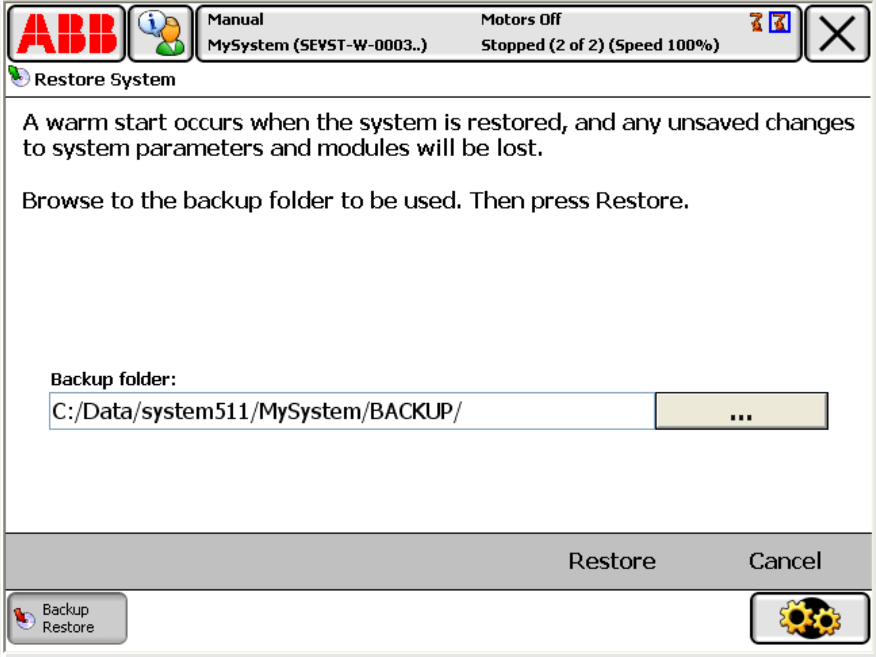
- if you suspect that the program file is corrupt.
- if any changes made to the instructions and/or parameters settings did not prove successful, and you want to return to the previous settings.

During the restore, all system parameters are replaced and all modules from the backup directory are loaded.

The Home directory is copied back to the new system’s HOME directory during the warm start.

Restore the system

This section describes how to restore the system.

	Action
1.	On the **ABB** menu, tap **Backup and Restore**.
2.	Tap **Restore System**. A display showing the selected path is shown. If a default path has been defined as detailed in section *Restore the system on page 280*, this is shown.
3.	Is the displayed backup folder the correct one? If YES: Tap **Restore** to perform the restore. The restore is performed, and the system is warm started automatically. If NO: Tap **...** to the right of the backup folder and select directory. Then tap **Restore**. The restore is performed, and the system is warm started automatically.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Restore System Restore Cancel Backup Restore A warm start occurs when the system is restored, and any unsaved changes to system parameters and modules will be lost. Browse to the backup folder to be used. Then press Restore. Backup folder: C:/Data/system511/MySystem/BACKUP/ ... xx0300000442

10.4.4. Important when performing backups!

General

When performing backups or restoring previously made backups, there are several things to keep in mind. Some of these are listed below.

BACKUP directory

A local default backup directory, BACKUP, is automatically created by the system. We recommend using this directory for saving backups! Such backups are not copied to the directory HOME in following backups.

Never change the name of the BACKUP directory.

Also, never change the name of the actual backup to BACKUP, since this would cause interference with this directory.

A default path may be created to any location on the network where the backup should be stored. How to perform this is detailed in section *Setting default paths on page 299*.

When is backup possible?

A backup of a system may be performed during program execution. When doing so, a few limitations apply:

- Start program, load program, load module, close program and erase module can not be done during backup in executing state. The RAPID instructions `Load` and `StartLoad` can, however, be used.

What happens during backup?

Beside the obvious, a backup being made, a few thing happen during backup:

- Background tasks continue to execute during a backup.

Duplicated modules?

No save operation is performed in the backup command. This implies that two revisions of the same modules can exist in the backup, one from the program memory saved in Rapid\Task\Progmod\ directory and one from the HOME directory copied to the backup's home directory.

Large data amount

Too many files in the HOME directory can result in a very large backup directory. The unnecessary files in the home directory can then be deleted without any problems.

Faults during backup

If a fault occurs during the backup, e.g. full disk or power failure, the whole backup structure is deleted.

10.5 Identifying system failure

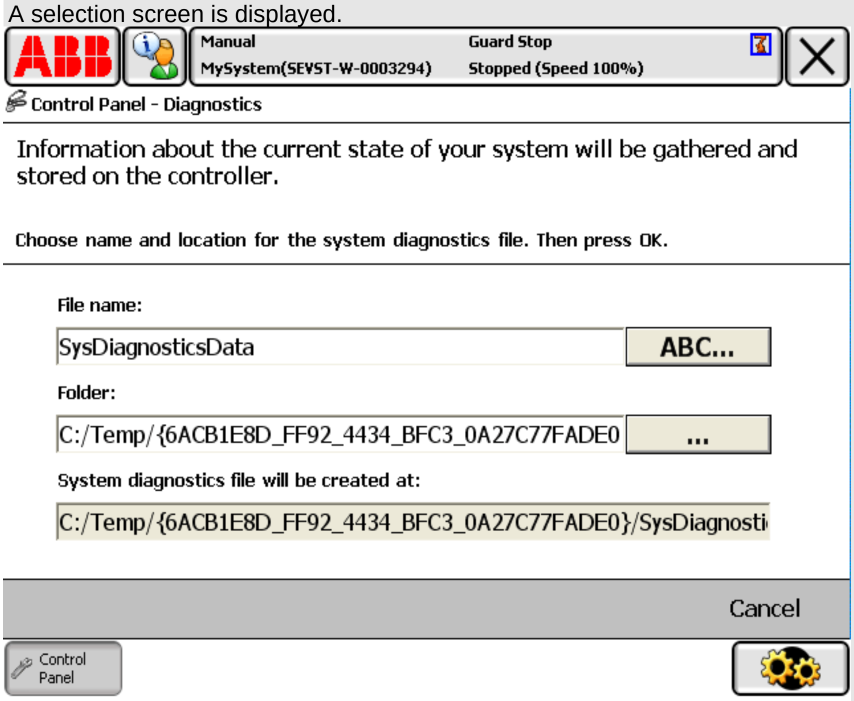
10.5.1. Creating a diagnostic file

When do I need this?

The diagnostic file can be useful when contacting ABB technical support personnel for trouble shooting. The diagnostic file contains the setup and a number of test results from your system. For more information, see *Operating manual - Trouble shooting*, section *Instructions, how to correct faults - Filling an error report*.

Create a diagnostic file

This section describes how to create a diagnostic file.

	Action
1.	On the **ABB** menu, tap **Control Panel**.
2.	Tap **Diagnostic**. A selection screen is displayed.  Manual MySystem(SEVST-W-0003294) Guard Stop Stopped (Speed 100%) Control Panel - Diagnostics Information about the current state of your system will be gathered and stored on the controller. Choose name and location for the system diagnostics file. Then press OK. File name: SysDiagnosticsData Folder: C:/Temp/{6ACB1E8D_FF92_4434_BFC3_0A27C77FADE0 System diagnostics file will be created at: C:/Temp/{6ACB1E8D_FF92_4434_BFC3_0A27C77FADE0}/SysDiagnosti en06000002630
3.	Tap **...** next to the File name to change the name of the diagnostic file.
4.	Tap **...** next to the Folder to change the destination for the file name.
5.	Tap **OK** to make a diagnostic file from the current system or tap **Cancel** to go back to the Control Panel

10.6 Configuring systems

10.6.1. Configuring system parameters

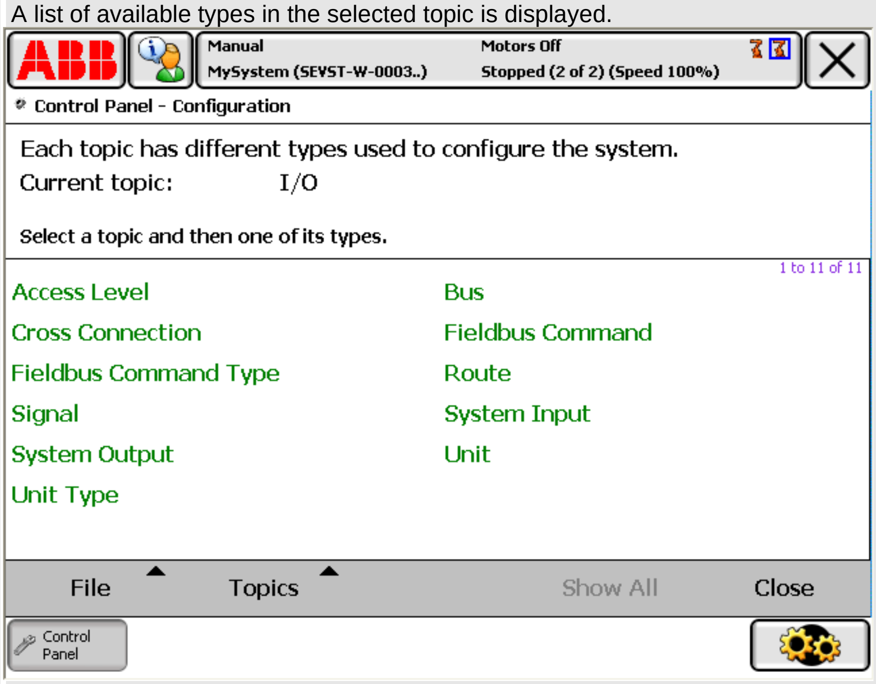
About system parameters

System parameters define the system configuration and are defined to order on delivery.

System parameters are edited using the FlexPendant or RobotStudio.

Viewing system parameters

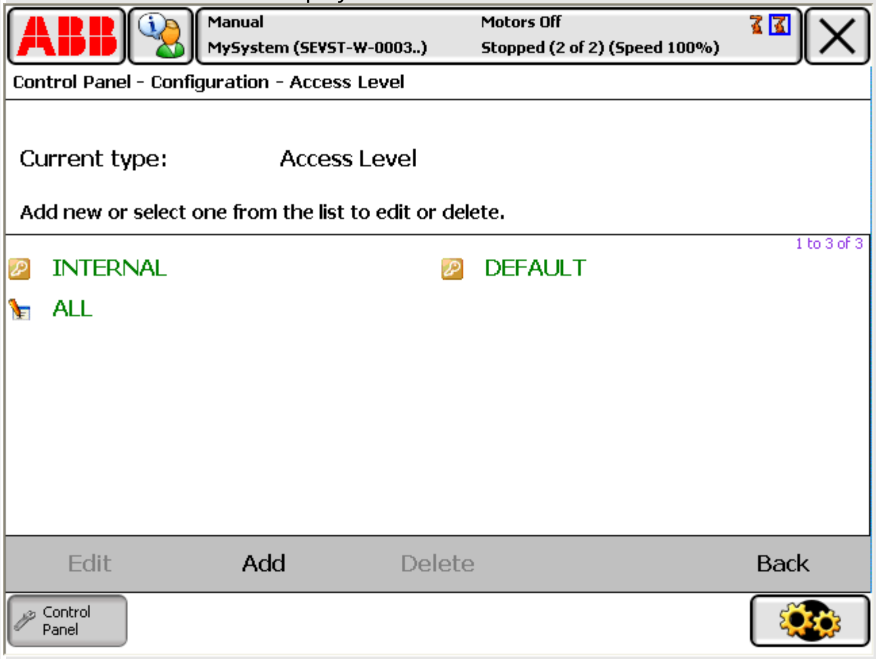
This procedure describes how to view system parameter configurations.

