

# Operating manual RobotStudio

Trace back information:  
Workspace R14-1 version a5  
Checked in 2014-04-07  
Skribenta version 4.0.378

# **Operating manual**

## **RobotStudio**

5.61

Document ID: 3HAC032104-001

Revision: M

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

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

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

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

Additional copies of this manual may be obtained from ABB.

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

© Copyright 2008-2014 ABB. All rights reserved.

ABB AB  
Robotics Products  
Se-721 68 Västerås  
Sweden

# Table of contents

Overview of this manual .....	11
Product documentation, IRC5 .....	17
Safety .....	19
<b>1 Introduction to RobotStudio .....</b>	<b>21</b>
1.1 What is RobotStudio .....	21
1.2 Terms and concepts .....	22
1.2.1 Hardware concepts .....	22
1.2.2 RobotWare concepts .....	24
1.2.3 RAPID concepts .....	26
1.2.4 Concepts of programming .....	27
1.2.5 Targets and paths .....	28
1.2.6 Coordinate systems .....	29
1.2.7 Robot axis configurations .....	35
1.2.8 Libraries, geometries and CAD files .....	37
1.3 Installing and licensing RobotStudio .....	40
1.4 User interface .....	48
1.4.1 Ribbon, tabs and groups .....	48
1.4.2 Layout browser .....	49
1.4.3 The Paths & Targets browser .....	50
1.4.4 The Modeling browser .....	52
1.4.5 The Controller browser .....	53
1.4.6 Files browser .....	55
1.4.7 Add-Ins browser .....	56
1.4.8 The Output window .....	57
1.4.9 The Controller Status window .....	58
1.4.10 The Operator Window .....	60
1.4.11 The Documents window .....	62
1.4.12 Using a mouse .....	69
1.4.13 Selecting an item .....	70
1.4.14 Attaching and detaching objects .....	71
1.4.15 Keyboard shortcuts .....	72
<b>2 Building stations .....</b>	<b>75</b>
2.1 Workflow of building a station .....	75
2.2 Conveyor tracking station with two robots .....	77
2.2.1 Two robot systems in same task frame position .....	77
2.2.2 Two robot systems in different task frame positions .....	79
2.3 Creating a system with external axes automatically .....	81
2.4 Manually setting up system with track motion .....	83
2.4.1 Track motion of type RTT or IRBTx003 .....	83
2.4.2 Track motion of type IRBTx004 .....	84
2.5 Virtual Controller .....	85
2.5.1 Starting a VC .....	85
2.5.2 Restarting a VC .....	86
2.6 Station components .....	87
2.6.1 Importing a station component .....	87
2.6.2 Converting CAD formats .....	89
2.6.3 Troubleshooting and optimizing geometries .....	90
2.7 Modeling .....	92
2.7.1 Objects .....	92
2.7.2 Mechanisms .....	94
2.7.3 Tools and tooldata .....	95
2.7.4 Setting the local origin of an object .....	96
2.8 Placement .....	97
2.8.1 Placing objects .....	97

## Table of contents

---

2.8.2	Placing external axes .....	98
2.8.3	Placing robots .....	100
<b>3</b>	<b>Programming robots .....</b>	<b>103</b>
3.1	Workflow for programming a robot .....	103
3.2	Workobjects .....	104
3.3	Jogging mechanisms .....	105
3.4	Targets .....	106
3.5	Paths .....	108
3.6	Orientations .....	111
3.7	RAPID Instructions .....	115
3.8	Testing positions and motions .....	122
3.9	Programming MultiMove systems .....	124
3.9.1	About programming MultiMove .....	124
3.9.2	Setting up the MultiMove .....	126
3.9.3	Testing the MultiMove .....	127
3.9.4	Tuning the motion behavior .....	128
3.9.5	Creating paths .....	130
3.10	Programming external axes .....	131
3.11	Loading and saving programs and modules .....	133
3.12	Synchronization .....	134
<b>4</b>	<b>Simulating programs .....</b>	<b>135</b>
4.1	Simulation Overview .....	135
4.2	Detecting collisions .....	137
4.3	Creating an event .....	140
4.4	Simulating I/O signals .....	141
4.5	Enabling simulation monitoring .....	142
4.6	Measuring simulation time .....	143
<b>5</b>	<b>Deploying and distributing .....</b>	<b>145</b>
5.1	Copying programs .....	145
5.2	Pack & Go / Unpack & Work .....	146
5.3	Screen Capture .....	147
<b>6</b>	<b>Working online .....</b>	<b>149</b>
6.1	Connecting a PC to the controller .....	149
6.2	Network settings .....	153
6.3	User Authorization .....	156
6.4	The System Builder .....	158
6.4.1	System Builder Overview .....	158
6.4.2	Viewing system properties .....	160
6.4.3	Building a new system .....	161
6.4.4	Modifying a system .....	165
6.4.5	Copying a system .....	169
6.4.6	Creating a system from backup .....	170
6.4.7	Downloading a system to a controller .....	171
6.4.8	Creating boot media .....	172
6.4.9	Examples using the System Builder when offline .....	173
6.4.9.1	A MultiMove system with two coordinated robots .....	173
6.4.9.2	A system with support for one robot and one positioner external axis ....	175
6.4.9.3	Options settings for systems with positioners .....	177
6.5	Handle I/O .....	178
6.6	Configure systems .....	179
6.7	Handle events .....	185
<b>7</b>	<b>File tab .....</b>	<b>189</b>
7.1	Overview .....	189

7.2	New .....	190
7.3	Share .....	191
7.3.1	Pack and Go .....	191
7.3.2	Unpack and Work .....	192
7.3.3	Station Viewer .....	193
7.4	Options .....	195
<b>8</b>	<b>Home tab</b> .....	<b>203</b>
8.1	Overview .....	203
8.2	ABB Library .....	204
8.3	Import Library .....	205
8.4	Robot System .....	206
8.4.1	Robot System .....	206
8.4.2	External Axis Wizard .....	209
8.5	Import Geometry .....	212
8.6	Frame .....	213
8.6.1	Frame .....	213
8.6.2	Frame from Three Points .....	214
8.7	Workobject .....	216
8.8	Tooldata .....	217
8.9	Target .....	218
8.9.1	Teach Target .....	218
8.9.2	Create Target .....	219
8.9.3	Create Jointtarget .....	221
8.9.4	Create Targets on Edge .....	222
8.10	Empty Path .....	224
8.11	AutoPath .....	225
8.12	MultiMove .....	227
8.13	Teach Instruction .....	235
8.14	Move Instruction .....	236
8.15	Action Instruction .....	237
8.16	Instruction Template Manager .....	238
8.17	Settings .....	241
8.17.1	Task .....	241
8.17.2	Workobject .....	242
8.17.3	Tool .....	243
8.18	The Freehand Group .....	244
8.18.1	Rotate .....	245
8.18.2	Jog Joint .....	246
8.18.3	Jog Linear .....	247
8.18.4	Jog Reorient .....	248
8.18.5	MultiRobot Jog .....	249
8.19	Graphics Tools .....	250
8.19.1	View Tab .....	251
8.19.2	Edit Tab .....	258
<b>9</b>	<b>Modeling tab</b> .....	<b>261</b>
9.1	Overview .....	261
9.2	Component Group .....	262
9.3	Empty Part .....	263
9.4	Smart Component .....	264
9.4.1	Smart Component .....	264
9.4.2	Smart Component Editor .....	265
9.4.3	The Compose tab .....	266
9.4.4	The Properties and Bindings tab .....	269
9.4.5	The Signals and Connections tab .....	272
9.4.6	The Design tab .....	275
9.4.7	Basic Smart Components .....	276
9.4.8	Property Editor .....	293

## Table of contents

---

9.4.9	The Simulation Watch window .....	294
9.5	Tags .....	296
9.6	Solid .....	297
9.7	Surface .....	301
9.8	Curve .....	303
9.9	Border .....	308
9.10	Intersect .....	310
9.11	Subtract .....	311
9.12	Union .....	312
9.13	Extrude Surface or Curve .....	313
9.14	Line from Normal .....	315
9.15	The Measure Group .....	316
9.16	Create Mechanism .....	317
9.17	Create Tool .....	324
<b>10</b>	<b>Simulation tab</b> .....	<b>327</b>
10.1	Overview .....	327
10.2	Create Collision Set .....	328
10.3	Simulation Setup .....	329
10.4	Event Manager .....	332
10.5	Station Logic .....	338
10.6	Activate Mechanical Units .....	339
10.7	Simulation Control .....	340
10.8	I/O Simulator .....	341
10.9	Monitor .....	343
10.10	Stopwatch .....	344
10.11	Signal Analyzer .....	345
10.11.1	Signal Analyzer for both real and virtual controllers .....	345
10.11.2	Signal Setup .....	346
10.11.3	Layout and usage .....	349
10.11.4	History .....	352
10.12	Record Movie .....	353
10.13	Conveyor Tracking Mechanism .....	354
10.13.1	Conveyor Tracking .....	354
10.13.2	Conveyor Simulation .....	355
<b>11</b>	<b>Controller tab</b> .....	<b>357</b>
11.1	Real and virtual controllers .....	357
11.2	Features for both virtual and real controllers .....	358
11.2.1	Add Controller .....	358
11.2.2	Events .....	360
11.2.3	Inputs / Outputs .....	361
11.2.4	ScreenMaker .....	363
11.2.5	Restart a controller .....	365
11.2.6	Back up a system .....	367
11.2.7	Restore a system .....	369
11.2.8	System Builder .....	371
11.2.9	Configuration editor .....	372
11.2.10	Load Parameters .....	374
11.2.11	Save Parameters .....	375
11.2.12	Transfer .....	376
11.2.13	Signal Analyzer Online .....	379
11.2.14	Safety Configuration .....	381
11.3	Features for real controllers .....	382
11.3.1	Request Write Access .....	382
11.3.2	Release Write Access .....	383
11.3.3	Authenticate .....	384
11.3.4	File transfer .....	385
11.3.5	FlexPendant Viewer .....	387

11.3.6	Import Options .....	388
11.3.7	Properties .....	389
11.3.8	Go Offline .....	392
11.3.9	Online Monitor .....	393
11.3.10	User Accounts .....	395
11.3.11	UAS Grant Viewer .....	400
11.3.12	Integrated Vision .....	404
11.4	Features for virtual controllers .....	405
11.4.1	Virtual FlexPendant .....	405
11.4.2	Control Panel .....	406
11.4.3	Shutdown .....	407
11.4.4	Set Task Frames .....	408
11.4.5	Edit System .....	409
11.4.6	Encoder Unit .....	411
<b>12</b>	<b>RAPID tab</b> .....	<b>413</b>
12.1	Overview of the RAPID tab .....	413
12.2	Synchronize to Station .....	414
12.3	Synchronize to VC .....	415
12.4	Edit RAPID code .....	416
12.5	Find and replace RAPID code .....	421
12.6	Manage RAPID modules .....	423
12.7	Edit RAPID data .....	425
12.8	Manage RAPID files and backups .....	426
12.9	Manage RAPID code on the controller .....	427
12.9.1	Manage RAPID programs .....	427
12.9.2	RAPID Tasks .....	428
12.9.3	Run Mode .....	431
12.9.4	Adjust Robtargets .....	432
12.10	Test and debug .....	435
12.10.1	Commands for testing and debugging .....	435
12.10.2	Using the Program Pointer .....	436
12.10.3	Using the RAPID Profiler .....	438
12.11	RAPID Watch window .....	440
12.12	Examples of using the RAPID editor .....	441
<b>13</b>	<b>Add-Ins tab</b> .....	<b>443</b>
<b>14</b>	<b>Context menus</b> .....	<b>447</b>
14.1	Add to Path .....	447
14.2	Align Frame Orientation .....	448
14.3	Align Target Orientation .....	449
14.4	Attach to .....	450
14.5	Configurations .....	451
14.6	Check Reachability .....	453
14.7	Configurations .....	454
14.8	Convert Frame to Workobject .....	455
14.9	Convert to Move Circular .....	456
14.10	Copy / Apply Orientation .....	457
14.11	Detach .....	458
14.12	Execute Move Instruction .....	459
14.13	External Axis Interpolation .....	460
14.14	Graphic Appearance .....	461
14.15	Go to Visualization and Go to Declaration .....	463
14.16	Interpolate Path .....	464
14.17	Invert .....	465
14.18	Jump to Target .....	466
14.19	Linked Geometry .....	467
14.20	Modify Library Component .....	468

## Table of contents

---

14.21	Mechanism Joint Jog .....	469
14.22	Mechanism Linear Jog .....	471
14.23	Mirror Path .....	472
14.24	Mirror .....	473
14.25	Modify Curve .....	474
14.26	Modify External Axis .....	476
14.27	Modify Instruction .....	477
14.28	Modify Mechanism .....	478
14.29	Modify Tooldata .....	479
14.30	Modify Workobject .....	480
14.31	Move Along Path .....	481
14.32	Move to Pose .....	482
14.33	Offset Position .....	483
14.34	Place .....	484
14.35	Protected Smart Component .....	486
14.36	Remove Unused Targets .....	487
14.37	Rename Targets .....	488
14.38	Reverse Path .....	489
14.39	Rotate .....	490
14.40	Rotate Path .....	491
14.41	Set Local Origin .....	492
14.42	Set Normal to Surface .....	493
14.43	Set Position .....	494
14.44	Tool Compensation .....	495
14.45	Translate Path .....	496
14.46	View Robot at Target .....	497
14.47	View Tool at Target .....	498
<b>15</b>	<b>ScreenMaker tab</b> .....	<b>499</b>
15.1	Introduction to ScreenMaker .....	499
15.2	Development environment .....	502
15.3	Working with ScreenMaker .....	507
15.3.1	Managing projects .....	507
15.3.2	Application variables .....	525
15.3.3	Data binding .....	526
15.3.4	ScreenMaker Doctor .....	529
15.4	Frequently asked questions .....	532
15.5	Tutorial .....	535
15.5.1	Overview .....	535
15.5.2	Designing the FlexArc operator panel .....	536
15.5.3	Designing the screen .....	539
15.5.4	Building and deploying the project .....	545
<b>Index</b>		<b>547</b>

# Overview of this manual

## About this manual

RobotStudio is a PC application for modeling, offline programming, and simulation of robot cells. This manual describes how to create, program and simulate robot cells and stations using RobotStudio. This manual also explains the terms and concepts related to both offline and online programming.