	Action
1.	On the **ABB** menu, tap **Control Panel**.
2.	Tap **Configuration**. A list of available types in the selected topic is displayed.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - Configuration 1 to 11 of 11 Access Level Bus Cross Connection Fieldbus Command Fieldbus Command Type Route Signal System Input System Output Unit Unit Type File Topics Show All Close Control Panel Each topic has different types used to configure the system. Current topic: I/O Select a topic and then one of its types. en0400001149
3.	Tap **Topics** to select the topic. • Controller • Communication • I/O • Man-machine Communication • Motion
4.	Tap **File** to save, load, or add new parameters from a file. Select folder and save or load. Proceed to section *Saving, loading system parameter configurations on page 285*.
5.	Tap to select a type and then tap **Show All**. To edit parameters proceed to section *Editing an instance on page 284*. To add instances proceed to section *Adding a new instance on page 284*.

Continues on next page

Editing an instance

This section describes how to edit an instance of a system parameter type.

	Action
1.	In the list of system parameter instances, tap to select an instance and then tap **Edit**. The selected instance is displayed.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - Configuration - Access Level 1 to 3 of 3 INTERNAL DEFAULT ALL Edit Add Delete Back Control Panel Current type: Access Level Add new or select one from the list to edit or delete. en0400001151
2.	Tap a parameter name or its value to edit the value. The way to edit values depend on the data type for the value, e.g. the soft keyboard is displayed for string or numerical values and dropdown menus are displayed for predefined values.
3.	Tap **OK**.

Adding a new instance

This section describes how to add a new instance of a system parameter type.

	Action
1.	In the list of system parameter instances, tap **Add**. A new instance with default values is displayed.
2.	Tap the parameter name or its value to edit the value.
3.	Tap **OK**.

Continues on next page

Saving, loading system parameter configurations

This section describes how to save system parameter configurations. It is recommended to save the parameter configurations before making larger changes to the robot system. The parameters are saved automatically when performing backups.

	Action
1.	In the list of types, tap the **File** menu and tap: • **Save As** to save the selected topic's parameter configurations. • **Save All As** to save all topics' parameter configurations.
2.	Select directory where you want to save the parameters.
3.	Tap **OK**.

Loading system parameters

This section describes how to load system parameter configuration and how to add parameters from a file.

	Action
1.	In the list of types, tap the **File** menu and tap **Load Parameters**.
2.	Select one of these actions, then tap **Load**: • Delete existing parameters before loading • Load parameters if no duplicates • Load parameters and replace duplicates.
3.	Select the directory and file where you want to load the parameters, then tap **OK**.

Related information

Technical reference manual - System parameters.

11 Calibrating

11.1 Basic procedures

11.1.1. How to check if the robot needs calibration

Check robot calibration status

This section describes how to check the robot's calibration status.

	Action
1.	On the **ABB** menu, tap **Calibration**.
2.	In the list of mechanical units, check the calibration status.

What kind of calibration is needed?

If the calibration status is...	then...
Not calibrated	the robot must be calibrated by a qualified service technician. See section *Loading calibration data using the FlexPendant on page 290*.
Rev. counter update needed	You must update the revolution counters. How to update the revolution counters is described in section *Updating revolution counters on page 288*.
Calibrated	No calibration is needed.

⚠ **DANGER!**

Do not attempt to perform the fine calibration procedure without the proper training and tools. Doing so may result in incorrect positioning that may cause injuries and property damage.

11.1.2. Updating revolution counters

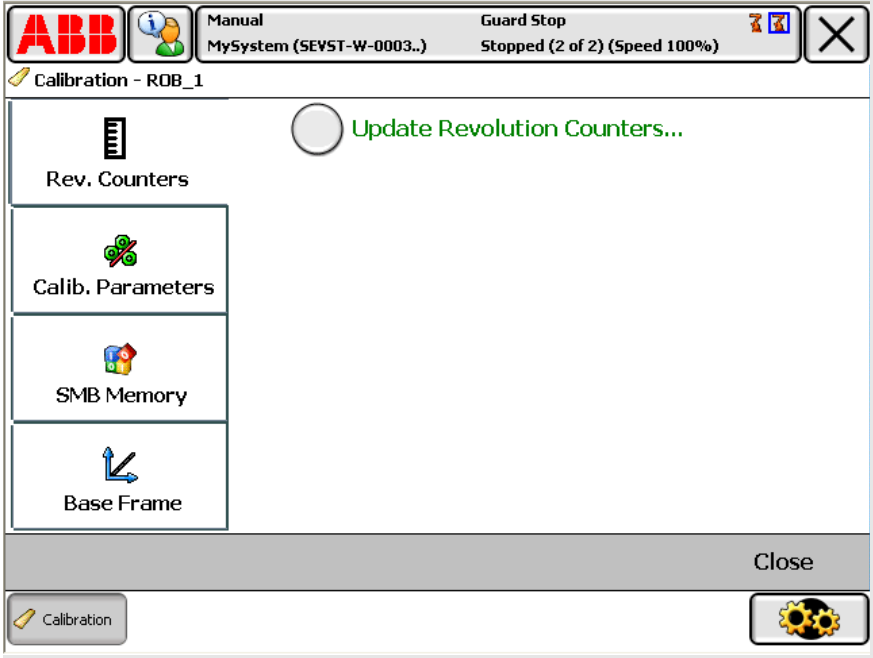
Overview

This section details how to perform a rough calibration of each robot axis, that is updating the revolution counter value for each axis, using the FlexPendant. Detailed information about revolution counters and how to update them, with calibration positions and scales, can be found in the respective robot product manual. Also, see the manual *Operating manual - Calibration Pendulum* for information on calibration.

For robots using the *Absolute Accuracy* option, the calibration data file absacc.cfg must be loaded first.

Storing the revolution counter setting

This procedure details the second step when updating the revolution counter; storing the revolution counter setting.

	Action
1.	On the **ABB** menu, tap **Calibration**. All mechanical units connected to the system are shown along with their calibration status.
2.	Tap the mechanical unit in question. A screen is displayed: tap **Rev. Counters**.  Manual MySystem (SEVST-W-0003..) Guard Stop Stopped (2 of 2) (Speed 100%) Calibration - ROB_1 Rev. Counters Calib. Parameters SMB Memory Base Frame ○ Update Revolution Counters... Close Calibration en0400000771
3.	Tap **Update revolution counters...**. A dialog box is displayed, warning that updating the revolution counters may change programmed robot positions: • Tap **Yes** to update the revolution counters. • Tap **No** to cancel updating the revolution counters. Tapping **Yes** displays the axis selection window.

Continues on next page

Continued

	Action
4.	Select the axis to have its revolution counter updated by: • Ticking in the box to the left • Tapping **Select all** to update all axes. Then tap **Update**.
5.	A dialog box is displayed, warning that the updating operation cannot be undone: • Tap **Update** to proceed with updating the revolution counters. • Tap **Cancel** to cancel updating the revolution counters. Tapping **Update** updates the selected revolution counters and removes the tick from the list of axes.
6.	❗ **CAUTION!** If a revolution counter is incorrectly updated, it will cause incorrect robot positioning, which in turn may cause damage or injury! Check the calibration position very carefully after each update. See section *Checking the calibration position* in either of the calibration manuals, depending on which calibration method to be used. The Product manual for the robot also contains more information about calibration.

Related information

Operating manual - Calibration Pendulum

11.1.3. Loading calibration data using the FlexPendant

Overview

This section describes how to load calibration data for using the FlexPendant.

The calibration data is delivered on a diskette and will have to be moved to a USB memory or transferred to the controller through FTP.

Load calibration data

This section describes how to load the calibration data.

	Action	Info
1.	On the **ABB** menu, tap **Calibration** and select a mechanical unit. Then tap **Calib. parameters**.	
2.	Tap **Load motor calibration...**. A dialog box is displayed, warning that loading new calibration offset values may change programmed robot positions: • Tap **Yes** to proceed. • Tap **No** to cancel.	Tapping **Yes** results in displaying a file selection window.
3.	For systems *not* running the *Absolute Accuracy* option the calibration data is normally stored on the serial measurement board (SMB).	In such case, Update data to controller from SMB memory as detailed in section *Serial Measurement Board memory on page 294*
4.	For system running the *Absolute Accuracy* option, the calibration data is normally delivered on a diskette.	In such case, proceed below.
5.	Select the *file containing the Absolute Accuracy calibration data* to be loaded into the system and tap **OK**. If a file containing invalid calibration data is selected, a dialog box will be displayed. Then re-select a file containing valid calibration data.	*Absacc.cfg* for systems *with* Absolute Accuracy measurement system

11.1.4. Editing motor calibration offset

Editing motor calibration offset

This procedure should be used when no specific file with motor calibration data is available, but only the numerical values. These values are normally found on a sticker on the rear of the robot.

Entering motor calibration values may be done in one of three ways:

- From a disk, using the FlexPendant (as detailed in section *Loading calibration data using the FlexPendant on page 290*).
- From a disk, using RobotStudio (as detailed in *Operating manual - RobotStudio*).
- Manually entering the values, using the FlexPendant (as detailed in section *Editing motor calibration offset on page 291*).

	Action	Info
1.	On the **ABB** menu, tap **Calibration**.	
2.	Tap to select mechanical unit and then tap **Calibration Parameters**.	
3.	Tap **Edit motor calibration offset...**. A dialog box is displayed, warning that updating the revolution counters may change programmed robot positions: • Tap **Yes** to proceed. • Tap **No** to cancel. Tapping **Yes** results in displaying a file selection view.	
4.	Tap the axis to have its motor calibration offset edited. The offset value box is opened for that particular axis.	
5.	Use the numerical keyboard to enter the desired value and then tap **OK**. After entering new offset values, a dialog box is displayed, urging you to restart the system to make use of the new values. Perform a warm restart if required.	
6.	After restarting, the contents of the calibration data in the controller cabinet and on the serial measurement board will differ. Update the calibration data.	Detailed in section *Serial Measurement Board memory on page 294*
7.	Update the revolution counters.	Detailed in section *Updating revolution counters on page 288*

11.1.5. Fine calibration procedure on FlexPendant

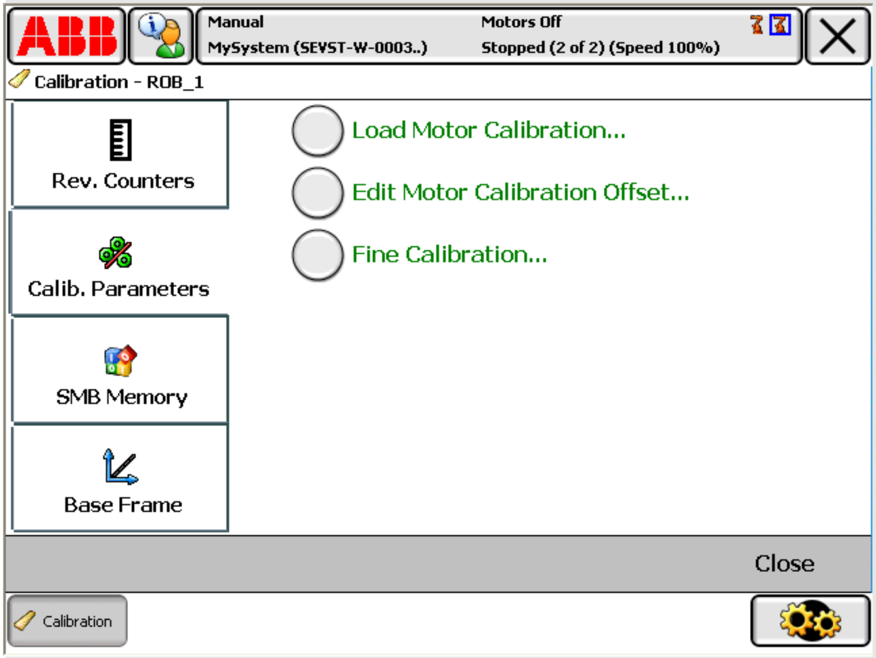
Overview

This section describes how to use the FlexPendant when performing a fine calibration of the robot.