## Usage

This manual should be used when working with the offline or online functions of RobotStudio.

## Who should read this manual?

This manual is intended for RobotStudio users, proposal engineers, mechanical designers, offline programmers, robot technicians, service technicians, PLC programmers, Robot programmers, and Robot System integrators.

## Prerequisites

The reader should have basic knowledge of:

- Robot programming
- Generic Windows handling
- 3D CAD programs

## Organization of chapters

The operating manual is structured in the following chapters:

Chapter		Contents
1	<a href="#">Introduction to RobotStudio on page 21</a>	Contains installation instructions, basic explanations of the terms and concepts related to robotics and programming, and a description of the GUI.
2	<a href="#">Building stations on page 75</a>	Describes how to build stations in RobotStudio. This includes importing and configuring the equipment to be simulated, as well as testing the reachability for finding the optimal station layout.
3	<a href="#">Programming robots on page 103</a>	Describes how to create robot movements, I/O signals, process instructions and logics in a RAPID program for the robots. It also describes how to run and test the program.
4	<a href="#">Simulating programs on page 135</a>	Describes how to simulate and validate robot programs.
5	<a href="#">Deploying and distributing on page 145</a>	Describes how to transfer systems between RobotStudio's virtual controllers and real IRC5 controllers, how to copy programs, how to package an active station for moving between RobotStudio PCs, and how to capture a screen.
6	<a href="#">Working online on page 149</a>	Covers the functionality of the Minimal Installation, describing such online functions as building systems (with offline examples), handling I/O and events, and configuring systems.

*Continues on next page*

Chapter		Contents
7	<a href="#">File tab on page 189</a>	Describes the options to create new station, create new robot system, connect to a controller, save station as viewer, and RobotStudio options.
8	<a href="#">Home tab on page 203</a>	Describes the controls required for building stations, creating systems, programming paths and placing items.
9	<a href="#">Modeling tab on page 261</a>	Describes the controls for creating and grouping components, creating bodies, measurements and CAD operations.
10	<a href="#">Simulation tab on page 327</a>	Describes the controls for setting up, configuring, controlling, monitoring, and recording simulations.
11	<a href="#">Controller tab on page 357</a>	Describes the controls for managing a real controller and also the controls for synchronization, configuration and tasks assigned to the virtual controller (VC).
12	<a href="#">RAPID tab on page 413</a>	Describes the features of the RAPID editor, management of RAPID files and other controls for RAPID programming.
13	<a href="#">Add-Ins tab on page 443</a>	Describes the control for PowerPacs .
14	<a href="#">Context menus on page 447</a>	Describes the options available from the context menus.
15	<a href="#">ScreenMaker tab on page 499</a>	Describes the ScreenMaker development tool, how to manage projects in ScreenMaker and the various menus and commands used in the application.

## References

Reference	Document Id
<i>Product manual - IRC5</i> IRC5 of design M2004	3HAC021313-001
<i>Product manual - IRC5</i> IRC5 of design 14	3HAC047136-001
<i>Operating manual - IRC5 with FlexPendant</i>	3HAC16590-1
<i>Technical reference manual - RAPID overview</i>	3HAC16580-1
<i>Technical reference manual - System parameters</i>	3HAC17076-1
<i>Application manual - MultiMove</i>	3HAC021272-001
<i>Application manual - Conveyor tracking</i>	3HAC16587-1
<i>Application manual - SafeMove</i>	3HAC030053-001
<i>Application manual - Electronic Position Switches</i>	3HAC027709-001
<i>Application manual - Integrated Vision</i>	3HAC044251-001

## Revisions

Revision	Description
A	First revision, called RobotStudio 2008, released for Partner Days. The entire manual has been adapted to the new GUI, in which RobotStudio On-line has been integrated.

Continues on next page

Revision	Description
B	<p>Released with RobotStudio 5.12.</p> <p>The following updates were made in the manual:</p> <ul style="list-style-type: none"> <li>• <a href="#">Conveyor Tracking on page 354</a></li> <li>• <a href="#">Create Conveyor mechanism on page 317</a></li> <li>• <a href="#">Conveyor Simulation on page 355</a></li> <li>• <a href="#">Two robot systems in same task frame position on page 77</a></li> <li>• <a href="#">Two robot systems in different task frame positions on page 79</a></li> <li>• <a href="#">Creating a system with external axes automatically on page 81</a></li> <li>• <a href="#">Track motion of type RTT or IRBTx003 on page 83</a></li> <li>• <a href="#">Track motion of type IRBTx004 on page 84</a></li> <li>• <a href="#">The Operator Window on page 60</a></li> <li>• <a href="#">Station Viewer on page 193</a></li> <li>• <a href="#">Recording the simulation on page 353</a></li> <li>• Viewpoint</li> <li>• <a href="#">Linked Geometry on page 467</a></li> </ul>
C	<p>Released with RobotStudio 5.13.</p> <ul style="list-style-type: none"> <li>• Merged chapters <i>The Offline tab</i> and <i>The Online tab</i></li> <li>• Added the missing information from RobotStudio Online manual.</li> <li>• Integrated ScreenMaker. See <a href="#">ScreenMaker on page 363</a>.</li> </ul> <p>Added the following new contents:</p> <ul style="list-style-type: none"> <li>• <a href="#">Smart Component on page 264</a></li> <li>• <a href="#">The Simulation Watch window on page 294</a></li> <li>• <a href="#">The Documents window on page 62</a></li> <li>• <a href="#">Station Logic on page 338</a></li> <li>• <a href="#">Simulation Setup on page 329</a></li> </ul> <p>Updated the changes related to handling Task Frames.</p> <ul style="list-style-type: none"> <li>• Updated <a href="#">Modifying Task frame on page 408</a>.</li> <li>• Added <a href="#">Placing robots on page 100</a>.</li> <li>• Updated <a href="#">Creating a system from layout on page 206</a>.</li> </ul>
D	<p>Released with RobotStudio 5.13.02.</p> <p>The ScreenMaker tutorial was updated. See <a href="#">Tutorial on page 535</a>.</p>
E	<p>Released with RobotStudio 5.14.</p> <ul style="list-style-type: none"> <li>• Added <a href="#">The Controller Status window on page 58</a>.</li> <li>• Updated the sections <a href="#">Simulation Setup on page 329</a> and <a href="#">Simulation Control on page 340</a>.</li> <li>• Moved <a href="#">RAPID Watch window on page 440</a> to the chapter <i>Common features in Online and Offline tabs</i>.</li> <li>• Updated <a href="#">The Documents window on page 62</a> (added <i>Station mode</i>).</li> <li>• Updated <a href="#">Creating and loading a Station Viewer on page 193</a> (Record to Viewer)</li> <li>• Added <a href="#">Jog Reorient on page 248</a>.</li> <li>• Added The 3D View group.</li> <li>• Updated <a href="#">The Compose tab on page 266</a> (added <i>Export to XML</i> and updated <i>Base Component</i> menu).</li> <li>• Updated <a href="#">Coordinate systems on page 29</a> (improved task frame description).</li> <li>• Updated <a href="#">Supported 3D formats on page 38</a> (information on <i>CAD Converters</i>)</li> </ul> <p>Added the following new contents:</p> <ul style="list-style-type: none"> <li>• <a href="#">AutoPath on page 225</a></li> </ul>

Continues on next page

Revision	Description
	<ul style="list-style-type: none"> <li>• <a href="#">Online Monitor on page 393</a></li> <li>• <a href="#">Adjust Robtargets on page 432</a></li> <li>• <a href="#">Using the RAPID Profiler on page 438</a></li> <li>• Markup</li> <li>• <a href="#">Signal Analyzer on page 345</a></li> <li>• <a href="#">External Axis Interpolation on page 460</a></li> <li>• <a href="#">Auto Configuration on page 451</a></li> <li>• <a href="#">The Design tab on page 275</a></li> </ul> <p>The following are the ScreenMaker updates:</p> <ul style="list-style-type: none"> <li>• Added <a href="#">ScreenMaker Doctor on page 529</a>.</li> <li>• Added new controls <a href="#">VariantButton on page 513</a> and <a href="#">Conditional-Trigger on page 514</a>.</li> <li>• Updated <a href="#">Creating a new project on page 507</a> (added <i>pre-defined templates</i>).</li> <li>• Updated <a href="#">Controller object data binding on page 527</a> (added information on <i>shared data</i>).</li> </ul>
F	<p>Released with RobotStudio 5.14.02.</p> <p>Added the following new contents:</p> <ul style="list-style-type: none"> <li>• Gearbox Heat Prediction</li> <li>• <a href="#">External Axis Wizard on page 209</a></li> </ul> <p>Added the following new contents in Settings tab:</p> <ul style="list-style-type: none"> <li>• <a href="#">Selecting a Task on page 241</a></li> <li>• <a href="#">Selecting a Workobject on page 242</a></li> <li>• <a href="#">Selecting a Tool on page 243</a></li> </ul> <p>Updated <a href="#">Creating boot media on page 172</a> (added information on creating a new system)</p> <p>Added information on Logic Expression in <a href="#">Signals and Properties on page 276</a></p> <p>Added a Note for Call .Net Method in <a href="#">Designing screens on page 509</a> of the ScreenMaker tab</p> <p>Added information on I-start in <a href="#">Result on page 168</a> for the Modifying a system section</p> <p>Added information on Offs in the Note for <a href="#">Prerequisites on page 432</a> in Using Adjust Robtargets</p> <p>Added a Note for the Execute button in <a href="#">Using Adjust Robtargets on page 432</a></p> <p>Added information on Always on top in Create Markup of the Markup section</p> <p>Updated Note for Using ScreenMaker Doctor in <a href="#">ScreenMaker Doctor on page 529</a></p>
G	<p>Released with RobotStudio 5.14.02.01.</p> <p>Added <a href="#">How to activate RobotStudio - Network License on page 43</a></p>
H	<p>Released with RobotStudio 5.14.03.</p> <p>Added a note regarding the usage of .NET DLLs under <a href="#">Advanced Actions on page 515</a></p> <p>Added scenarios in <a href="#">Errors fixed by ScreenMaker Doctor on page 529</a></p> <p>Updated the procedure in Creating Markup</p> <p>Updated the note in <a href="#">Prerequisites on page 432</a> for Adjust Robtargets</p> <p>Updated the procedure for <a href="#">Using Adjust Robtargets on page 432</a></p> <p>Updated the table under LogicExpression for <a href="#">Signals and Properties on page 276</a></p>

Continues on next page

Revision	Description
	<p>Added a procedure for adding events to a menu item under <a href="#">CommandBar on page 513</a></p> <p>Updated the details for Creating autopath under <a href="#">AutoPath on page 225</a></p> <p>Updated the table showing the <a href="#">Supported 3D formats on page 38</a></p>
J	<p>Released with RobotStudio 5.15.</p> <p><b>In addition to the following important updates, numerous minor improvements and corrections have been made throughout the document.</b></p> <p>Introduced the following new chapters containing both new features and also updated features;</p> <ul style="list-style-type: none"> <li>• <a href="#">Controller tab on page 357</a>, which contain features related to real and virtual controllers.</li> <li>• <a href="#">RAPID tab on page 413</a>, which contains features related to RAPID programming.</li> </ul> <p>Added the following new content:</p> <ul style="list-style-type: none"> <li>• <a href="#">Edit RAPID data on page 425</a></li> <li>• <a href="#">Transfer on page 376</a></li> <li>• <a href="#">Stopwatch on page 344</a></li> <li>• <a href="#">Go to Visualization and Go to Declaration on page 463</a></li> <li>• <a href="#">Offset Position on page 483</a></li> <li>• <a href="#">Protected Smart Component on page 486</a></li> </ul> <p>Updated, reworked the following sections:</p> <ul style="list-style-type: none"> <li>• <a href="#">Edit RAPID code on page 416</a></li> <li>• <a href="#">RAPID Watch window on page 440</a></li> <li>• <a href="#">Installing and licensing RobotStudio on page 40</a>, and in particular <a href="#">How to activate RobotStudio - Network License on page 43</a></li> <li>• <a href="#">Virtual Controller on page 85</a></li> <li>• <a href="#">Screen Capture on page 147</a></li> <li>• <a href="#">Pack and Go on page 191</a> and <a href="#">Unpack and Work on page 192</a></li> </ul>
K	<p>Released with RobotStudio 5.15.01.</p> <ul style="list-style-type: none"> <li>• Added section <a href="#">What is RAPID array on page 533</a>.</li> <li>• Added a sample code snippet to the advanced option <a href="#">Call .Net Method</a>. See <a href="#">Advanced Actions on page 515</a>.</li> <li>• Added an advanced option <a href="#">Call FP Standard View</a>. See <a href="#">Advanced Actions on page 515</a>.</li> <li>• Added a note in the section <a href="#">Controller object data binding on page 527</a>.</li> <li>• Renamed the section <a href="#">ScreenMaker Doctor scenarios</a> as <a href="#">Errors fixed by ScreenMaker Doctor</a> and made some updates. See <a href="#">Errors fixed by ScreenMaker Doctor on page 529</a>.</li> </ul>
L	<p>Released with RobotStudio 5.60.</p> <ul style="list-style-type: none"> <li>• Removed all instances of VSTA, Import of S4 stations and Defeaturizing and updated the ScreenRecorder section.</li> <li>• Updated the information on <a href="#">Backup and save in Controller grants on page 401</a>.</li> <li>• Updated the section <a href="#">Working online on page 149</a> with information on the new main computer DSQC1000.</li> <li>• Updated the section <a href="#">Overview on page 474</a>.</li> <li>• Updated the section <a href="#">Adjust Robtargets on page 432</a>.</li> <li>• Added a new section <a href="#">Visualizing Safety Zones in Online Monitor on page 393</a> in Online Monitor.</li> <li>• Updated the section <a href="#">AutoPath on page 225</a>.</li> <li>• Added a new section <a href="#">Migrate backup feature on page 446</a>.</li> </ul>

Continues on next page

Revision	Description
	<ul style="list-style-type: none"><li>• Added a new section <a href="#">Graphics Tools on page 250</a>.</li><li>• Added a new section <a href="#">Tags on page 296</a>.</li></ul>
M	<p>Released with RobotStudio 5.61.</p> <ul style="list-style-type: none"><li>• Added description about the <b>General</b> tab in <a href="#">Modifying Project properties on page 517</a> in ScreenMaker chapter.</li><li>• Added a new section <a href="#">Creating Production Screen Widget on page 521</a> in ScreenMaker chapter.</li><li>• Added a note in <a href="#">General keyboard shortcuts on page 72</a> section.</li><li>• Updated the <a href="#">Supported 3D formats on page 38</a> section.</li><li>• Updated the prerequisites in the section <a href="#">ScreenMaker on page 363</a>.</li></ul>

# Product documentation, IRC5

---

## Categories for manipulator documentation

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

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

---

## Product manuals

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

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

---

## Technical reference manuals

The technical reference manuals describe reference information for robotics products.