The method of fitting the calibration equipment to each axis is detailed in the calibration instruction for the axis, see the Product manual for the robot.

Fine calibration procedure

The procedure below details how to perform the fine calibration procedure on the FlexPendant.

	Action
1.	**WARNING!** Do not fine calibrate the robot without special equipment used for axis calibration! It would cause an unsatisfied accuracy in the robot movement.
2.	On the **ABB** menu, tap **Calibration**. All mechanical units connected to the system are shown along with their calibration status.
3.	Tap to select the mechanical unit. A screen is displayed: tap **Calib. Parameter**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Calibration - ROB_1 Rev. Counters Calib. Parameters SMB Memory Base Frame ○ Load Motor Calibration... ○ Edit Motor Calibration Offset... ○ Fine Calibration... Close Calibration en0400001127

Continues on next page

	Action
4.	Tap **Fine Calibration...**. A dialog box is displayed, urging you to use external equipment to performing the actual calibration. Make sure all necessary calibration equipment is fitted, as detailed in the calibration instruction, for the axis to be calibrated. A warning that updating the revolution counters may change programmed robot positions is also displayed: • Tap **Yes** to proceed. • Tap **No** to cancel.
5.	Select the axis to calibrate by ticking the box to the left.
6.	Tap **Calibrate**. A dialog box is displayed, warning that calibration of the selected axes will be changed, which cannot be undone: • Tap **Calibrate** to proceed. • Tap **Cancel** to cancel. Tapping **Calibrate** results in briefly displaying a dialog box, announcing that the calibration process has started. The axis is calibrated and the system returns to the list of available mechanical units.

11.1.6. Serial Measurement Board memory

Serial Measurement Board, SMB

The Serial Measurement Board, SMB, primarily gathers resolver data from the robot’s (or additional axes) motors. This data is used to measure the speed and position of each axis. Each SMB is capable of measuring up to 7 axes. It also stores a number of data pertaining to each robot.

This data is used by the controller and can be transferred between the SMB and the controller. Normally, the data is transferred automatically, but it can also be done manually.

The SMB data is affected when:

- the robot is replaced
- the SMB is replaced
- the controller (or its flash disk or mass memory unit) is replaced.
- updating with new calibration data

The following data is stored on the SMB:

- serial number for the mechanical unit
- joint calibration data
- Absolute Accuracy data
- SIS data (Service Information System)

Note that if the IRC5 controller is to be connected to a robot with an older SMB, not equipped with data storage capability, the SMB must be replaced.

SMB data update

If...	then...
the **flash disk** or mass memory or the complete **controller** is new or replaced by an unused spare part...	the data stored in the SMB is automatically copied to the controller memory.
the **SMB** is replaced by a new, unused, spare part SMB...	the data stored in the controller memory is automatically copied to the SMB memory.
the **flash disk** or the complete **controller** is replaced by a spare part, previously used in another system...	the data in the controller memory and the SMB memory is different. You must **update the controller memory manually** from the the SMB memory.
the **SMB** is replaced by a spare part SMB, previously used in another system...	the data in the controller memory and the SMB memory is different. You must first **clear the data in the new SMB memory**, and then **update the SMB memory** with the data from the controller memory.
new **calibration data** has been loaded via RobotStudio or using the FlexPendant and the system has been restarted...	the data in the controller memory and the SMB memory is different. You must **update the SMB memory manually** from the controller memory. Check that the new calibration values belong to a manipulator with the serial number defined in your system.

Continues on next page

View SMB data status

This section describes how to view the data status in the Serial Measurement Board and the controller.

	Action
1.	On the **ABB** menu, tap **Calibration** and select a mechanical unit.
2.	Tap **SMB memory** and then tap **Show status**. The data is displayed with status on the SMB and on the controller.

Update controller data from SMB memory

This section describes how to load data from the Serial Measurement Board **to** the controller.

	Action	Info
1.	On the **ABB** menu, tap **Calibration** and select a mechanical unit.	
2.	Tap **SMB memory** and then tap **Update**.	
3.	Tap the button **Cabinet or manipulator has been exchanged**. A warning is displayed. Tap **Yes** to proceed or **No** to cancel.	It is vital that you load calibration data correctly.
4.	The data is loaded. Tap **Yes** to acknowledge and restart the robot system.	The following data is updated: • serial numbers for mechanical units • calibration data • Absolute Accuracy data • SIS data

Update data in SMB memory

This section describes how to update data on the Serial Measurement Board **from** the controller. This is e.g. after calibration data has been loaded to the controller via RobotStudio or using the FlexPendant.

If the SMB already contains data, you must first clear the memory, see *Delete SMB data on page 296*.

	Action	Info
1.	On the **ABB** menu, tap **Calibration** and select a mechanical unit.	
2.	Tap **SMB memory** and then tap **Update**.	
3.	Tap the button **Serial measurement board has been replaced**. A warning is displayed. Tap **Yes** to proceed or **No** to cancel.	It is vital that you load calibration data correctly.
4.	The data is updated.	

Continues on next page

Delete SMB data

This section describes how to delete the data stored on the SMB memory or the controller memory, when creating spare parts.

	Action
1.	On the **ABB** menu, tap **Calibration** and tap to select a mechanical unit.
2.	Tap **SMB memory** and then tap **Advanced**. The following functions are available: • Clear Cabinet Memory • Clear SMB Memory
3.	Tap **Clear Cabinet Memory** if the controller should be replaced and used as a spare part. A list of the SMB data stored in the controller is displayed. Tap **Clear** to delete the memory for the selected robot. Repeat the procedure for all robots in the controller memory.
4.	Tap **Clear SMB Memory** if the SMB should be replaced and used as a spare part. A list of the SMB data stored is displayed. Tap **Clear** to delete the memory for the selected robot. Repeat the procedure for all robots using this SMB board.

Related information

Operating manual - RobotStudio.

Operating manual - Service Information System.

Application manual - Motion performance.

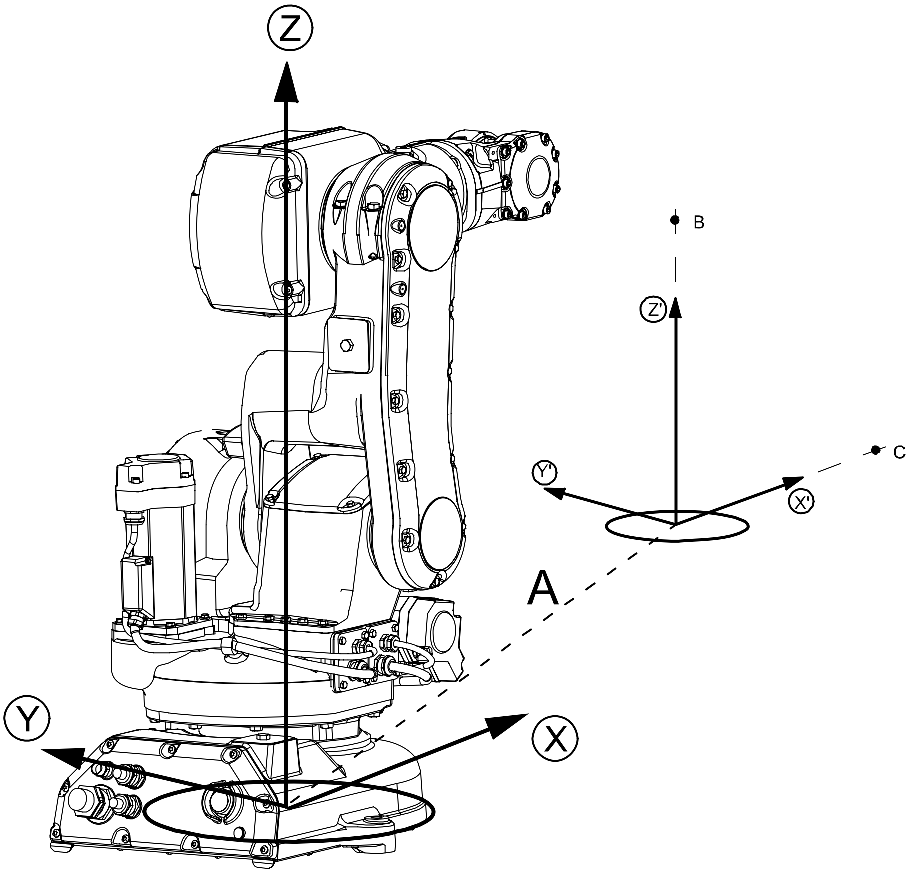
11.1.7. 4 points XZ calibration

Base Frame calibration

This section describes the 4 points XZ calibration, in the Base Frame calibration options. Other calibration methods may be available in this menu depending on your installed options.

Overview

This section describes how to define the base frame using the 4 points XZ method. This method can move and rotate the base frame in relation to the world frame. Normally the base frame is centered and aligned with the world frame. Note that the base frame is fixed to the base of the robot.



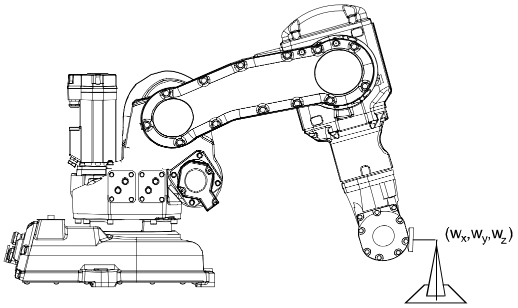
xx0400000782

A	Displacement distance between base frame and world frame
B	Elongator point Z’
C	Elongator point X’
X	X-axis in the base frame
Y	Y-axis in the base frame
Z	Z-axis in the base frame
X'	X-axis in the world frame
Y'	Y-axis in the world frame
Z'	Z-axis in the world frame

Continues on next page

Fixed reference Position

The calibration procedure requires that the tip of the tool is calibrated against a fixed reference position. The fixed position could be a manufactured World fixed tip device to facilitate finding the elongator points. The fixed reference position is the distance (in (x,y,z)) between the fixed position and the world frame.



xx0600003322

Running 4 points XZ calibration

	Action	Info
1.	On the **ABB** menu, tap **Calibration** and select a mechanical unit. Then tap **Base Frame**.	
2.	Tap **4 points XZ...**.	
3.	Set up a fixed reference position within the working range of the robot.	
4.	Tap **...** to change reference point. Enter the coordinates of the fixed reference position A numerical keyboard and boxes for X, Y and Z values are displayed.	
5.	If the calibration positions exists in a file, follow the instructions below. Otherwise proceed to the next step • Tap **Positions** menu and then **Load** the file containing the values.	
6.	Tap **Point 1** to highlight the line.	
7.	Manually jog the robot to the previously fixed reference point.	
8.	Tap **Modify position**. Modified is displayed on the status line.	
9.	Re-orient the robot and again, run it to the reference point but from a different angle.	Repeat these steps until points 1, 2, 3, and 4 have been modified.
10.	Tap **Elongator X** and manually run the robot to a position where the tool center point (TCP) touches an imaginary extension of the X-axis.	The imaginary X-axis is shown in the illustration above.
11.	Tap **Modify position**. Modified is displayed on the status line.	Repeat these steps to modify **Elongator Z**.
12.	To save the entered transformation data to a file, tap the **Positions** menu and then **Save**. Enter the name of the file and then tap **OK**.	
13.	To delete all entered transformation data, tap the **Positions** menu and then **Reset All**.	

12 Changing FlexPendant settings

12.1 System settings

12.1.1. Setting default paths

Overview

You may set individual default paths for performing a number of actions using the FlexPendant.