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

---

## Application manuals

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

*Continues on next page*

An application manual generally contains information about:

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

---

### Operating manuals

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

The group of manuals includes (among others):

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

# Safety

---

## Safety of personnel

A robot is heavy and extremely powerful regardless of its speed. A pause or long 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.

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

---

## Safety regulations

Before beginning work with the robot, make sure you are familiar with the safety regulations described in the manual *Operating manual - General safety information*.

**This page is intentionally left blank**

# **1 Introduction to RobotStudio**

## **1.1 What is RobotStudio**

RobotStudio is a PC application for modeling, offline programming, and simulation of robot cells.

RobotStudio allows you to work with an off-line controller, which is a virtual IRC5 controller running locally on your PC. This offline controller is also referred to as the virtual controller (VC). RobotStudio also allows you to work with the real physical IRC5 controller, which is simply referred to as the real controller.

When RobotStudio is used with real controllers, it is referred to as the online mode. When working without being connected to a real controller, or while being connected to a virtual controller, RobotStudio is said to be in offline mode.

RobotStudio offers the following installation options:

- Complete
- Custom, allowing user-customized contents and paths
- Minimal, allowing you to run RobotStudio in online mode only.

# 1 Introduction to RobotStudio

## 1.2.1 Hardware concepts

## 1.2 Terms and concepts

### 1.2.1 Hardware concepts

#### Overview

This section introduces the hardware in a typical IRC5 robot cell. For detailed explanations, see the manuals related to IRC5 robots specified in [References on page 12](#).

#### Standard hardware

The table below describes the standard hardware in an IRC5 robot cell.

Hardware	Explanation
Robot manipulator	An ABB industrial robot.
Control module	Contains the main computer that controls the motion of the manipulator. This includes RAPID execution and signal handling. One control module can be connected to 1 – 4 drive modules.
Drive module	A module containing the electronics that power the motors of a manipulator. The drive module can contain up to nine drive units, each controlling one manipulator joint. Since the standard robot manipulators have six joints, you usually use one drive module per robot manipulator.
FlexController	The controller cabinet for the IRC5 robots. It consists of one control module and one drive module for each robot manipulator in the system.
FlexPendant	The programming pendant, connected to the control module. Programming on the FlexPendant is referred to as “online programming”.
Tool	A device usually mounted on the robot manipulator to allow it to perform specific tasks, such as gripping, cutting or welding. The tool can also be stationary, see below for more information.

#### Optional hardware

The table below describes the optional hardware for an IRC5 robot cell.

Hardware	Explanation
Track manipulator	A moving stand holding the robot manipulator to give it a larger work space. When the control module controls the motion of a track manipulator, it is referred to as a “Track External Axis”.
Positioner manipulator	A moving stand normally holding a work piece or a fixture. When the control module controls the motion of a positioner manipulator, it is referred to as an “External Axis”.
FlexPositioner	A second robot manipulator acting as a positioner manipulator. It is controlled by the same control module as the positioner manipulator.
Stationary tool	A device that stands in a fixed location. The robot manipulator picks up the work piece and brings it to the device to perform specific tasks, such as gluing, grinding or welding.
Work piece	The product being worked on.

*Continues on next page*

Hardware	Explanation
Fixture	A construction holding the work piece in a specific position so that the repeatability of the production can be maintained.

# 1 Introduction to RobotStudio

## 1.2.2 RobotWare concepts

## 1.2.2 RobotWare concepts

### Overview

This section introduces terminology regarding RobotWare. For detailed explanations, see the manuals related to IRC5 robots specified in [References on page 12](#).

### RobotWare

The table below describes the RobotWare terminology and concepts that can be useful when working with RobotStudio.

Concept	Explanation
RobotWare	As a concept, refers to both the software used to create a RobotWare System and the RobotWare systems themselves.
RobotWare DVD	Delivered with each control module. On the DVD you will find the RobotWare installation and some other useful software. Check the Release Notes on your DVD for specifications.
RobotWare installation	<p>When installing RobotWare on a PC, you install into the mediapool the specific versions of the files from which RobotStudio uses to create the RobotWare system.</p> <p>When installing RobotStudio, only one version of RobotWare will be installed. To simulate a specific RobotWare system, the RobotWare version used for this particular RobotWare system must be installed on your PC.</p>
RobotWare Key	<p>Used when you create a new RobotWare system or upgrade an existing system. The RobotWare keys unlock the RobotWare options included in the system, and determine the RobotWare version from which the RobotWare system will be built.</p> <p>For IRC5 systems there are three types of RobotWare keys:</p> <ul style="list-style-type: none"><li>• The controller key, which specifies the controller and software options.</li><li>• The drive keys, which specify the robots in the system. The system has one drive key for each robot it uses.</li><li>• Additional option keys, which specify additional options, like positioner external axes.</li></ul> <p>A virtual key allows you to select any RobotWare options you wish, but a RobotWare system created from a virtual key can only be used in a virtual environment such as RobotStudio.</p>
RobotWare system	<p>A set of software files that, when loaded into a controller, enables all functions, configurations, data and programs controlling the robot system.</p> <p>RobotWare systems are created in the RobotStudio software. The systems can be stored and saved on a PC, as well as on the control module.</p> <p>RobotWare systems can be edited by RobotStudio or the FlexPendant.</p>
RobotWare version	<p>Each RobotWare is released with a major and a minor version number, separated by a dot. The RobotWare version for IRC5 is 5.xx, where xx identifies the minor version.</p> <p>When ABB releases a new robot model, a new RobotWare version will be released with support for the new robot.</p>

*Continues on next page*

Concept	Explanation
Mediapool	<p>The mediapool is a folder on the PC in which each RobotWare version is stored in a folder of its own.</p> <p>The files of the mediapool are used to create and implement all the different RobotWare options. Therefore, the correct RobotWare version must be installed in the mediapool when creating RobotWare systems or running them on virtual controllers.</p>

# 1 Introduction to RobotStudio

## 1.2.3 RAPID concepts

### 1.2.3 RAPID concepts

#### Overview

This section introduces the basic terminology of RAPID. The manuals related to RAPID and programming are listed in [References on page 12](#).

#### Terminology of the RAPID structure

The table below describes the RAPID terminology that you may come across when working with RobotStudio. The concepts are listed by size, from most basic to increasingly large.

Concept	Explanation
Data declaration	Used to create instances of variables or data types, like num or tooldata.
Instruction	The actual code commands that make something happen, for example, setting data to a specific value or a robot motion. Instructions can only be created inside a routine.
Move instructions	Create the robot motions. They consist of a reference to a target specified in a data declaration along with parameters that set motion and process behavior. If inline targets are used, the position is declared in the move instructions.
Action instruction	Instructions that perform other actions than moving the robot, such as setting data or sync properties.
Routine	Usually a set of data declarations followed by a set of instructions implementing a task. Routines can be divided into three categories: procedures, functions and trap routines.
Procedure	A set of instructions that does not return a value.
Function	A set of instructions that returns a value.
Trap	A set of instructions that is triggered by an interrupt.
Module	A set of data declarations followed by a set of routines. Modules can be saved, loaded and copied as files. Modules are divided into program modules and system modules.
Program module (.mod)	Can be loaded and unloaded during execution.
System module (.sys)	Used mainly for common system-specific data and routines, for example, an arcware system module that is common for all arc robots.
Program files (.pgf)	In IRC5 a RAPID program is a collection of module files (.mod) and the program file (.pgf.) that references all the module files. When loading a program file, all old program modules are replaced by those referenced in the .pgf file. System modules are unaffected by program load.

## 1.2.4 Concepts of programming

### Overview

This section introduces the terminology regarding programming. The manuals related to programming and IRC5 Robots are listed in [References on page 12](#).

### Programming concepts

The table below describes the terminology and concepts that are used in robot programming.

Concept	Explanation
Online programming	Programming when connected to a real controller. This expression also implies using the robot to create positions and motion.
Offline programming	Programming without being connected to the robot or the real controller.
True offline programming	Refers to the ABB Robotics concept of connecting a simulation environment to a virtual controller. This enables not only program creation, but also program testing and optimizing offline.
Virtual controller	A software that emulates a FlexController to allow the same software (the RobotWare system) that is controlling the robots to run on a PC. This gives the same behavior of the robots offline as you get online.
MultiMove	Running multiple robot manipulators with the same control module.
Coordinate systems	Used to define positions and orientations. When programming a robot, you can take advantage of using different coordinate systems to more easily position objects relative to each other.
Frame	A synonym for coordinate system.
Workobject calibration	If all your targets refer to workobjects, you only need to calibrate the workobjects when deploying offline programs.

# 1 Introduction to RobotStudio

---

## 1.2.5 Targets and paths

### 1.2.5 Targets and paths

---

#### Overview

Targets (positions) and paths (sequences of move instructions to targets) are used when programming robot motions in RobotStudio.

When you synchronize the RobotStudio station to the virtual controller, RAPID programs are created from the paths.

#### Targets

A target is a coordinate that the robot shall reach. It contains the following information:

Information	Description
Position	The position of the target, defined in a workobject coordinate system, see <a href="#">Coordinate systems on page 29</a> .
Orientation	The orientation of the target, relative to the orientation of the workobject. When the robot reaches the target, it will align the TCP's orientation with the target's orientation, see <a href="#">Coordinate systems on page 29</a> .
Configuration	Configuration values that specify how the robot shall reach the target. For more information, see <a href="#">Robot axis configurations on page 35</a> .

Targets are converted to instances of the data type *robtarget* when synchronized to the virtual controller.

#### Paths

A sequence of move instructions, paths are used to make the robot move along a sequence of targets.

Paths are converted to procedures when synchronized to the virtual controller.

#### Move instructions

A move instruction consists of:

- a reference to a target
- motion data, such as motion type, speed and zone
- a reference to a tooldata
- a workobject reference

#### Action instructions

An action instruction is a RAPID string that can be used for setting and changing parameters. Action instructions can be inserted before, after or between instruction targets in paths.

### 1.2.6 Coordinate systems

---

#### Overview

This section provides an introduction to the coordinate systems used mostly for offline programming. In RobotStudio, you can either use the coordinate systems (that are explained below) or the user-defined coordinated systems for co-relating elements and objects.

#### Hierarchy

The coordinate systems are co-related hierarchically. The origin of each coordinate system is defined as a position in one of its ancestries. The following are the descriptions of the commonly used coordinate systems.

#### Tool Center Point Coordinate system

The tool center point coordinate system, also called TCP, is the center point of the tool. You can define different TCPs for one robot. All robots have one predefined TCP at the robot's tool mounting point, called *tool0*.

When a program runs, the robot moves the TCP to the programmed position.

#### RobotStudio World Coordinate system

The RobotStudio world coordinate system represents the entire station or robot cell. This is the top of the hierarchy to which all other coordinate systems are related (when using RobotStudio).

#### Base Frame (BF)

The base coordinate system is called the Base Frame (BF). Each robot in the station, both in RobotStudio and the real world has a base coordinate system which is always located at the base of the robot.

#### Task Frame (TF)

The Task Frame represents the origin of the robot controller world coordinate system in RobotStudio.

The following picture illustrates the difference between the base frame and the task frame.

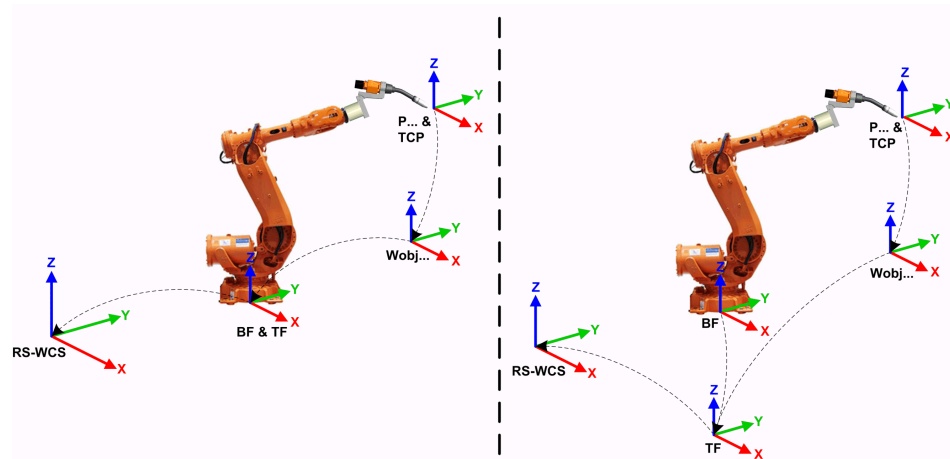
*Continues on next page*

# 1 Introduction to RobotStudio

## 1.2.6 Coordinate systems

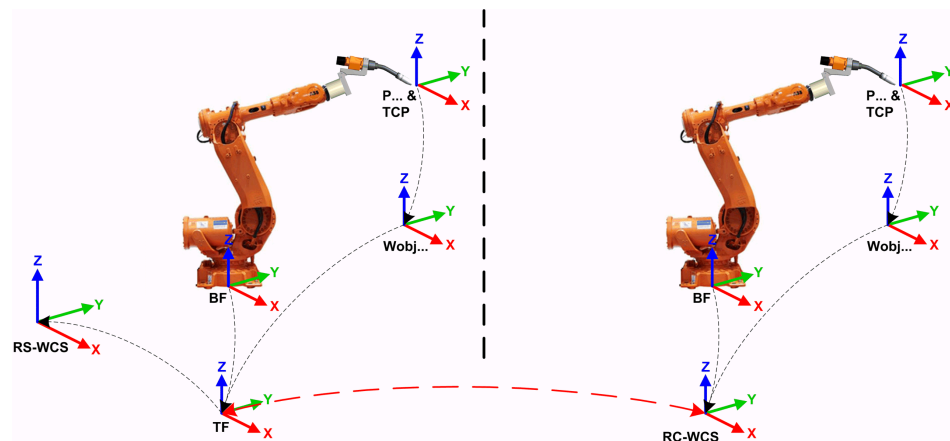
*Continued*

In the picture to the left, the task frame is located at the same position as the robot base frame. In the picture to the right, the taskframe has been moved to another position.



en1000001303

The following picture illustrates how a task frame in RobotStudio is mapped to the robot controller coordinate system in the real world. For example, on the shop floor.



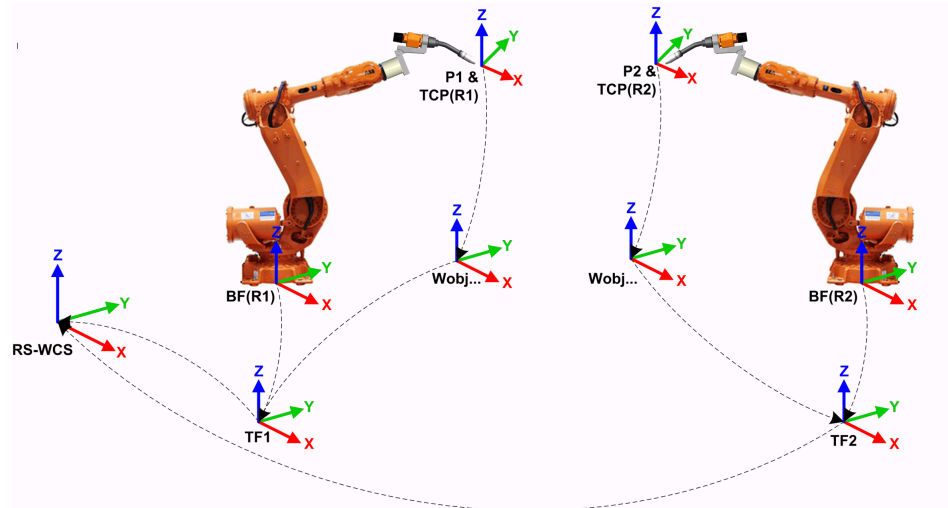
en1000001304

RS-WCS	World coordinate system in RobotStudio
RC-WCS	World coordinate system as defined in the robot controller. It corresponds to the task frame of RobotStudio.
BF	Robot Base Frame
TCP	Tool Center Point
P	Robot target
TF	Task Frame
Wobj	Workobject

*Continues on next page*

### Stations with multiple robot systems

For a single robot system, RobotStudio's task frame corresponds to the robot controller world coordinate system. When several controllers are present in the station, the task frame allows the connected robots to work in different coordinate systems. That is, the robots can be located independent of each other by defining different task frames for each robot.



en1000001442

RS-WCS	World coordinate system in RobotStudio
TCP(R1)	Tool Center Point of robot 1
TCP(R2)	Tool Center Point of robot 2
BF(R1)	Base Frame of robot system 1
BF(R2)	Base Frame of robot system 2
P1	Robot target 1
P2	Robot target 2
TF1	Task Frame of robot system 1
TF2	Task Frame of robot system 2
Wobj	Workobject

Continues on next page

# 1 Introduction to RobotStudio

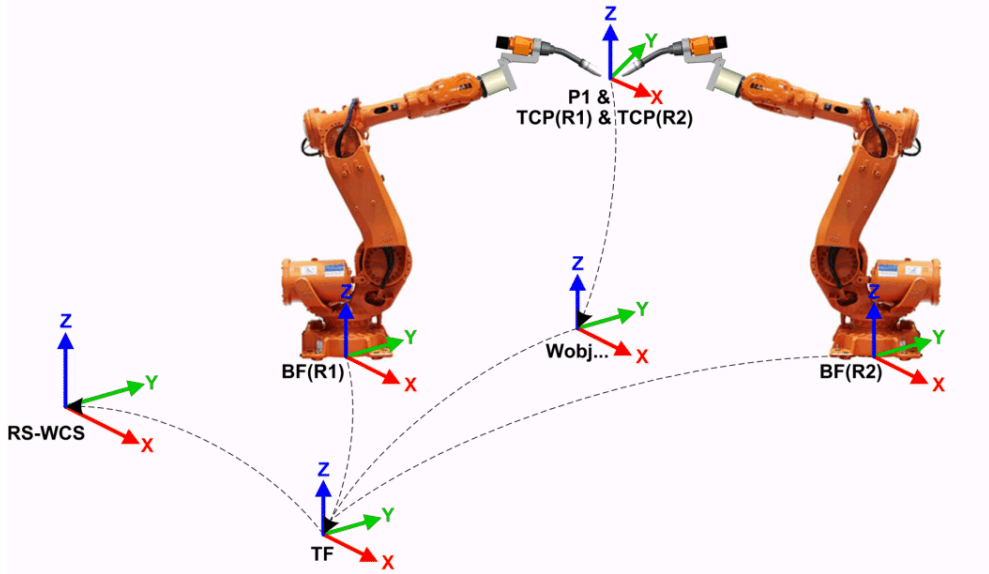
## 1.2.6 Coordinate systems

Continued

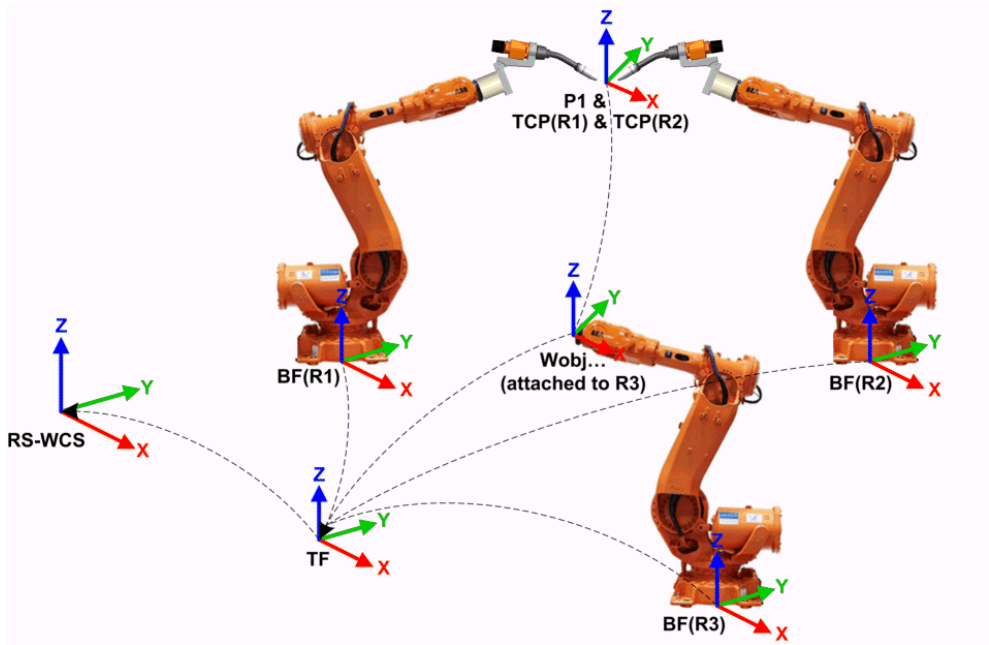
### MultiMove Coordinated systems

The MultiMove functions helps you create and optimize programs for MultiMove systems where one robot or positioner holds the work piece and other robots operate on it.

When using a robot system with the RobotWare option *MultiMove Coordinated*, it is important that the robots are working in the same coordinate system. As such, RobotStudio do not allow task frames of the controller to be separated.



en1000001305



en1000001306

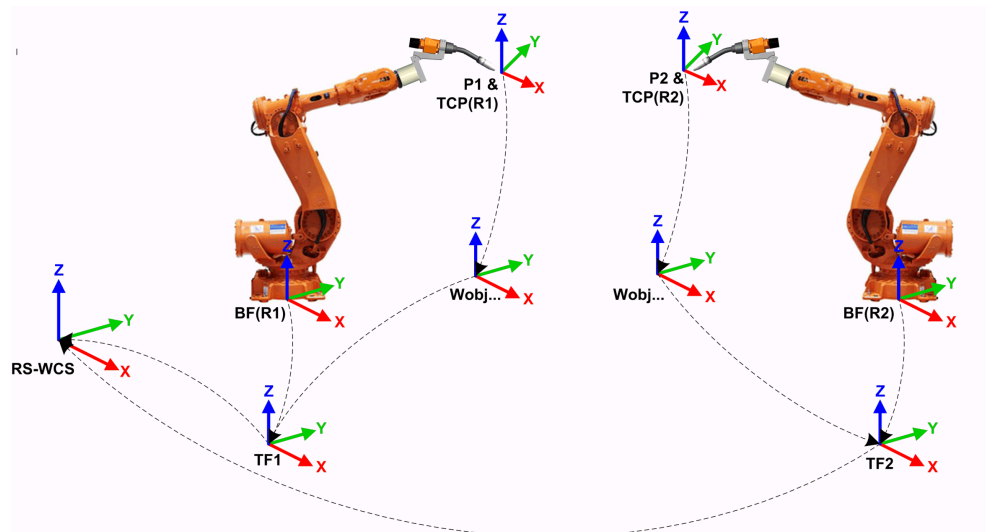
RS-WCS	World coordinate system in RobotStudio
TCP(R1)	Tool Center Point of robot 1
TCP(R2)	Tool Center Point of robot 2

Continues on next page

BF(R1)	Base Frame of robot 1
BF(R2)	Base Frame of robot 2
BF(R3)	Base Frame of robot 3
P1	Robot target 1
TF	Task Frame
Wobj	Workobject

### MultiMove Independent systems

For a robot system with the RobotWare option *MultiMove Independent*, robots operate simultaneously and independently while being controlled by one controller. Even though there is only one robot controller world coordinate system, robots often work in separate coordinate systems. To allow this setup in RobotStudio, the task frames for the robots can be separated and positioned independent of each other.



en1000001308

RS-WCS	World coordinate system in RobotStudio
TCP(R1)	Tool Center Point of robot 1
TCP(R2)	Tool Center Point of robot 2
BF(R1)	Base Frame of robot 1
BF(R2)	Base Frame of robot 2
P1	Robot target 1
P2	Robot target 2
TF1	Task Frame 1
TF2	Task Frame 2
Wobj	Workobject

Continues on next page

# 1 Introduction to RobotStudio

---

## 1.2.6 Coordinate systems

*Continued*

---

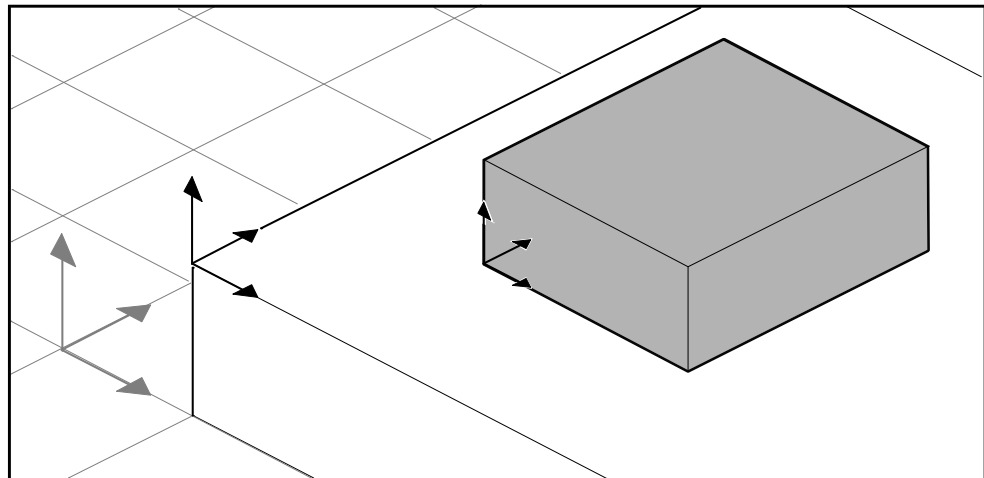
### Workobject coordinate system

The workobject normally represents the physical work piece. It is composed of two coordinate systems: the *User frame* and the *Object frame*, where the latter is a child to the former. When programming a robot, all targets (positions) are related to the object frame of a workobject. If no other workobject is specified, the targets will be related to the default *Wobj0*, which always coincides with the base frame of the robot.

Using workobjects provides the chance to easily adjust robot programs with an offset, if the location of the work piece has been changed. Thus, workobjects can be used for calibrating offline programs. If the placement of the fixture or work piece relative to the robot in the real station does not completely match the placement in the offline station, you simply adjust the position of the workobject.

Workobjects are also used for coordinated motion. If a workobject is attached to a mechanical unit (and the system uses the option for coordinated motion), the robot will find the targets in the workobject even when the mechanical unit moves the workobject.