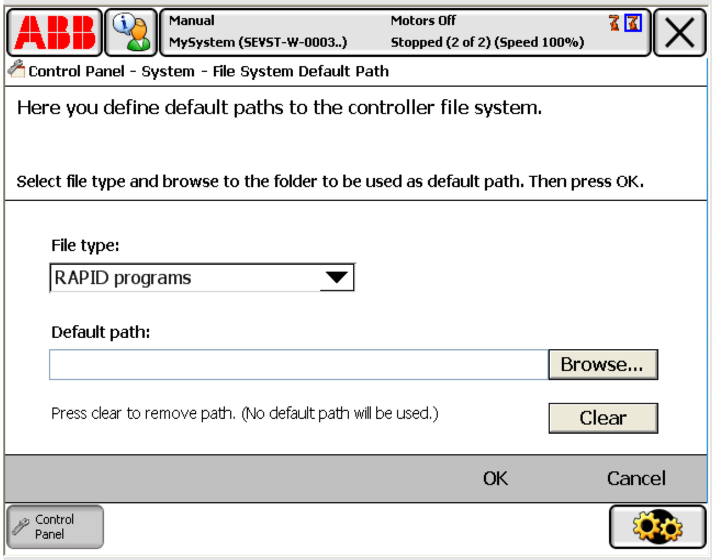
These are listed below:

- Backup and restore
- Saving and storing configuration files
- Saving and loading RAPID programs
- Saving and loading RAPID modules

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

How to set default paths

The procedure below details how to set a default path for the functions listed above:

	Action
1.	On the **ABB** menu, tap **Control Panel** and then **FlexPendant**.
2.	Tap **File System Default Path**. A selection screen is shown:  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - System - File System Default Path Here you define default paths to the controller file system. Select file type and browse to the folder to be used as default path. Then press OK. File type: RAPID programs Default path: Browse... Press clear to remove path. (No default path will be used.) Clear OK Cancel Control Panel en0500002361
3.	Tap the **File type** menu to choose from: • RAPID programs • RAPID modules • Backup/restore files • Configurations files
4.	Type the default path or tap **Browse**, to choose the desired location.

Continues on next page

Continued

	Action
5.	If required, any previously entered path may be removed by tapping **Clear**.
6.	Tap **OK**.

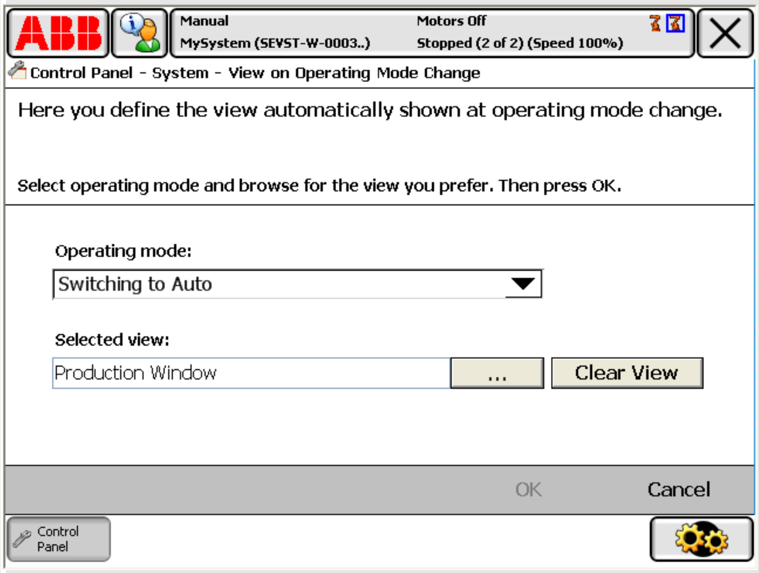
12.1.2. Defining a view to be shown at operating mode change

View on operating mode change

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

How to define view on operating mode change

This procedure details s how to configure the FlexPendant to automatically, at an operating mode change, show a specified view.

	Action
1.	On the **ABB** menu, tap **Control Panel** and then tap **FlexPendant**.
2.	Tap **View on Operating Mode change**.  Control Panel - System - View on Operating Mode Change Here you define the view automatically shown at operating mode change. Select operating mode and browse for the view you prefer. Then press OK. Operating mode: Switching to Auto Selected view: Production Window Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) ... Clear View OK Cancel Control Panel en0400001152
3.	Tap the **Operating mode** menu to select the mode change to be defined: • Switching to Auto • Switching to Manual • Switching to Manual Full Speed
4.	Tap **...** and select the desired application from the list.
5.	Tap **OK**.

NOTE!

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

12.1.3. Changing the background image

Background images

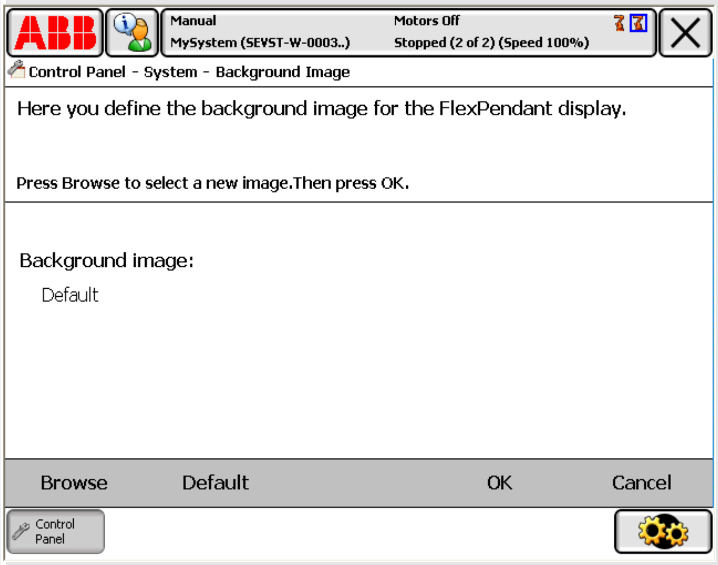
The background image on the FlexPendant can be changed. Any image file on the controller hard disk can be used, a photo as well as an illustration.

For best result, use an image following these recommendations:

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

How to change background image

This procedure details how to change background image on FlexPendant.

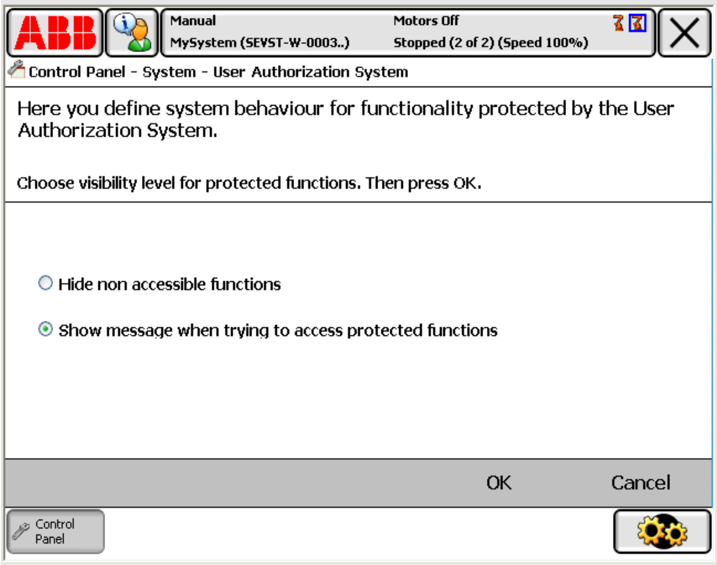
	Action
1.	On the **ABB** menu, tap **Control panel**.
2.	Tap **FlexPendant** and then **Background Image**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - System - Background Image Here you define the background image for the FlexPendant display. Press Browse to select a new image.Then press OK. Background image: Default Browse Default OK Cancel Control Panel en0500001547
3.	Tap **Browse** to locate another picture on the controller hard disk.
4.	Tap **Default** to restore the original background image.
5.	Tap **OK**.

12.1.4. Defining visibility level for UAS protected functions

Overview

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

How to define visibility level for UAS protected functions

	Action
1.	On the **ABB** menu, tap **Control Panel** and then tap **FlexPendant**.
2.	Tap **User Authorization System**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - System - User Authorization System Here you define system behaviour for functionality protected by the User Authorization System. Choose visibility level for protected functions. Then press OK. ○ Hide non accessible functions ● Show message when trying to access protected functions en0400001153
3.	Tap to select the level of visibility for UAS protected functions: • Hide non accessible functions OR • Show message when trying to access protected functions.
4.	Tap **OK**.

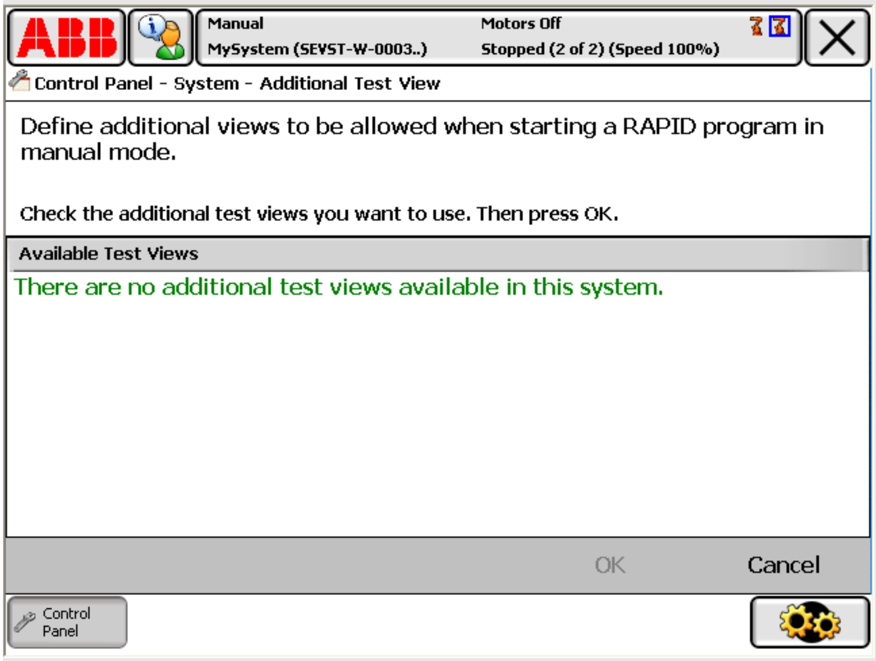
12.1.5. Defining an additional test view

Overview

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

How to define an additional test view

This procedure details how to define an additional test view.

	Action
1.	On the **ABB** menu, tap **Control Panel** and then **FlexPendant**.
2.	Tap **Additional Test View**. The displayed screen might look like this:  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - System - Additional Test View Define additional views to be allowed when starting a RAPID program in manual mode. Check the additional test views you want to use. Then press OK. Available Test Views There are no additional test views available in this system. en0600003110
3.	Usually only the Program Editor and the Production Window are allowed test views. In case there are additional views to choose from, these will appear in the list. Check one or several applications to be used as additional test views.
4.	Tap **OK**.

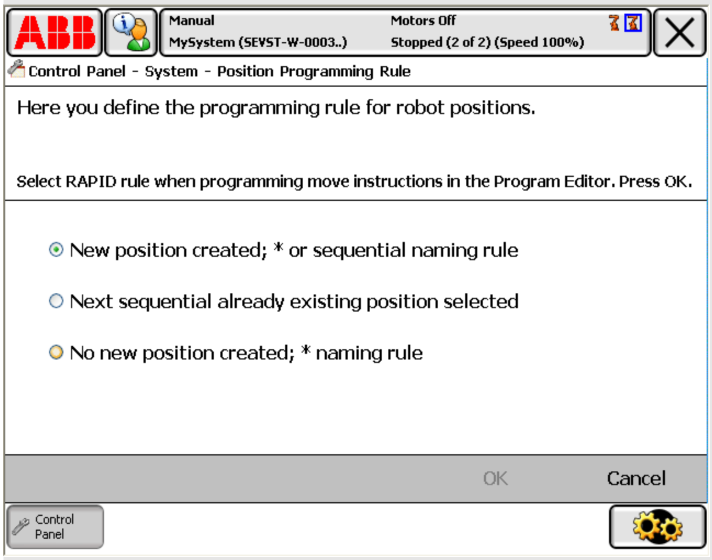
12.1.6. Defining position programming rule

Overview

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

How to define position programming rule

This procedure details how to define a naming rule for new robot positions.