In the picture below the grey coordinate system is the world coordinate system, and the black ones are the object frame and the user frame of the workobject. Here the user frame is positioned at the table or fixture and the object frame at the workpiece.



xx0500001519

---

### User coordinate systems

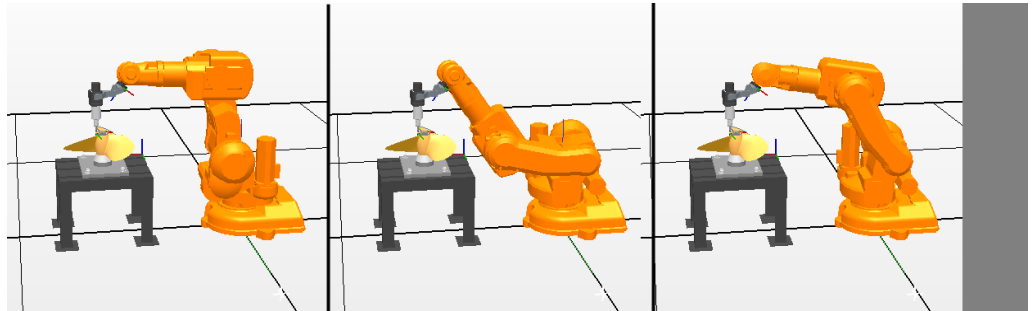
User coordinate systems are used for creating reference points of your choice. For example, you can create user coordinate systems at strategic points in the work piece to simplify programming.

### 1.2.7 Robot axis configurations

---

#### Axis configurations

Targets are defined and stored as coordinates in a WorkObject coordinate system. When the controller calculates the position of the robot axes for reaching the target, it will often find more than one possible solution to configuring the robot axes.



configur

To distinguish between the different configurations, all targets have a configuration value that specifies the quadrant in which each axis shall be located.

---

#### Storing axis configurations in targets

For targets that are taught after jogging the robot to the position, the used configuration will be stored in the target.

Targets created by specifying or calculating positions and orientations get a default configuration value (0,0,0,0), which might not be valid for reaching the target.

---

#### Common problems related to robot axis configurations

It is most likely that targets created by other ways than jogging cannot be reached at their default configuration.

Even if all targets in a path have validated configurations, you might encounter problems when running the path if the robot cannot move from one configuration to the other. This is likely to occur where an axis shifts greater than 90 degrees during linear movements.

Repositioned targets keep their configuration, but the configurations are no longer validated. As a result, the problems described above might occur when moving targets.

#### Common solutions for configuration problems

To resolve the problems described above, you can assign a valid configuration to each target and verify that the robot can move along each path. You can also turn configuration monitoring off, which means that you ignore the stored configurations and let the robot find working configurations at runtime. If this is not done the proper way, you might get unexpected results.

In some cases there might not be any working configurations. Possible solutions might then be to reposition the work piece, reorient targets (if acceptable for the process) or add an external axis that either moves the work piece or the robot for increasing reachability.

*Continues on next page*

# 1 Introduction to RobotStudio

---

## 1.2.7 Robot axis configurations

*Continued*

---

### How configurations are denoted

The robot's axis configurations are denoted by a series of four integers, specifying in which quadrant of a full revolution significant axes are located. The quadrants are numbered from zero for positive (counterclockwise) rotation and from -1 for negative (clockwise) rotation.

For a linear axis, the integer specifies the range (in meters) from the neutral position in which the axis is located.

A configuration for a six-axis industrial robot (like IRB 140) may look like:

[ 0 -1 2 1 ]

The first integer (0) specifies the position of axis 1: somewhere in the first positive quadrant (between 0 and 90 degrees rotation).

The second integer (-1) specifies the position of axis 4: somewhere in the first negative quadrant (between 0 and -90 degrees rotation).

The third integer (2) specifies the position of axis 6: somewhere in the third positive quadrant (between 180 and 270 degrees rotation).

The fourth integer (1) specifies the position of axis x, a virtual axis used for specifying the wrist center in relation to other axes.

---

### Configuration monitoring

When executing a robot program, you can choose whether to monitor configuration values. If configuration monitoring is turned off, configuration values stored with the targets are ignored, and the robot will use the configuration closest its current configuration for reaching the target. If turned on, it will only use the specified configuration for reaching the targets.

Configuration monitoring can be turned off and on for joint and linear movements independently and is controlled by the ConfJ and ConfL action instructions.

#### Turning configuration monitoring off

Running a program without configuration monitoring may result in different configurations each time a cycle is executed: When the robot returns to the start position after completing a cycle, it may choose a different configuration than the original.

For programs with linear move instructions this might cause a situation where the robot gets closer and closer its joint limits and eventually will not be able to reach the target.

For programs with joint move instructions this might cause sweeping, unpredictable movements.

#### Turning configuration monitoring on

Running a program with configuration monitoring forces the robot to use the configurations stored with the targets. This results in predictable cycles and predictable motions. In some situations, however, like when the robot moves to a target from an unknown position, using configuration monitoring may limit the robot's reachability.

When programming offline, you must assign a configuration to each target if the program shall be executed with configuration monitoring.

### 1.2.8 Libraries, geometries and CAD files

---

#### Overview

For programming or simulating in RobotStudio, you need models of your work pieces and equipment. Models for some standard equipment are installed as libraries or geometries with RobotStudio. If you have CAD models of your work pieces and custom equipment, these can be imported as geometries to RobotStudio. If you do not have CAD models, you can create them in RobotStudio.

---

#### Difference between geometries and libraries

The objects you import to a station can be either geometries or libraries.

Geometries are basically CAD files, which, when imported, are copied to the RobotStudio station.

Libraries are objects that have been saved in RobotStudio as external files. When you import a library, a link from the station to the library file is created. Accordingly, the station file does not grow in the same way as when importing geometries. Furthermore, besides the geometrical data, library files can contain RobotStudio-specific data. For example, if a tool is saved as a library, the tool data is saved together with the CAD data.

---

#### How geometries are constructed

An imported geometry is displayed as one part in the Objects browser. From RobotStudio's Modeling tab, you can see the components of the geometry.

The top node of the geometry is called a **Part**. The part contains **Bodies**, which can be of the types solid, surface or curve.

**Solid bodies** are 3D objects, made up of **Faces**. You recognize a true 3D solid by this one body containing multiple faces.

**Surface bodies** are 2D objects of just one face. A part that contains several bodies with one face each that together constitute a 3D object is created from 2D surfaces, and is therefore not a true 3D solid. If these parts are not created correctly, they might cause problems both in their display and graphical programming. see [Troubleshooting and optimizing geometries on page 90](#).

**Curved bodies**, represented by the body node alone in the Modeling browser, do not contain any child nodes.

From the Modeling tab, you can edit the parts by adding, moving, rearranging or deleting bodies. Thus, you can optimize existing parts by removing unnecessary bodies, as well as create new parts by grouping bodies.

---

#### Importing and converting CAD files

For importing geometries from single CAD files, you use RobotStudio's import function, see [Importing a station component on page 87](#).

If you need to convert CAD files to other formats or want to change the default settings for the conversion before making the import, you can use the CAD converter installed with RobotStudio before making the import, see [Converting CAD formats on page 89](#).

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.2.8 Libraries, geometries and CAD files

*Continued*

### Supported 3D formats

The native 3D format of RobotStudio is ACIS. RobotStudio contains ACIS R24SP2 which supports later versions of its supported CAD formats.

RobotStudio also supports other formats for which you need an option. The following table shows the supported formats and the corresponding options:

Format	File extensions	Option required	Default target formats
ACIS, reads versions R1 - R24, writes versions R18 - R24	sat	-	IGES, STEP, VDA-FS
IGES, reads up to version 5.3, writes version 5.3	igs, iges	IGES	ACIS, STEP, VDA-FS
STEP, reads versions AP203 and AP214 (geometry only), writes version AP214	stp, step, p21	STEP	ACIS, IGES, VDA-FS
VDA-FS, reads 1.0 and 2.0, writes 2.0	vda, vdafs	VDA-FS	ACIS, IGES, STEP
CATIA V4, reads versions 4.1.9 to 4.2.4	model, exp	CATIA V4	ACIS, IGES, STEP, VDA-FS
CATIA V5, reads versions R6 – R23 (V5-6 R2013)	CATPart, CATProduct	CATIA V5	ACIS, IGES, STEP, VDA-FS
Pro/ENGINEER, reads versions 16 – Creo 2.0	prt, asm	Pro/ENGINEER	ACIS, IGES, STEP, VDA-FS
Inventor, reads V6 – V2014	ipt	Inventor	ACIS, IGES, STEP, VDA-FS
VRML, reads VRML2 (VRML1 not supported)	wrl, vrml, vrml2	-	RsGfx
STL, ASCII STL supported (binary STL not supported)	stl	-	RsGfx
3DStudio	3ds	-	RsGfx
COLLADA 1.4.1	dae	-	RsGfx
OBJ	obj	-	RsGfx

To import these files into RobotStudio, use the **Import Geometry** function.

To convert files to VDA-FS, STEP and IGES, use the standalone **CAD Converter** tool. For converting to other formats, use the **Export Geometry** function in RobotStudio. You need the option for both the target and the source formats when converting files.

*Continues on next page*

#### **Mathematical versus graphical geometries**

A geometry in a CAD file always has an underlying mathematical representation. Its graphical representation, displayed in the graphics window, is generated from the mathematical representation when the geometry is imported to RobotStudio, after which the geometry is referred to as a part.

For this kind of geometry, you can set the detail level of the graphical representation, thus reducing the file size and rendering time for large models and improving the visual display for small models you might want to zoom in on. The detail level only affects the visual display; paths and curves created from the model will be accurate both with coarse and fine settings.

A part can also be imported from a file that simply defines its graphical representation; in this case, there is no underlying mathematical representation. Some of the functions in RobotStudio, such as snap mode and creation of curves from the geometry, will not work with this kind of part.

To customize the detail level settings, see [Options on page 195](#).

# 1 Introduction to RobotStudio

## 1.3 Installing and licensing RobotStudio

### 1.3 Installing and licensing RobotStudio

#### Installation options and prerequisites



#### Note

You should have administrator privileges on the PC before installing RobotStudio.

RobotStudio is categorized into the following two feature levels:

- **Basic** - Offers selected RobotStudio functionality to configure, program, and run a virtual controller. It also includes online features for programming, configuring, and monitoring a real controller connected over Ethernet.
- **Premium** - Offers full RobotStudio functionality for offline programming and simulation of multiple robots. The Premium level includes the features of the Basic level and requires activation.

In addition to the Premium functionality, there are add-ins like PowerPacs and options for CAD converters available.

- PowerPacs provides enhanced features for selected applications.
- Options for CAD converters allows import of different CAD formats.

RobotStudio offers the following installation options:

- **Minimal** - Installs only the features required to program, configure, and monitor a real controller connected over Ethernet.
- **Complete** - Installs all the features required to run the complete RobotStudio. If installed with this option, additional features of Basic and Premium functionality are available.
- **Custom** - Installs user-customized features. This option allows excluding unwanted robot libraries and CAD converters.



#### Note

RobotStudio 5.61 is installed for the Complete installation option on computers that have a 64-bit operating system. The 64-bit edition allows large CAD-models to be imported as it can address more memory than the 32-bit version.

However, the 64-bit edition has the following limitations:

- ScreenMaker, SafeMove Configurator, and EPS Wizard are not supported.
- Add-ins will be loaded from the following folder

```
C:\Program Files (x86)\ABB Industrial IT\Robotics  
IT\RobotStudio 5.61\Bin64\Addins
```

#### How to install RobotStudio on a PC

Action	
1	Insert the <b>RobotWare and RobotStudio DVD</b> in the PC. <ul style="list-style-type: none"><li>• If a menu for the DVD is opened automatically, continue with step 5.</li><li>• If no menu for the DVD is opened, continue with step 2.</li></ul>
2	On the <b>Start</b> menu, click <b>Run</b> .

*Continues on next page*

	Action
3	In the <b>Open</b> box, type the drive letter for your DVD drive followed by: :\launch.exe If your DVD drive has the letter D, then type: D:\launch.exe.
4	Click <b>OK</b> .
5	Select the language for the DVD menu.
6	On the DVD menu, click <b>Install Products</b> .
7	On the <b>Install Products</b> menu, click <b>RobotStudio</b> . This opens the installation wizard, which guides you through the rest of the software installation.
8	After installing <b>RobotStudio</b> , you can proceed with installing <b>RobotWare</b> . Go to the <b>Install Products</b> menu, and click <b>RobotWare</b> . This opens this installation wizard, which guides you through the rest of the RobotWare installation.
9	This step is optional, and is for installing the Track mediapool. On the <b>Install Products</b> menu, click <b>Additional Options</b> . This opens a file browser that displays the Track mediapool installation and other available options. Double-click the <b>TrackMotion</b> folder and then the file <b>setup.exe</b> to start the installation wizard and proceed.

After installing RobotStudio, proceed with activating your RobotStudio installation.

### Knowing which RobotStudio version is installed

The version number of your RobotStudio installation is displayed on the RobotStudio title bar.

### Activation of RobotStudio license

When you start RobotStudio for the first time after installation, you are prompted to enter your 25-digit Activation Key (xxxxx-xxxxx-xxxxx-xxxxx-xxxxx). The software performs in the Basic Functionality mode if you do not use a valid Activation Key. After the installation is activated, you will have valid licenses for the features covered by your subscription.



#### Note

Activation is not required for Minimal installation, or for Basic Functionality mode of the Complete or Custom installation.

### What is Basic Functionality mode

In Basic Functionality mode, which is a reduced functionality mode, RobotStudio allows only the use of the basic features for the real and the virtual controller. No existing files or stations are harmed in this mode. After activating your software, you will have full functionality for the features you have purchased.

A real controller can be programmed, configured and monitored over Ethernet without activating your installation of RobotStudio. Activation, however, will provide access to the Premium productivity features that will make your engineering work more efficient.

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.3 Installing and licensing RobotStudio

*Continued*

### How to activate RobotStudio - Standalone license

Use the Activation Wizard to activate your RobotStudio installation. When you start RobotStudio for the first time after installation, the wizard starts automatically and prompts you for the Activation Key. If you do not want to activate your copy of RobotStudio at installation, you can do so later using the Activation Wizard.