	Action
1.	On the **ABB** menu, tap **Control Panel** and then **FlexPendant**.
2.	Tap **Position Programming Rule**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - System - Position Programming Rule Here you define the programming rule for robot positions. Select RAPID rule when programming move instructions in the Program Editor. Press OK. ● New position created; * or sequential naming rule ○ Next sequential already existing position selected ○ No new position created; * naming rule OK Cancel Control Panel en0500002415
3.	Tap to select the preferred position programming rule.
4.	Tap **OK**.

Position programming rules

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

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

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

Continues on next page

New position created; * or sequential naming rule

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

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

Next sequential already existing position selected

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

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

No new position created; * naming rule

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

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

12.1.7. Defining which tasks should be selectable in the tasks panel

Tasks panel

The tasks panel is found in the Quickset menu. See *Quickset menu, Tasks on page 214*.

How to set which tasks to show

This procedure details how to set which tasks should be selectable in the tasks panel in the Quickset menu.

	Action
1.	On the **ABB** menu, tap **Control Panel** and then tap **FlexPendant**.
2.	Tap **Task Panel Settings**.
3.	Select **Only Normal tasks** or **All tasks**. **All tasks** makes all tasks with trustlevel set to `No safety` selectable in manual mode.
4.	Tap **OK**.

12.2 Basic settings

12.2.1. Changing brightness and contrast

Appearance options

This section describes the Appearance menu, where you can adjust the screen’s brightness and contrast.

Changing brightness and contrast

This procedure describes how to change brightness and contrast of the screen.

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

NOTE!

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

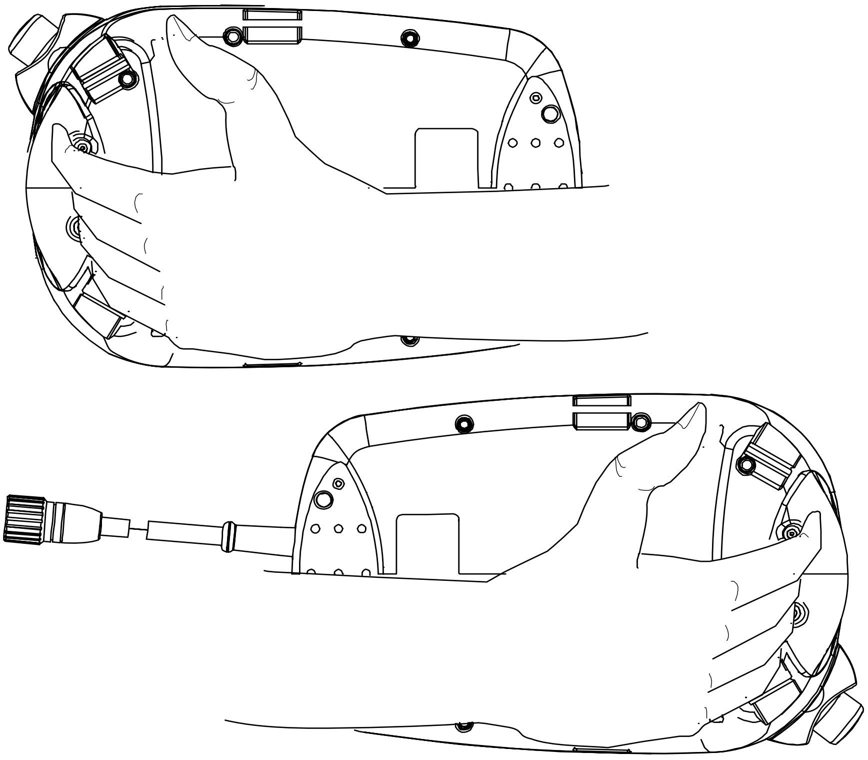
12.2.2. Adapting the FlexPendant for left-handed users

Overview

The device is usually operated while being supported by the left hand. A left-hander, however, normally prefers to use his left hand for using the touch screen. however, can easily rotate the display through 180 degrees and use his right hand to support the device. The FlexPendant is set to suit right-handers on delivery, but can easily be adapted to suit the needs of the left-handed.

Illustration

The FlexPendant operated by a right-handed person at the top and by a left-handed person at the bottom. Especially note the location of the emergency button when the display is rotated through 180 degrees.



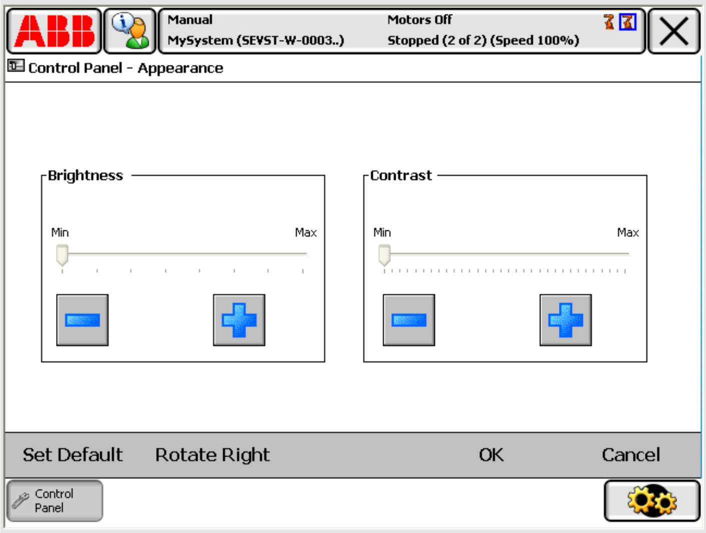
en0400000913

Procedure

This section details how to adapt the FlexPendant to suit a left-handed user.

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

Continues on next page

	Action
3.	Tap **Rotate right**.  en0400000915
4.	Rotate the FlexPendant and move it to the other hand.

What is affected?

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

Setting	Effect	Information
Jogging directions	The joystick directions are adjusted automatically.	The illustrations of jogging directions in the jogging menu are adjusted automatically.
Hardware buttons and programmable keys	Start, Stop, Forward, and Backward buttons **do not** change place with programmable keys.	See buttons A-G in the illustration *Hard buttons on page 41*.
Emergency stop	No effect.	Differently located, at the bottom instead of at the top.
Enabling device	No effect	

12.2.3. Changing date and time

Changing date and time

This procedure details how to set the controller clock.

	Action
1.	On the **ABB** menu, tap **Control Panel**.
2.	Tap **Date and Time**. The current date and time is displayed.
3.	Tap the appropriate **Plus** or **Minus** button to change the date or time.
4.	Tap **OK** to use the time and date settings.

NOTE!

The date and time is always displayed according to ISO standard, that is, year-month-day and hour:minute, the time using 24-hour format.

12.2.4. Configuring Most Common I/O

Most Common I/O

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

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

Most Common I/O can also be configured using system parameters in the topic *Man-machine Communication*. However, sorting the list can only be done by using the function under the Control Panel. See section *Configuring system parameters on page 283*.

Configuring Most Common I/O

This section describes how to configure the list Most Common I/O.

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

12.2.5. Changing language

Languages

This procedure details how to change between the currently installed languages. The individual FlexPendant supports up to three languages, selected before the installation of the system to the robot controller.

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

Changing language

This section describes how to change language on the FlexPendant.

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

12.2.6. Changing programmable keys

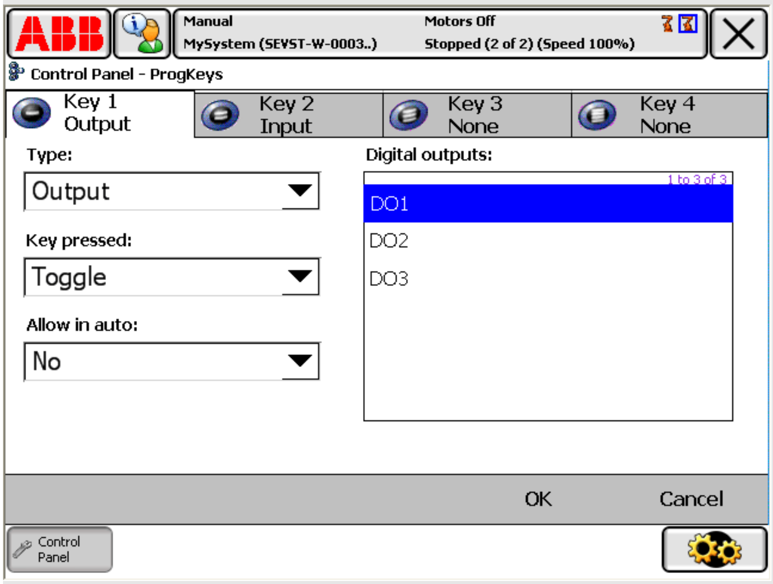
Overview

Programmable keys are four hardware buttons on the FlexPendant that may be used for dedicated, specific functions set by the user. See *Hard buttons on page 41*.

The keys can be programmed to simplify programming or testing of programs. They can also be used to activate menus on the FlexPendant.

Change programmable keys

This section describes how to set the programmable keys.

	Action
1.	On the **ABB** menu, tap **Control Panel**.
2.	Tap **ProgKeys**.  Manual MySystem (SEVST-W-0003..) Motors Off Stopped (2 of 2) (Speed 100%) Control Panel - ProgKeys Key 1 Output Key 2 Input Key 3 None Key 4 None Type: Output Key pressed: Toggle Allow in auto: No Digital outputs: 1 to 3 of 3 DO1 DO2 DO3 en0400001154
3.	Select key to set, Key 1-4 in the upper selection list.
4.	Tap the **Type** menu to select type of action: • None • Input • Output • System
5.	If Type **Input** is selected. • Tap to select one of the digital inputs from the list. • Tap the **Allow in auto** menu to select if the function is also allowed in automatic operating mode. **Note!** A digital input signal cannot be set by using the programmable keys. Its value can only be pulsed from high to low and will result in an event which can be connected to a RAPID event routine.

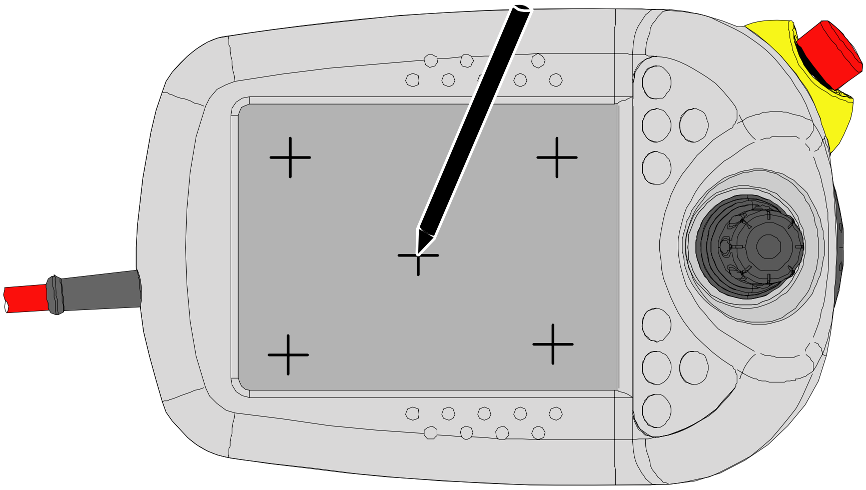
Continues on next page

	Action
6.	If Type **Output** is selected. • Tap to select one of the digital outputs from the list • Tap the **Key pressed** menu to define how the signal should behave when the key is pressed. • Tap the Allow in auto menu to select if the function is also allowed in automatic operating mode Key pressed functions: • Toggle - switches signal value from 0 to 1 or vice versa • Set to 1 - sets the signal to 1 • Set to 0 - sets the signal to 0 • Press/Release - sets signal value to 1 while key is pressed (note that an inverted signal will be set to 0) • Pulse - the signal value pulses once
7.	If Type **System** is selected. • Tap the **Key pressed** menu to select Move PP to main • Tap the **Allow in auto** menu to select if the function is also allowed in automatic operating mode
8.	Set the other keys as described in steps 3 to 7 above.
9.	Tap **OK** to save the settings.