#### Note

If you have a problem with your activation, contact your local ABB customer support representative at the e-mail address or telephone number provided at [www.abb.com/contacts](http://www.abb.com/contacts).

Alternatively, with your activation key attached, send an e-mail to [softwarefactory\\_support@se.abb.com](mailto:softwarefactory_support@se.abb.com).

For using the Activation Wizard, follow this procedure.

	Action
1	On the <b>File</b> tab, click <b>Options</b> and go to <b>General:Licensing</b> .
2	On the Licensing page to the right, click <b>Activation wizard</b> to launch the Activation Wizard.
3	<p>In the Activation Wizard, on the <i>Activate RobotStudio</i> page, indicate whether you have a <b>Standalone License</b> or a <b>Network License</b>, and then click <b>Next</b>.</p> <p>If you have chosen <b>Standalone License</b>, you will proceed to the <i>Activate a Standalone License</i> page. See <a href="#">Activate automatically over the Internet or manually on page 42</a> for further steps.</p> <p>If you have chosen <b>Network License</b>, you will proceed to the <i>Network License</i> page. See <a href="#">How to activate RobotStudio - Network License on page 43</a> for further steps.</p>

### Activate automatically over the Internet or manually

The Activation Wizard gives you two choices on how to proceed. You can choose either automatic activation over the Internet or manual activation. These are explained in the following section.

#### Automatic Activation (recommended)

In automatic activation, the Activation Wizard automatically contacts and sends your activation request to the ABB licensing servers over your Internet connection. Your license will then be automatically installed and your product will be ready for use.

For automatic activation you need an Internet connection and also a valid activation key that has not exceeded the number of installations allowed.

RobotStudio must be restarted after the activation has been successfully completed.



#### Note

If you choose to activate over the Internet but are not currently connected to the Internet, then the wizard alerts you that there is no connection.

*Continues on next page*

### Manual Activation

If the computer does not have an Internet connection, you must proceed with manual activation.

- 1 Create a license request file by selecting the option **Step 1:Create a license request file**.

Proceed through the wizard, enter your activation key and save the license request file to your computer.

- 2 Use removable storage, such as a USB stick, to transfer the file to a computer with an Internet connection. On that computer, open a web browser, go to <http://www101.abb.com/manualactivation/> and follow the instructions given.

The result will be a license key file that should be saved and transferred back to the computer having the installation awaiting activation.

- 3 Relaunch the activation wizard and go through the steps until you reach the *Activate a Standalone License* page.

- 4 Under *Manual Activation*, select the option **Step 3:Install a license file**.

Proceed through the wizard, selecting the license key file when requested. Upon completion, RobotStudio is activated and ready for use.

RobotStudio must be restarted after the activation has been successfully completed.

---

### How to activate RobotStudio - Network License

Network licensing allows you to centralize license management by installing licenses on a single server rather than on each individual client machine. The server administers the licenses to the clients as required. A single network license allows several clients to use the software.

Network Licensing is set up in the following stages:

- 1 Install the server for network licensing (See [Installing the Network Licensing Server on page 43](#))
- 2 Activate the licenses for network licensing (See [Using the SLP Server Web Interface on page 44](#))
- 3 Set up the client for network licensing (See [Setting up Network Licensing in the client on page 46](#))



#### Tip

Network licenses are displayed as *Network* in the **View Installed Licenses** link of the Licensing page.

### Installing the Network Licensing Server

Network licensing in RobotStudio uses the SLP Distributor server as the network licensing server. It manages the allocation of network licenses to the clients.

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.3 Installing and licensing RobotStudio

*Continued*

You can install the SLP Distributor server from the *Utilities\SLP Distributor* directory of the RobotStudio distribution.



### Note

You require administrative privileges for installing and configuring the SLP Distribution server.

The installer requires the following:

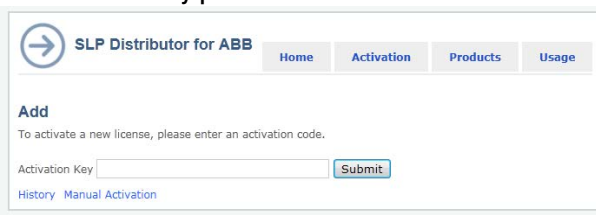
- Windows Server 2008, Windows 8, Windows 7, or Windows Vista
- 32-bit or 64-bit version of Windows
- .NET Framework 3.5 SP1

The SLP Distributor server is installed as a service that starts automatically with Windows. It requires two open TCP ports, by default 2468 (for the web interface) and 8731 (for licensing). The installer opens these ports in the standard Windows Firewall, but any third-party firewall must be configured manually by the system administrator.

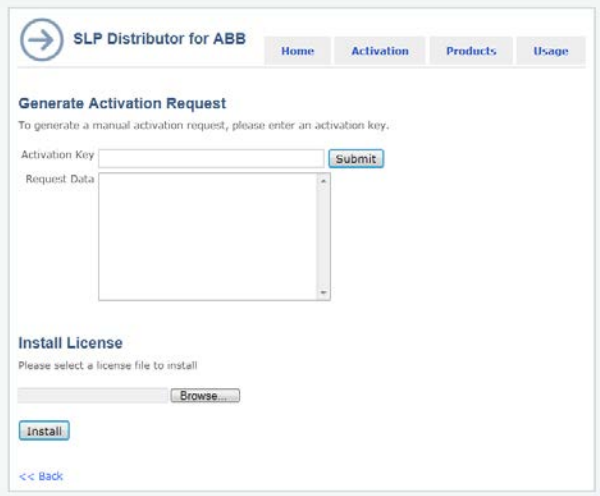
### Using the SLP Server Web Interface

Once the SLP server is online, you can access its web interface on the address `http://<server>:2468/web`.

The following table shows how to use the server's web interface.

To...	Use...
Activate a network license automatically (for PCs with Internet connection)	<p>The <b>Activation</b> tab.</p> <p>Type in the <b>Activation Key</b> provided by ABB, and then click <b>Submit</b>.</p> <p>The number of network licenses activated depends on the activation key provided.</p>  <p>xx1300000052</p>

*Continues on next page*

To...	Use...
Activate a network license manually (for PCs <i>without</i> Internet connection)	<p>The <b>Activation</b> tab.</p> <ol style="list-style-type: none"> <li>1 Click <b>Manual Activation</b>.</li> <li>2 Type in your activation key, provided by ABB, and then click <b>Submit</b>.</li> <li>3 Copy and send the <b>Request Data</b> which appears, by mail to <i>softwarefactory_support@se.abb.com</i>. The license file will be mailed to you.</li> <li>4 Once you receive the license file, click <b>Browse</b> to upload and install the license file.</li> </ol> <p>Your network license is now activated.</p>  <p>xx1300000051</p>
View the installed licenses	<p>The <b>Home</b> tab. Under <b>Dashboard</b>, click <b>Details</b>. Alternatively, click the <b>Products</b> tab. Both these open the <i>Product details for RobotStudio</i> page, where details about the installed licenses are displayed.</p>
View the usage of licenses	<p>The <b>Home</b> tab. Under <b>Dashboard</b>, click <b>Usage</b>. Alternatively, click the <b>Usage</b> tab. Both these open the <i>Current usage of RobotStudio</i> page, which lists the following in a table:</p> <ul style="list-style-type: none"> <li>• Licenses which are currently allocated</li> <li>• Client to which each license is allocated to</li> <li>• Number of remaining licenses available for use</li> </ul> <p>Each table row corresponds to one client system.</p>

# 1 Introduction to RobotStudio

## 1.3 Installing and licensing RobotStudio

*Continued*



### Note

Certain proxy issues upon activation in the SLP server's web interface may produce a messages which only states *Activation failed*. Such a case may happen when the system account that the SLP-Distributor-service is executed in, does not have the rights to read the user profile. As a workaround, follow this procedure:

- 1 Open the **Services** control panel (services.msc)
- 2 Open properties for **Software Potential Distributor**
- 3 Change **Log on as** to an actual named user, preferably the currently logged on user
- 4 Restart the service and re-attempt an activation.
- 5 After your re-attempt, change **Log on as** back to Local System account and restart the service.

### Setting up Network Licensing in the client



### Note

You require Administrative privileges to store this configuration.

You need to use the RobotStudio Activation Wizard in the client system for setting up Network Licensing.

Use this procedure to set up Network Licensing for a client system.

Action
1 On the <b>File</b> tab, click <b>Options</b> and go to <b>General:Licensing</b> .
2 On the Licensing page to the right, click <b>Activation wizard</b> to launch the Activation Wizard.
3 In the Activation Wizard, on the <i>Activate RobotStudio</i> page, choose the option <b>I want to specify a network license server and manage server license</b> , and then click <b>Next</b> . You will proceed to the <i>License Server</i> page.
4 Specify the name or IP address of the License Server, and then click <b>Finish</b> . If Windows UAC is enabled, a confirmation dialog appears. This prompts you to restart RobotStudio in order to start using the specified server. To go to the SLP Distributor server web interface, click the <b>Open the server dashboard</b> link. For information about using the server dashboard, see <a href="#">Using the SLP Server Web Interface on page 44</a> . Note that the changes made are not applied until RobotStudio is restarted.



### Note

For Network Licensing to work, the client system should be online with the server. For information about enabling licensing while working offline, see [Using Commuter Licenses on page 47](#).

*Continues on next page*

### Using Commuter Licenses

Commuter licenses allow a client system to work offline from the license server. You can check out a license from the server for a specified number of days. During this period the checked out license is unavailable to other users. The commuter license is made available for other clients only when it is manually checked in back to the server.

The commuter license in the client system expires when the check out time expires. In this case, on starting RobotStudio in the client system, the Network License dialog opens automatically and prompts you to check in the license back to the server.



#### Note

It is not possible to check out specific features in the license. All features in a license are included when it is checked out.

To check in/check out a commuter license, you need to use the Activation Wizard. Use this procedure to check in/check out a commuter license.

	Action
1	On the <b>File</b> menu, click <b>Options</b> and select <b>General: Licensing</b>
2	On the Licensing page to the right, click <b>Activation wizard</b> to launch the Activation Wizard.
3	In the Activation Wizard, on the <i>Activate RobotStudio</i> page, choose <b>I want to check out or check in a commuter license</b> and click <b>Next</b> . You will proceed to the <i>Commuter License</i> page.
4	Under <i>Commuter License</i> , you will be presented one of the following options as per your requirement: <ul style="list-style-type: none"> <li><b>Check out a commuter license</b> - In the <b>Check out days</b> box specify the specify the number of days for which you wish to keep the license. This option is disabled if you already have a commuter license checked out.</li> <li><b>Check in a commuter license</b> - Choose this option to return the currently checked out license to the server. This option is enabled only if a commuter license is already checked out. If so, the expiration date and time of the license is also displayed.</li> </ul>
5	Click <b>Finish</b> to complete the check in/check out.



#### Tip

Network licenses checked-out as commuter licenses are displayed as **Floating (checked out)** in the **View Installed Licenses** link of the Licensing page.

### Knowing whether your RobotStudio installation is activated

	Action
1	On the <b>File</b> tab, click <b>Options</b> and then go to <b>General:Licensing</b> .
2	On the Licensing page to the right, select <b>View Installed Licenses</b> to see the status of your current license. The <i>Licenses</i> opens, where you can view all the valid licenses for the features covered by your subscription.

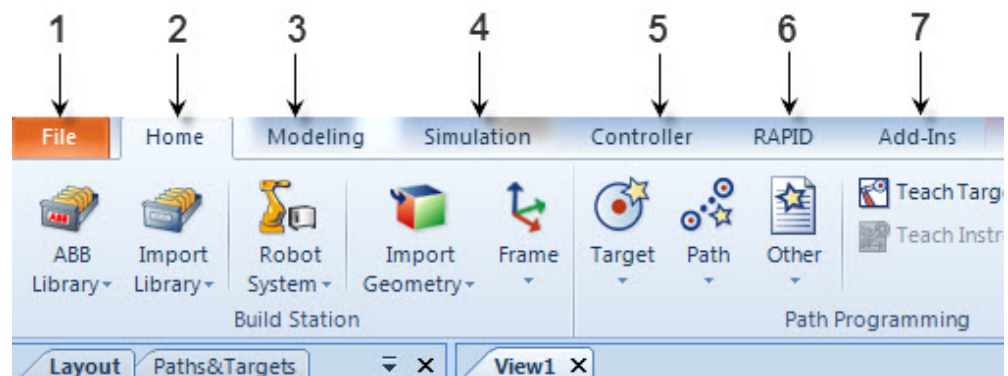
# 1 Introduction to RobotStudio

## 1.4.1 Ribbon, tabs and groups

## 1.4 User interface

### 1.4.1 Ribbon, tabs and groups

The following figure shows the ribbon, tabs and groups of the Graphical User Interface.



en0900000215












	Tab	Description
1	File	Contains the options to create new station, create new robot system, connect to a controller, save station as viewer, and RobotStudio options. For more information, see <a href="#">File tab on page 189</a> .
2	Home	Contains the controls required for building stations, creating systems, programming paths and placing items. For more information, see <a href="#">Home tab on page 203</a> .
3	Modeling	Contains the controls for creating and grouping components, creating bodies, measurements and CAD operations. For more information, see <a href="#">Modeling tab on page 261</a> .
4	Simulation	Contains the controls for setting up, configuring, controlling, monitoring and recording simulations. For more information, see <a href="#">Simulation tab on page 327</a> .
5	Controller	Contains the controls for synchronization, configuration and tasks assigned to the Virtual Controller (VC). It also contains the controls for managing the real controller. For more information, see <a href="#">Controller tab on page 357</a> .
6	RAPID	Contains the integrated RAPID editor, used for editing all robot tasks other than robot motion. For more information, see <a href="#">RAPID tab on page 413</a> .
7	Add-Ins	Contains the control for PowerPacs . For more information, see <a href="#">Add-Ins tab on page 443</a> .