12.2.7. Calibrating the touch screen

Recalibration

This section describes how to recalibrate the touch screen. The touch screen is calibrated on delivery and normally never needs to be recalibrated.



en0400000974

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

About the touch calibration function

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

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

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

13 Descriptions of terms and concepts

13.1. About this chapter

Overview

This chapter provides definitions and explanations of important concepts and words used in this manual.

Note that there may also be additional information in any of the chapters dealing with the feature at hand.

13.2. What is the robot system?

Description

The concept *robot system* comprises the manipulator(s), controller(s), and all equipment controlled by the controller (tool, sensors, etc.). It includes all hardware as well as software required to operate the robot. Application specific hardware and software, such as spot welding equipment, is not included in the term.

13.3. What are mechanical units, manipulators and positioners?

Mechanical unit

A *mechanical unit* can be jogged. It can either be a robot, a single additional axis, such as a motor, or a set of additional axes, for instance a two axis positioner or a non-ABB robot.

Manipulator

Manipulator is a generic term for mechanical units used to move objects, tools, etc. The term *manipulator* includes robots as well as positioners.

Robot

A *robot* is a mechanical unit with TCP. A robot can be programmed to move to a position specified in Cartesian coordinates (x, y and z) of the TCP and in tool orientation.

Positioner

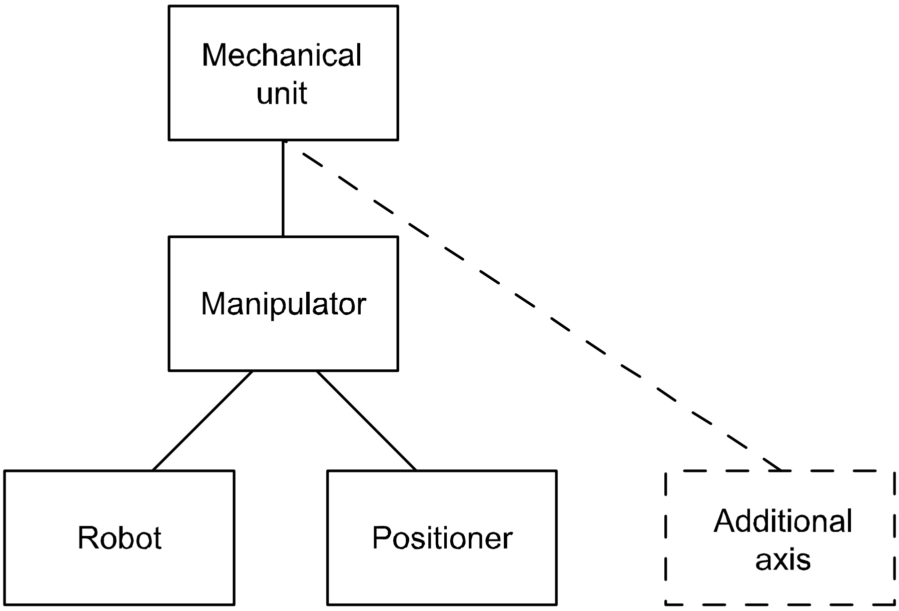
A *positioner* is a mechanical unit used to move a work object. It may have one or several axes, however normally no more than 3 axes. A positioner normally does not have a TCP.

Additional axis

The robot controller can control additional axes besides the robot axes. These mechanical units can be jogged and coordinated with the movements of the robot.

Illustration

The illustration depicts the relation between the concepts: mechanical unit, manipulator, robot, positioner and additional axis.



en0400000940

13.4. What is a tool?

Tool

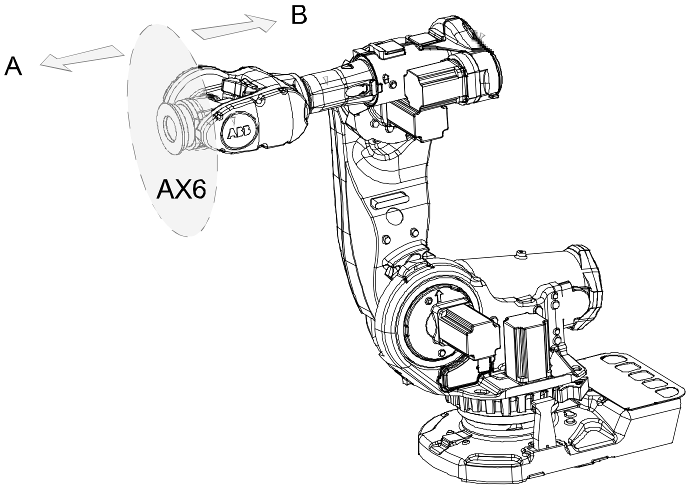
A tool is an object that can be mounted directly or indirectly on the robot turning disk or fitted in a fixed position within the robot working range.

A fixture (jig) is not a tool.

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

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

Illustration



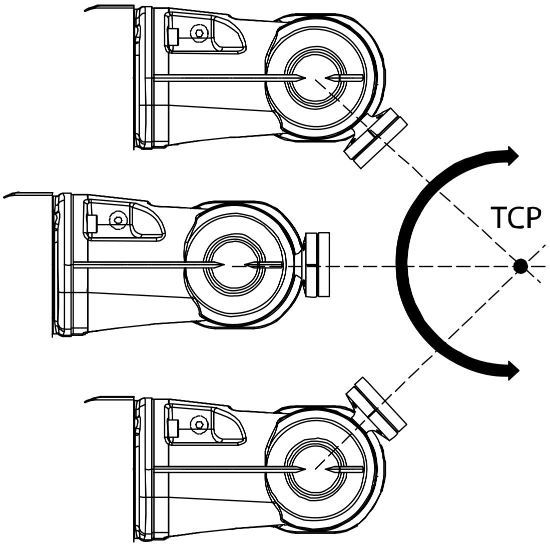
en0400000803

A	Tool side
B	Robot side

13.5. What is the tool center point?

Illustration

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



xx0300000604

Description

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

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

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

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

Moving TCP

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

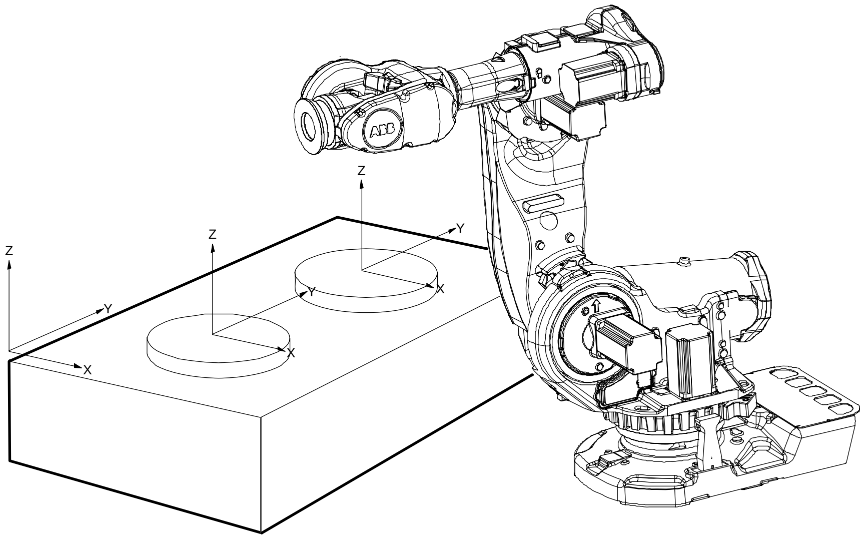
A typical moving TCP may be defined in relation to, e.g. the tip of a arc welding gun, the center of a spot welding gun or the end of a grading tool.

Stationary TCP

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

13.6. What is a work object?

Illustration



en0400000819

Description

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

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

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

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

13.7. What is a coordinate system?

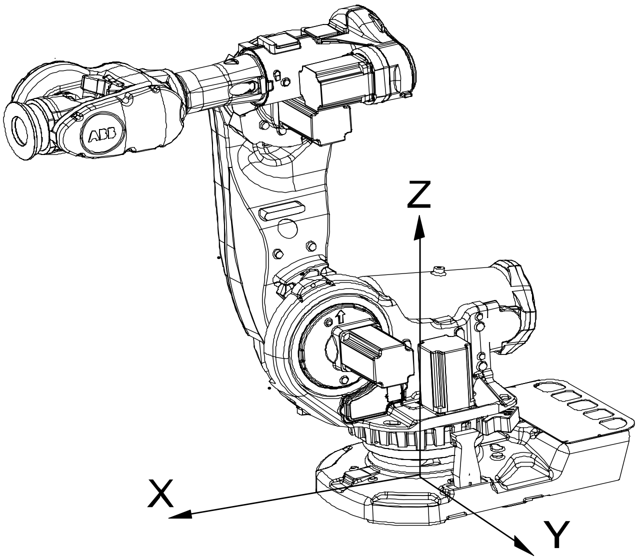
Overview

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. See *The base coordinate system on page 323*for more information.
- The *work object coordinate system* is related to the work piece and is often the best one for programming the robot. See *The work object coordinate system on page 325*for more information.
- The t*ool coordinate system* defines the position of the tool the robot uses when reaching the programmed targets. See *The tool coordinate system on page 326*for more information.
- 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. See *The world coordinate system on page 324*for more information.
- The u*ser coordinate system* is useful for representing equipment that holds other coordinate systems, like work objects. See *The user coordinate system on page 327* for more information.

The base coordinate system



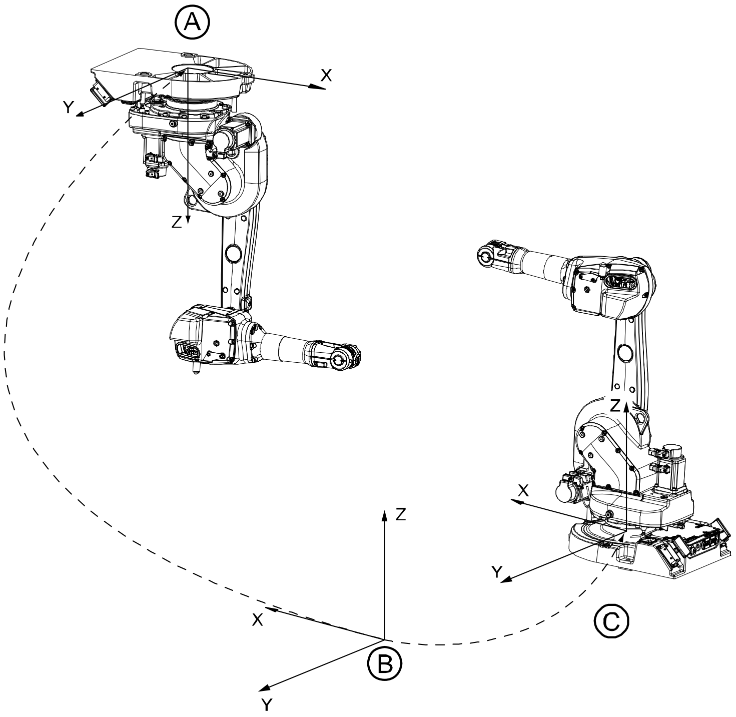
xx0300000495

Continues on next page

The base coordinate system has its zero point in the base of the robot, which makes movements predictable for fixed mounted robots. It is therefore useful for jogging a robot from one position to another. For programming a robot, other coordinate systems, like the work object coordinate system are often better choices. See *The work object coordinate system on page 325*for more information.

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

The world coordinate system



en0300000496

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

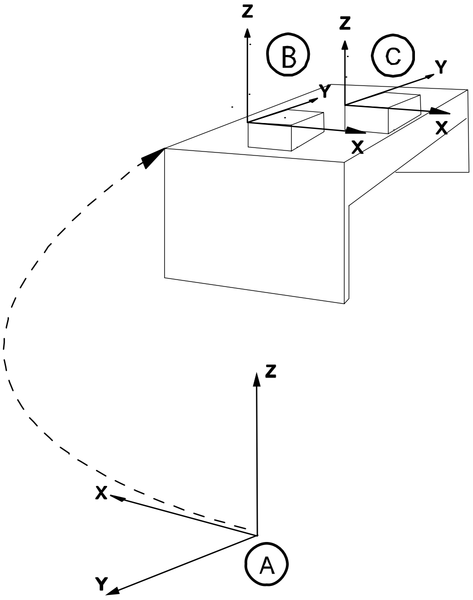
The world coordinate system has its zero point on a fixed position in the cell or station. This makes it useful for handling several robots or robots moved by external axes.

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

Continues on next page

Continued

The work object coordinate system



xx0600002738

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

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

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

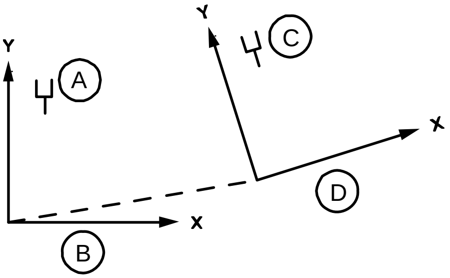
A robot can have several work object coordinate systems, either for representing different work pieces or several copies of the same work piece at different locations.

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

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

Continues on next page

The displacement coordinate system



en0400001227

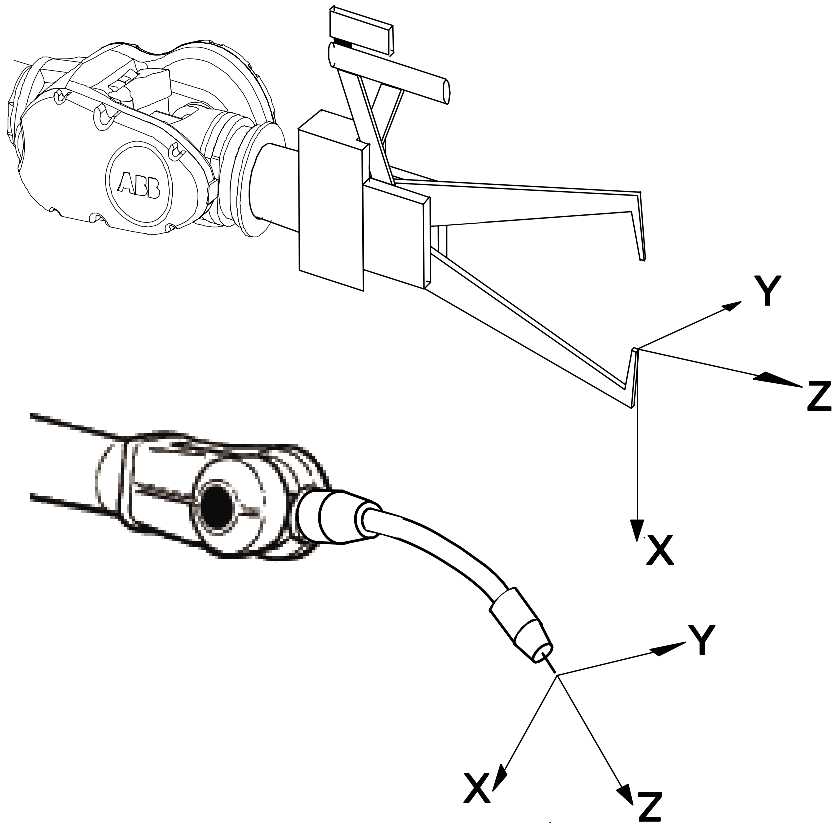
A	Original position
B	Object coordinate system
C	New position
D	Displacement coordinate system

Sometimes, the same path is to be performed at several places on the same object, or on several work pieces located next to each other. To avoid having to reprogram all positions each time a displacement coordinate system can be defined.

This coordinate system can also be used in conjunction with searches, to compensate for differences in the positions of the individual parts.

The displacement coordinate system is defined based on the object coordinate system.

The tool coordinate system



xx0300000506

Continues on next page

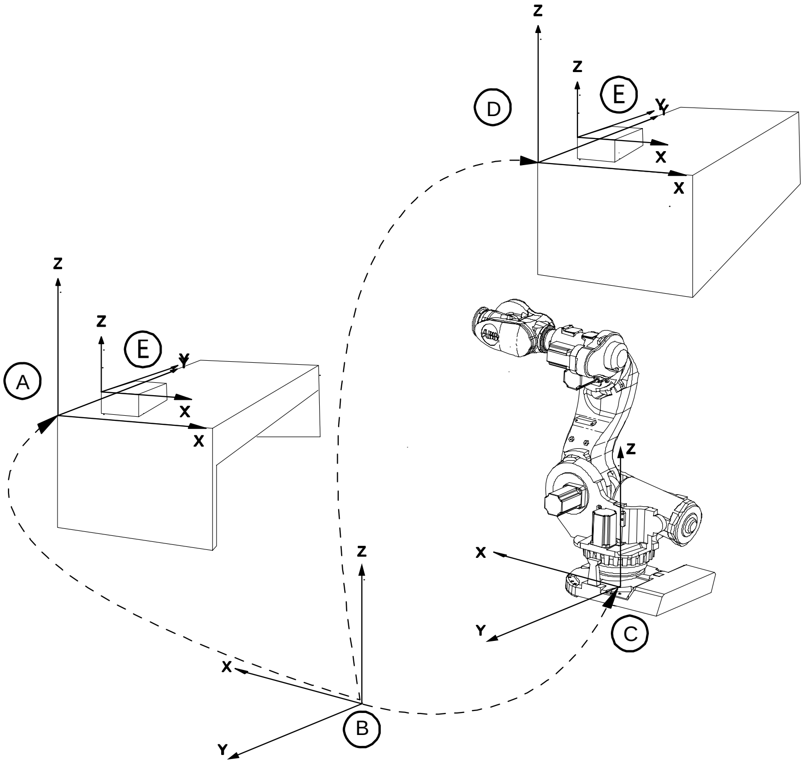
The tool coordinate system has its zero position at the center point of the tool. It thereby defines the position and orientation of the tool. The tool coordinate system is often abbreviated TCPF (Tool Center Point Frame) and the center of the tool coordinate system is abbreviated TCP (Tool Center Point).

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

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

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

The user coordinate system



en0300000497

A	User coordinate system
B	World coordinate system
C	Work object coordinate system
D	Moved user coordinate system
E	Work object coordinate system, moved with user coordinate system

The user coordinate system can be used for representing equipment like fixtures, workbenches. This gives an extra level in the chain of related coordinate systems, which might be useful for handling equipment that hold work objects or other coordinate systems.

13.8. What is a RAPID application?

Purpose

A RAPID application contains a sequence of instructions that controls the robot so that it may perform the operations it is intended for.

Contents of the RAPID application

An application is written using a particular vocabulary and syntax called *RAPID programming language*.

The RAPID programming language is in English and contains instructions to enable the robot to move, setting outputs and reading inputs. It also contains instructions for making decisions, to repeat other instructions, to structure the program, to communicate with the system operator and more.

Structure of the RAPID application

The structure of a RAPID application is shown in section *The structure of a RAPID application on page 134*.

How is an application stored?

An application you work with or run must be loaded in the controller's program memory. This procedure is called to *Load* the application.

You *Save* applications on the controller's mass memory unit or other disk memory to keep them safe when you want to work on another application.

See also *What is “the memory”? on page 260* and *Setting default paths on page 299*.

13.9. What is mirroring?

Description

Mirroring creates a copy of a program, module, or routine in a specific mirror plane. The mirror function can be applied to any program, module, or routine.

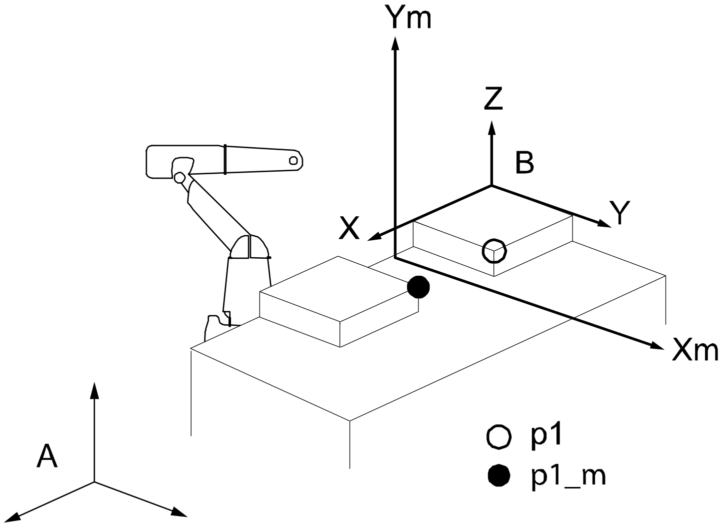
Mirroring can be performed in two different ways:

- Default against the base frame coordinate system. The mirror operation will be performed across the xz-plane in the base frame coordinate system. All positions and work object frames that are used in an instruction in the selected program, module or routine are mirrored. The position orientation axes x and z will be mirrored.
- Advanced against a specific mirror frame. The mirror operation will be performed across the xy-plane in a specified work object frame, mirror frame. All positions in the selected program, module or routine are mirrored. If the work object argument in an instruction is another work object than specified in the mirror dialog, the work object in the instruction is used in the mirror operation. It is also possible to specify which axis in the position orientation that will be mirrored, x and z or y and z.

The following descriptions of mirroring describes advanced mirroring.

Mirror plane

The mirror function will mirror all positions (robtargets) in the mirror plane, i.e. the mirrored position will be located symmetrically on the other side of the plane, relative to the original position. The mirror plane is always the xy-plane of an object frame, used for mirroring. This object frame is defined by a work object data, e.g. with the name MIRROR_FRAME.



xx0600002815

Ym, Xm	Mirror plane
A	World frame
B	Work object frame
p1	Original point
p1_m	Mirrored point

Continues on next page

Mirroring routines

Mirroring creates a copy of a routine with all positions (robtargets) mirrored in a specific mirror plane. In general, all data of the type robtarget used in the routine, both local and global, will be mirrored. It makes no difference whether the robtarget data is declared as a constant (which it should be), as a persistent, or as an ordinary variable. Any other data, e.g. of type pos, pose, orient, etc., will not be mirrored.

Mirroring data only affects the initialization value, i.e. any current value will be ignored. This means that if a robtarget variable has been defined without an init value, this variable will **not** be mirrored.

The new, mirrored routine will be given a new name (a default name is proposed). All stored data of type robtarget, used in the routine, will be mirrored and stored with a new name (the old name ending with "_m"). All immediate robtarget data, shown with an "*", in movement instructions will also be mirrored.

Mirrored values and arguments

When mirroring a routine, the new routine is scanned for any local robtarget data, declared inside the routine with an init value. All init values of such data are mirrored. Then the new routine is scanned for statements with one or more arguments of type robtarget.

When such a statement is found, the following actions will take place:

- If the argument is programmed with a reference to a local variable or a constant, this argument will be ignored, since it has already been mirrored as described above.
- If the argument is programmed with an immediate robtarget data, shown with an asterisk" *", then this value will be mirrored directly.
- If the argument is programmed with a reference to a global variable, persistent or a constant, defined outside the routine with an init value, then a duplicate is created and stored in the module with a new name (the old name ending with "_m"). The init value of this new data is mirrored, and then the argument in the statement is changed to the new name. This means that the module data list will expand with a number of new mirrored robtarget data.