## 1.4.2 Layout browser

### Overview

The layout browser is a hierarchical display of physical items, such as robots and tools.

### Icons

Icon	Node	Description
 xx050000	Robot	The robot in the station.
 xx050001	Tool	A tool.
 xx050002	Link collection	Contains all the links of the objects.
 xx050003	Link	A physical object in a joint connection. Each link is made up of one or several parts.
 xx050004	Frames	Contains all the frames for an object.
 xx050005	Component group	A grouping of parts or other assemblies, carrying its own coordinate systems. It is used to structure a station.
 xx050006	Part	A physical object in RobotStudio. Parts with geometric information are made up of one or more 2D or 3D entities. Parts without geometric information (such as imported .jt files) are empty.
 xx050007	Collision set	Contains all collision sets. Each collision set includes two groups of objects.
 xx050008	Objects group	Contains references to the objects that are subject to collision detection.
 xx050009	Collision set mechanisms	The objects in the collision set.
 xx050010	Frame	The frames in the station.

# 1 Introduction to RobotStudio















## 1.4.3 The Paths & Targets browser

### 1.4.3 The Paths & Targets browser



#### Overview

The paths & targets browser is a hierarchical display of non-physical items.

#### Icons

Icon	Node	Description
 xx050011	Station	Your station in RobotStudio.
 xx050012	Virtual Controller	The system for controlling the robots, just like a real IRC5 controller.
 xx050013	Task	Contains all logical elements in the station, such as targets, paths, workobjects, tooldata and instructions.
 xx0500001376	Tooldata Collection	Contains all tooldata.
 xx050014	Tooldata	A tooldata for a robot or a task.
 xx050015	Workobjects & Targets	Contains all workobjects and targets for the task or robot.
 xx050016	Jointtarget Collection and Jointtarget	A specified position of the robot axes.
 xx050017	Workobject Collection and Workobject	The workobject collection node and the workobjects it contains.
 xx050018	Target	A defined position and rotation for a robot. A target equals a RobTarget in a RAPID program.
 xx050019	Target without assigned configuration	A target for which no axis configuration has been assigned, for example, a repositioned target or a new target created by means other than teaching.
 xx050020	Target without found configuration	An unreachable target, that is, for which no axis configuration has been found.
 xx050021	Path Collection	Contains all paths in the station.
 xx050022	Path	Contains the instructions for the robot movements.
 xx050023	Linear Move Instruction	A linear TCP motion to a target. If the target has no valid configuration assigned, the move instruction gets the same warning symbols as the target.

*Continues on next page*

Icon	Node	Description
 xx050024	Joint Move Instruction	A joint motion to a target. If the target has no valid configuration assigned, the move instruction gets the same warning symbols as the target.
 xx050025	Action Instruction	Defines an action for the robot to perform at a specified location in a path.

# 1 Introduction to RobotStudio

---

## 1.4.4 The Modeling browser

### 1.4.4 The Modeling browser




---

#### Overview

The modeling browser is a display of editable objects and their building blocks.

---

#### Icons











Icon	Node	Description
 modeling	<b>Part</b>	Geometric items corresponding to the objects in the <b>Layout</b> browser.
 modelin0	<b>Body</b>	Geometric building blocks that comprise the parts. 3D bodies contain several faces, 2D bodies one face, and curves no faces.
 modelin1	<b>Face</b>	The faces of the bodies.

## 1.4.5 The Controller browser

### Overview

The Controller browser is a hierarchical display of controller and configuration elements found in the **Controller** tab view.

### Icons










Icon	Node	Description
 control	<b>Controllers</b>	Contains the controllers that are connected to the Robot View.
 control0	<b>Connected Controller</b>	Represents a controller with a working connection.
 control1	<b>Connecting Controller</b>	Represents a controller which is currently being connected.
 control2	<b>Disconnected Controller</b>	Represents a controller that has lost its connection. It might have been turned off or disconnected from the network.
 control3	<b>Denied login</b>	Represents a controller that denies you access to login. Possible reasons for denied access are: <ul style="list-style-type: none"> <li>• User lacks necessary access privileges</li> <li>• Too many clients connected to the controller</li> <li>• The RobotWare version of the system running on the controller is newer than the RobotStudio version</li> </ul>
 configu0	<b>Configuration</b>	Contains the configuration topics.
 configu1	<b>Topic</b>	Each parameter topic is represented by a node: <ul style="list-style-type: none"> <li>• Communication</li> <li>• Controller</li> <li>• I/O</li> <li>• Man-machine communication</li> <li>• Motion</li> </ul>
 eventrec	<b>Event Log</b>	With the Event Log you can view and save controller events.
 io	<b>I/O System</b>	Represents the controller I/O system. The I/O system consists of I/O buses and units.
 io-node	<b>I/O Bus</b>	An I/O bus is a connector for one or several I/O units.

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.4.5 The Controller browser

*Continued*



 io-devic	<b>I/O Unit</b>	An I/O Unit is a board, panel or any other device with ports through which the I/O signals are sent.
 rapid16t	<b>RAPID Tasks</b>	Contains the active tasks (programs) of the controller.
 prgintas	<b>Task</b>	A task is a robot program, which executes alone or together with other programs. A program is composed of a set of modules.
<u>Program Modules</u>	<b>Program Modules</b>	Program modules contain a set of data declarations and routines for a specific task. Program modules contains data specific for this program.
<u>System Modules</u>	<b>System Modules</b>	System modules contain a set of type definitions, data declarations and routines. System modules contains data that applies to the robot system, regardless which program modules that are loaded.
 nostepin	<b>Nostepin Module</b>	A module that cannot be entered during step-by-step execution. That is all instructions in the module are treated as one if the program is executed step-by-step.
 modules	<b>View-only and Read-only Program Modules</b>	A icon for program modules that are either view-only or read-only.
 module_e	<b>View-only and Read-only System Modules</b>	A icon for system modules that are either view-only or read-only.
 procedur	<b>Procedure</b>	A routine that does not return any value. Procedures are used as subprograms.
 function	<b>Function</b>	A routine that returns a value of a specific type.
 trap16tr	<b>Trap</b>	A routine that provides a means of responding to interrupts.

## 1.4.6 Files browser

### Overview

The Files browser in the RAPID tab allows you to manage RAPID files and system backups. Using the Files browser you can access and then edit standalone RAPID modules and system parameter files that are not residing in the controller memory.

### Icons

Icon	Node	Description
 xx1200000824	Files	See <a href="#">Managing RAPID files on page 426</a> .
 xx1200000825	Backups	See <a href="#">Managing system backups on page 426</a> .

# 1 Introduction to RobotStudio

---

## 1.4.7 Add-Ins browser

### 1.4.7 Add-Ins browser




---

#### Overview

The Add-Ins browser shows the installed PowerPacs, General add-ins , if available, under their respective nodes.

---

#### Icons

Icon	Node	Description
 xx1200000826	Add-In	Indicates an available add-in loaded into the system
 xx1200000827	Disabled Add-In	Indicates a disabled add-in
 xx1200000828	Unloaded Add-In	Indicates an add-in un-loaded from the system

## 1.4.8 The Output window

### Overview

The output window displays information about events that occur in the station, such as when simulations are started or stopped. This information is useful when troubleshooting stations.

### Layout of the Output tab

The **Output** tab contains two columns: the first states the event, the second the time the message was generated. Each row is a message.

### Event types

The three event types indicate the severity of the event:

Event type	Description
<b>Information</b>	An information message is a normal system event, for example, starting and stopping programs, changing the operational mode, and turning motors on and off. Information messages never require an action from you. They can be useful for tracking errors, collecting statistics or monitoring user-triggered event routines.
<b>Warning</b>	A warning is an event of which you should be aware, but it is not so severe that the process or RAPID program needs to be stopped. Warnings must occasionally be acknowledged. Warnings often indicate underlying problems that at some point will need to be resolved.
<b>Error</b>	An error is an event that prevents the robot system from proceeding. The running process or RAPID program cannot continue and will be stopped. An error must occasionally be acknowledged. Some errors require some immediate action from you in order to be resolved. Double-click an error to display a detailed information box.

Some of the events are active. These are linked to an action for resolving the problem that generated the event. To activate the linked action, double-click the message.

### Handling messages in the Output window

Goal	Procedure
To filter messages...	Right-click in the Output window and then click <b>Show messages</b> . From the options <b>All Errors</b> , <b>Information</b> , <b>Warnings and Warnings and Errors</b> , select the type of messages you want to display.
To save a message to file...	Select it, right-click and then click <b>Save to file</b> . Choose a name and location in the dialog box. Multiple messages can be selected by pressing <b>SHIFT</b> while clicking each.
To clear the Output window...	Right-click in the Output window and then click <b>Clear</b> .

# 1 Introduction to RobotStudio

## 1.4.9 The Controller Status window

### 1.4.9 The Controller Status window

#### Overview

The Controller Status window shows the operational status of the controllers in your robot view.

#### Layout of the Controller Status window

The Controller Status window has the following columns:

- 1 **System Name** : Displays the name of the system running on the controller.
- 2 **Controller Name** : Displays the name of the controller.
- 3 **Controller State** : Displays the state of the controller.

When the controller is in state...	the robot is...
Initializing	starting up. It will shift to state <i>motors off</i> when it has started.
Motors off	in a standby state where there is no power to the robot's motors. The state has to be shifted to motors on before the robot can move.
Motors on	ready to move, either by jogging or by running programs.
Guard Stop	stopped because the safety runchain is opened. For instance, a door to the robot's cell might be open.
Emergency Stop	stopped because emergency stop was activated.
Waiting for motors on after e-stop	ready to leave emergency stop state. The emergency stop is no longer activated, but the state transition isn't yet confirmed.
System Failure	in a system failure state. A warm start is required.

- 4 **Program Execution State** : Displays if the robot is running any program or not.

When the controller is in state...	the robot...
Running	is running a program.
Ready	has a program loaded and is ready to run it when a PP (starting point in the program) has been set.
Stopped	has a program loaded, with a PP, and is ready to run it.
Uninitialized	has not initialized the program memory. This indicates an error condition.

- 5 **Operating Mode** : Displays the operating mode of the controller.

When the controller is in mode...	the robot is...
Initializing	starting up. It will shift to the mode selected on the controllers cabinet when it has started.
Auto	ready to run programs in production.  In Auto mode it is possible to get remote Write access to the controller, which is necessary for editing programs, configurations and other things when connected to a real controller.

*Continues on next page*

When the controller is in mode...	the robot is...
Manual	only able to move if the enabling device on the FlexPendant is activated. Furthermore, the robot can only moved with reduced speed in manual mode.  In manual mode it is not possible to get remote Write access to the controller, unless it is configured for this and the remote Write access granted on the FlexPendant.
Manual full speed	only able to move if the enabling device on the FlexPendant is activated.  In manual mode it is not possible to get remote Write access to the controller, unless it is configured for this and the remote Write access granted on the FlexPendant.
Waiting for acknowledge	about to enter Auto mode, but the mode transition has not yet been acknowledged.

- 6 **Logged on as:** Displays the user name the PC is logged on to the controller with.
- 7 **Access :** Displays the users having write access to the controller, or if it is available.

### 1.4.10 The Operator Window

---

#### Overview

Operator Window is an alternative to the corresponding feature in the Virtual FlexPendant for communicating with the user during RAPID program execution. It displays the same output as displayed on the Virtual FlexPendant Operator Window.

When running in a Virtual Controller, the RAPID program communicates with the operator via messages on the FlexPendant screen. The Operator Window integrates this functionality and allows the user to run interactive RAPID programs without starting the Virtual FlexPendant.

---

#### Enabling Operator Window

To enable an operator window:

- 1 On the **File** menu, click **Options**.
- 2 On the Navigation pane to the left, select **Robotics:Virtual Controller**.
- 3 On the **Virtual Controller** page to the right, select **Automatically open virtual Operator Window**.
- 4 Click **Apply**.



#### Note

When the **Show virtual Operator Window** feature is enabled, an Operator Window is automatically created for each controller in the station. By default, the window is located in the tab area below the graphics window.

---

#### RAPID Instructions

The following are the RAPID instructions supported by the Operator Window. When these instructions are executed, the behavior is similar to that of Virtual FlexPendant:

- TPErase
- TPReadFK
- TPReadNum
- TPWrite
- UIAlphaEntry
- UIMsgBox
- UINumEntry

The following are the RAPID instructions not supported by the Operator Window. When these instructions are executed, an error message is displayed in the Operator Window prompting you to use the Virtual Flexpendant instead:

- TPShow
- UIShow
- UINumTune
- UIListView

*Continues on next page*



#### Note

You should not run both the Virtual Flexpendant and Operator Window simultaneously.

# 1 Introduction to RobotStudio

## 1.4.11 The Documents window

### 1.4.11 The Documents window

#### Overview

The Documents window allows you to search and browse the RobotStudio documents like libraries, geometry and so on in large numbers and from different locations. You can also add associate documents with a station, either as a link or by embedding a file in the station.

#### Opening a Documents window

- 1 On the **Home** tab, click **Import Library** and select **Documents** from the dropdown menu.

The **Documents** window appears.

#### Layout of the Documents window

The Documents window is a docked area that by default takes the right-hand corner. The upper part of the window contains controls for searching and browsing the document locations. The lower part consists of a list view that displays the documents and folders and a status area.

Control	Description
Station	Allows to add documents associated with the station, either by adding the file/folder as a reference (link) or by embedding the file in the station. See <a href="#">Using the Station mode on page 62</a> .
Search	Allows to search for keywords or query. See <a href="#">Using the Search mode on page 63</a> .
Browse	Displays a folder structure of the document locations. See <a href="#">Using the Browse mode on page 65</a> .
Location	Allows to configure the document location. See <a href="#">Document Locations window on page 67</a> .