Error handlers or backward handlers in the routine are not mirrored.

Work object frame

All positions which are to be mirrored are related to a specific work object frame (B in figure above). This means that the coordinates of the robtarget data are expressed relative to this work object frame. Furthermore, the mirrored position will be related to the same work object frame.

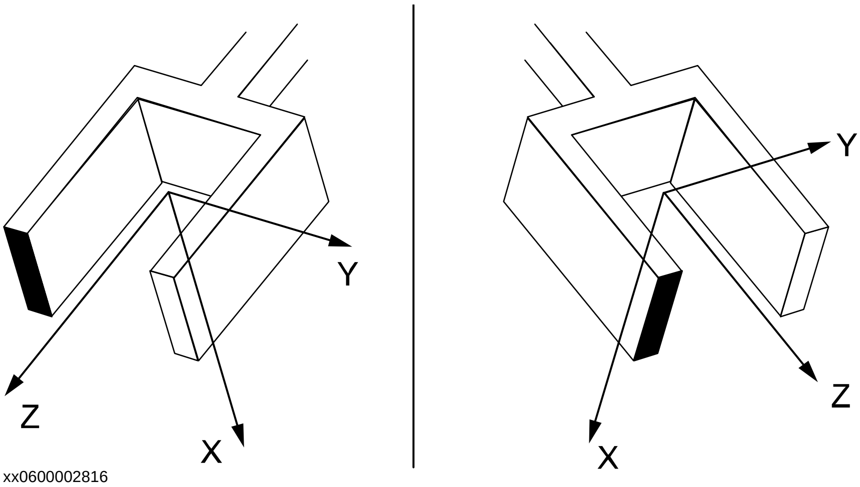
Before mirroring, this specific work object must be stated. This work object will be used as the reference frame for all variables that are to be mirrored.

Make sure to state the same work object as was originally used when defining the robtarget data, and which was used as a parameter in the movement instructions. If no work object was used, the wobj0 should be stated.

Continues on next page

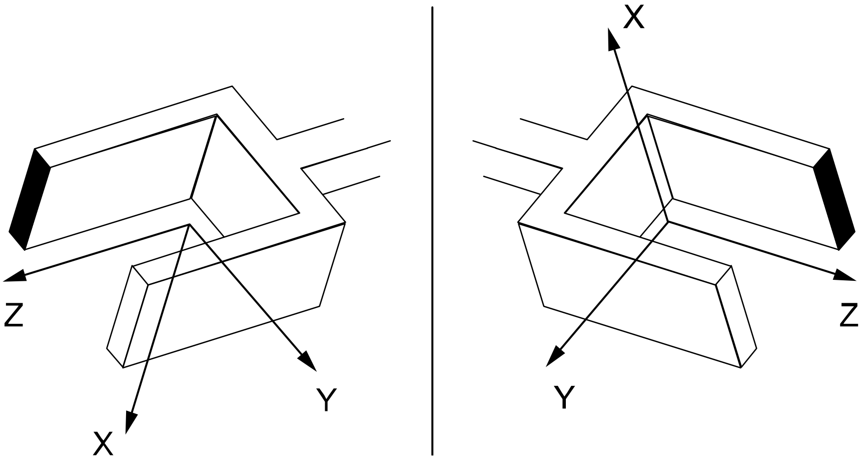
Orientation of mirrored positions

The orientation of the robtarget position is also mirrored. This mirroring of the orientation can be done in two different ways, where either the x and z axes are mirrored or the y and z axes. The method used, x or y axis (the z axis is always mirrored), is dependent on the tool used and how the tool coordinate system is defined.



xx0600002816

Mirroring of x and z axes.



xx0600002817

Mirroring of y and z axes.

Arm configurations

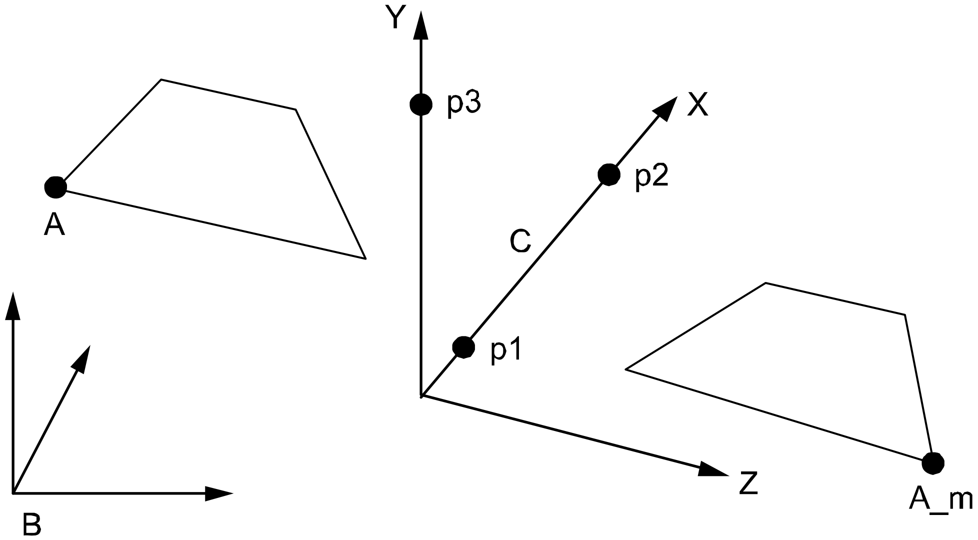
The arm configuration will not be mirrored, which means that after mirroring, it has to be carefully checked by executing the path in manual mode. If the arm configuration has to be changed, this must be done manually and the position corrected with a modpos command.

Continues on next page

Example 1: Mirroring with one robot

A mirrored copy of the routine `org` is to be created and stored with the name `mir`. All positions are related to the work object, wobj3. The mirror plane is known from three positions in the plane, p1, p2, and p3.

An original position in `org`, A, is mirrored to A_m.



xx0600002818

A	Original position
A_m	Mirrored position
B	Object frame wobj3
C	Mirror plane

To perform this mirroring, the mirror frame must first be defined. To do this, create a new work object and name it (e.g. `mirror`). Then, use the three points, p1 to p3, to define the object coordinate system by using the robot. This procedure is described in *Defining the work object coordinate system on page 156*, in the *Operating manual - IRC5 with FlexPendant*.

After this, the routine, `org`, can be mirrored using wobj3 and mirror as input data.

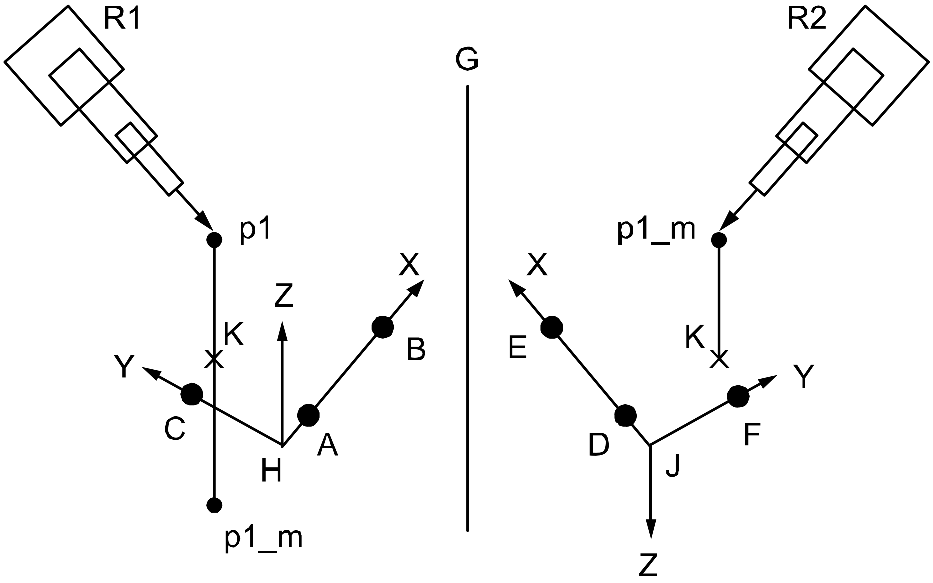
Continues on next page

Example 2: Mirroring with two robots

The routine `org` was created on one robot and should be mirrored and used on another robot. Suppose that a spot welding robot, robot 1, is used for the left side of a car body. When the program for the left side is done, it should be mirrored and used again for the right side by robot 2.

The original program, `org`, is programmed relative to a work object, wobj1, which is defined with the help of three points, A, B and C on the left side of the car body. The mirrored program, `mir`, is to be related to a corresponding work object, wobj1, defined by the corresponding points D, E and F on the right side of the car body. Wobj1 for robot 2 is defined with robot 2.

Note that since the points D, E, F are mirrored images of points A, B, and C, the wobj1 for robot 2 will also be mirrored. One of the consequences of this is that the z-axis will point downwards.



xx0600002819

R1	Robot 1
R2	Robot 2
G	Virtual mirror plane
H	wobj1 = mirror frame
J	wobj1 for robot 2
K	Projection of p1 in xy-plane
p1	Original position
p1_m	Mirrored position

After the work object, wobj1, has been defined, all programming is done in this frame. Then the program is mirrored using the same wobj1 frame as the mirroring frame. A position, p1, will be mirrored to the new position p1_m.

After this, the mirrored program is moved to robot 2, using the work object wobj1, as described above. This means that the mirrored position, p1_m, will be “turned up” as if it were mirrored in a “virtual” mirror plane between the two robots.

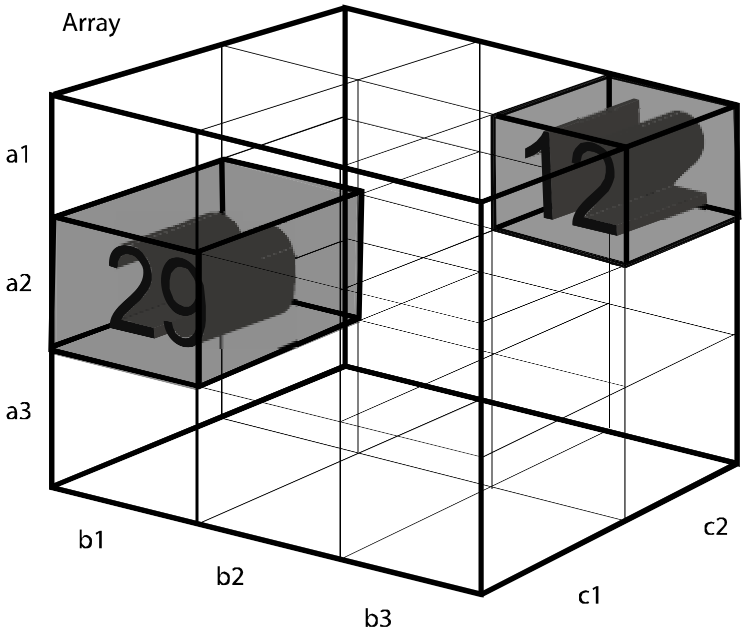
13.10. What is a data array?

Overview

A data array is a special type of variable: a regular variable may contain one data value, but an array may contain several.

It may be described as a table, which may have one or more dimensions. This table may be populated with data (e.g. numerical values, character strings, or variables) to be used during programming or operation of the robot system.

An example of a three dimensional array is shown below:



en0400001006

This array, called "Array" is defined by its three dimensions a, b and c. Dimension a has three rows, b has three rows (columns) and c has two rows. The array and its contents may be written as `Array {a, b, c}`.

Example 1: `Array {2, 1, 1}=29`

Example 2: `Array {1, 3, 2}=12`

3HAC16590-1, Revision K, en

ABB AB
Robotics Products
S-721 68 VÄSTERÅS
SWEDEN
Telephone: +46 (0) 21 344000
Telefax: +46 (0) 21 132592