#### Using the Station mode

Use this procedure to add documents associated with the current station:

- 1 Click **Station** from the Document Manager.
- 2 Click **Add** button and select what to add to the current station:
  - File Reference
  - Folder Reference
  - Embedded File
  - New Text Document



#### Note

- The referenced file/folder is displayed with an arrow icon
- The embedded file and new text document are displayed with a diskette icon

- 3 In the Documents window, right-click the document.

*Continues on next page*

The following context menu items appears depending on the document type selected:

Item	Description
Open	Opens the document in the program associated with its file type. For example, a .docx file is opened in Microsoft Word. An embedded file is saved to a temporary location before opening. If RobotStudio detects that the temporary file has changed, you will be asked to update the embedded file.
Open containing folder	Opens the folder containing the file in Windows Explorer. This option is not available in embedded files.
Copy to Station	Converts a referenced file to an embedded file.
Save as	Saves an embedded file to disk.
Include in Pack and Go	Specifies if a referenced file/folder should be included when a <i>Pack and Go</i> file is created. For a referenced folder, all files in the folder will be included. To use this option, the file must be located in the parent folder of the station file. For example, if the station file is <i>D:\Documents\Stations\My.rsstn</i> , the reference must be located in <i>D:\Documents</i> to be included in <i>Pack and Go</i> . Embedded files are always included when a <i>Pack and Go</i> file is created, since they are part of the station file.
Include Subfolders	Specifies that subfolders of a referenced folder should be included in <i>Pack and Go</i> .
Remove	Removes the selected document.



#### Note

Some context menu items can be disabled and the document marked as *Locked* in the API.

### Using the Search mode

- 1 Click the option **Search** and enter a query or syntax in the text box.

For more information about the available syntaxes, see [Search syntax on page 64](#).



#### Note

The drop-down list contains the search history of the previous ten queries between sessions.

- 2 Click the **Expand** button to access additional controls.

This allows you to specify if the search should cover all the enabled locations or one specific location.

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.4.11 The Documents window

*Continued*

- 3 Select **Search in results** check box to search for the resulting documents of the previous search.



### Note

The search is performed automatically, after you stop typing in the text box or manually, by clicking the glass icon. During the search, this icon is changed to a cross and clicking this cancels the search operation.

## Search syntax

The search field supports certain keywords and operators which allows you to specify an advanced search query.



### Note

Keywords are not localised.

The following table displays the keywords that specifies an advanced search query:

Keywords	Description
filename	matches the filename of the documents.
title	matches the title field of the document metadata.
type	matches the type field of the document metadata. For library (.rslib) files, this is a user-defined string. For example, Robot. For other files, this is the Windows description of the file type. For example, Text Document.
author	matches the author field of the document metadata.
comments	matches the comment field of the document metadata.
revision	matches the revision field of the document metadata.
date	matches the last time the file was modified. For the colon operator, the match is done against a string representation of the modified date. For other operators, the search string should interpret as a date according to .NET standards.
size	matches the file size (in KB).
and, or, parentheses (), not	used to group or invert queries.

The following table displays the operators that specifies an advanced search query

Operator	Description
:	matches if the field contains the search string.
=	matches if the field equals the search string.
<	matches if the field is smaller than the search string.
>	matches if the field is greater than the search string.

*Continues on next page*



### Note

- Quotation marks can be used to specify a string with whitespace. An empty string is specified by "".
- All search strings are case insensitive.
- Text without a preceding keyword is matched against the filename and all metadata.
- If queries are specified without a grouping keyword, "and" is implied.
- Some metadata (title, author, comments and revision) is not available for all file types.

### Examples

- **1400** - Matches documents with the string 1400 in the filename or any metadata.
- **not author:ABB** - Matches documents where the author field does not contain the string ABB.
- **size>1000 and date<1/2009** - Matches documents larger than 1000KB and modified before 1/1/2009.
- **IRBP comments="ABB Internal"** - Matches documents with the string IRBP in the filename or any metadata, and where the comment field equals ABB Internal.

### Using the Browse mode

- 1 Click the option **Browse** from the Document Manager.  
A folder structure of the document location is displayed.



### Note

The top level of the folder structure lists the configured locations. If a location is unavailable (for example, an offline network path) it is marked as **Unavailable** and cannot be opened. The text box displays the path of the current folder relative to the location root.

- 2 You can open a folder in either of the two ways:
  - Double-click the document location.
  - Right-click the document location and select **Open** from the context menu.
- 3 You can navigate through the folders in either of the two ways:
  - Click the folder icon at the top-right corner.
  - Select the parent folder from the dropdown list.



### Note

You can browse and add component xml files (\*.rsxml) to your station.

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.4.11 The Documents window

*Continued*

- 4 Click **Refresh** icon in the text box to manually refresh the contents of the folder.



### Note

The refresh operation can be time consuming, if a folder resides in a network location or contains many documents. During this time, the refresh icon changes to a cross icon allowing you to cancel the operation.

## Results view

In the Browse mode, items are grouped into folders and documents. The resulting folders and documents are displayed in a list view.

The search result appears in the status bar at the bottom, displaying the number of items found, and progress made during the search. The Search results are grouped beneath headers according to their location.

Each document is represented by an image, the document title or filename in black text, and the metadata and file information in grey text. For library files, the image can be a screenshot or other custom image. For other document types, the image is the icon associated with the file type.

## Using the Context menu

In the results view, right-click a document or folder. The following context menu items appears:

Item	Description
Open	This command opens the selected folder, library or geometry files, station files, and document. <ul style="list-style-type: none"><li>• For folders, browses into the selected folder.</li><li>• For library or geometry files, imports the file into the station. (If no station is open ,a new empty station is first created.)</li><li>• For station files, opens the station.</li><li>• For other documents, attempts to open the selected document according to its file association. For example, Microsoft Word starts when a .doc file is opened.</li></ul>
Open containing folder	This command opens the folder containing the document or the folder in the Windows Explorer.
Properties	This command is disabled for folders. This command opens a dialog that displays complete metadata and file information about the selected document.



### Tip

Double-click an item to import the library and geometry files and open the other documents.

*Continues on next page*

In the results view, right-click an empty area. The following context menu appears that controls how the documents are grouped and sorted:

Items	Description
Group by:	Controls how the documents are arranged in groups. The following options are available: <ul style="list-style-type: none"> <li>• Location</li> <li>• Folder</li> <li>• Type</li> </ul>
Sort by:	Controls how the documents are sorted within the group. The following options are available: <ul style="list-style-type: none"> <li>• Name</li> <li>• Date</li> <li>• Size</li> </ul>
Ascending and Descending	Items are sorted in the ascending and descending order.

#### Using the drag and drop feature

You can import a library or a geometry file into the station by dragging it from the results view into either the graphics window, or onto a object node in the Layout browser.

- When dragging into the Layout browser, the component will be placed as a child object under the station, component group or smart component.
- When dragging into the graphics window, the component will be positioned at the point on the station floor where you drop it. You can snap the point to the UCS grid by enabling the **Snap Grid** or by holding down the ALT key while dragging.

#### Document Locations window

You can launch the Document Locations window in any one of the following ways:

- 1 Select **Locations** from the **Documents** window.
- 2 On the **File** menu, click **Options** and select **Files & Folders** in the navigation pane. Click **Document Locations** on the right side.
- 3 On the **Home** tab, click **Import Library** and select **Locations** from the dropdown menu.

#### Layout of the Document Locations window

It consists of a menu bar and a list displaying the configured locations. The list displays general information about the locations. The menu bar contains the following controls:

Controls	Description
Locations	The following options are available from the dropdown menu: <ul style="list-style-type: none"> <li>• <b>Import</b> : Opens a dialog box to import document locations from an xml file. If a location with the same URL already exists, you have the option to retain or delete the existing location.</li> <li>• <b>Export</b> : Opens a dialog box to export all the configured locations to an xml file.</li> <li>• <b>Reset to Default</b> : Loads the default locations (ABB Library, User Library, and User Geometry).</li> </ul>

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.4.11 The Documents window

*Continued*

Controls	Description
Add Location	Opens a dialog box to add a document location. By default, there is one location type available. For more information, see <a href="#">File System location on page 68</a> .
Remove	Deletes the selected location.
Edit	Opens a dialog box to modify the selected location. For more information, see <a href="#">File System location on page 68</a> .

### File System location

- 1 Click **Add Locations** and select **File System** from the dropdown menu. The **File System** dialog box appears.













The File System dialog box contains the following controls:

Control	Description
Location Name	Specifies a name associated with the location.
Path	Specifies the file system directory that corresponds to the root folder of the location. This can be on a local or network disk.
Filter	Specifies a filename filter to include only certain files when searching and browsing. Multiple filters are separated by a semicolon. If the filter is empty all files are included.
Cache files from network	Specifies that the library and geometry files from a network location should be copied to a local directory and imported from there, rather than directly from the network path. This will ensure that a station containing such files can be opened even if the network location is unavailable. This option is only available for network locations.
Directory	Specifies the directory where to store the local copies. This must be on a local disk.
Show as gallery	Specifies that the contents of the location should be displayed as a gallery in the specified ribbon menu.
Style	<ul style="list-style-type: none"><li>• Flat - Specifies that all documents are displayed in a single gallery with subfolder names as headers.</li><li>• Recursive - Specifies that the documents are displayed in submenus corresponding to the folder structure.</li></ul>
Include when searching all locations	Specifies if the search should cover all the enabled locations.

## 1.4.12 Using a mouse

## Navigating the graphics window using the mouse

The table below shows how to navigate the graphics window using the mouse:

To	Use the keyboard /mouse combination	Description
<b>Select items</b>  selectio	 left-cli	Just click the item to select. To select multiple items, press CTRL key while clicking new items.
<b>Rotate the station</b>  rotate	<b>CTRL + SHIFT +</b>  left-cli	Press CTRL + SHIFT + the left mouse button while dragging the mouse to rotate the station.  With a 3-button mouse you can use the middle and right buttons, instead of the keyboard combination.
<b>Pan the station</b>  pan	<b>CTRL +</b>  left-cli	Press CTRL + the left mouse button while dragging the mouse to pan the station.
<b>Zoom the station</b>  zoom	<b>CTRL +</b>  right-cl	Press CTRL + the right mouse button while dragging the mouse to the left to zoom out. Dragging to the right zooms in.  With a 3-button mouse you can also use the middle button, instead of the keyboard combination.
<b>Zoom using window</b>  window_z	<b>SHIFT +</b>  right-cl	Press SHIFT + the right mouse button while dragging the mouse across the area to zoom into.
<b>Select using window</b>  window_s	<b>SHIFT +</b>  left-cli	Press SHIFT + the left mouse button while dragging the mouse across the area to select all items that match the current selection level.

### 1.4.13 Selecting an item

---

#### Overview

Each item in a station can be moved to achieve the required layout, so you first have to determine its selection level. The selection level makes it possible to select only specific types of items, or specified parts of objects.

The selection levels are curve, surface, entity, part, mechanism, group, target/frame and path. The target/frame and path selection can be combined with any of the other selection levels.

Objects may also be grouped together as component groups, see [Component Group on page 262](#).

---

#### Selecting an item in the graphics window

To select items in the graphics window, follow these steps:

- 1 At the top of the graphics window, click the desired selection level icon.
- 2 Optionally, click the desired snap mode icon for the part of the item you wish to select.
- 3 In the graphics window, click the item. The selected item will be highlighted.

---

#### Multiple selection of items in the graphics window

To select multiple items in the graphics window, do the following:

- 1 Press the **SHIFT** key, and in the graphics window drag the mouse diagonally over the objects to select.

---

#### Selecting an item in the browsers

To select items in a browser, do the following:

- 1 Click the item. The selected item will be highlighted in the browser.

---

#### Multiple selection of items in the browsers

To select multiple items in a browser, follow these steps:

- 1 Make sure that all the items to be selected are of the same type and located in the same branch of the hierarchical structure; otherwise, the items will not be operable.
- 2 Do one of the following:
  - To select adjacent items: In the browser, hold down the **SHIFT** key and click the first and then the last item. The list of items will be highlighted.
  - To select separate items: In the browser, hold down the **CTRL** key and click the items you want to select. The items will be highlighted.

#### 1.4.14 Attaching and detaching objects

---

##### Overview

You can attach an object (child) to another object (parent). Attachments can be created on part level and on mechanism level. When an object has been attached to a parent, moving the parent also moves the child.

One of the most common attachments is to attach a tool to a robot. For procedures, see [Attach to on page 450](#) and [Detach on page 458](#).

# 1 Introduction to RobotStudio

## 1.4.15 Keyboard shortcuts

### 1.4.15 Keyboard shortcuts

#### General keyboard shortcuts

The following table lists general keyboard shortcuts in RobotStudio.

Command	Key Combination
<b>General Shortcuts</b>	
Activate menu bar	F10
Open API Help	ALT + F1
Open Help	F1
Open Virtual FlexPendant	CTRL + F5
Switch between windows	CTRL + TAB
<b>General Commands</b>	
Add Controller System	F4
Open Station	CTRL + O
Take Screenshot	CTRL + B
Teach Move Instruction	CTRL + SHIFT + R
Teach Target	CTRL + R
Import Geometry	CTRL + G
Import Library	CTRL + J
New Empty Station	CTRL + N
Save Station	CTRL + S
<b>General Editing Commands</b>	
Copy	CTRL + C
Cut	CTRL + X
Paste	CTRL + V
Delete	DELETE
Redo	CTRL + Y
Refresh	F5
Rename	F2
Select All	CTRL + A
Undo	CTRL + Z

#### RAPID Editor shortcuts

The following table lists the keyboard shortcuts specific to the RAPID Editor.

Command	Key Combination
<b>RAPID Editor Intellisense</b>	
Complete Word	CTRL + SPACEBAR
Parameter Info	CTRL + SHIFT + SPACEBAR
Auto Complete	TAB (when the cursor located at the end of an identifier)

*Continues on next page*

Command	Key Combination
<b>General RAPID Editor Commands</b>	
Start Program Execution	F8
Step In	F11
Step Out	SHIFT + F11
Step Over	F12
Stop	SHIFT + F8
Toggle Breakpoint	F9
Apply Changes	CTRL + SHIFT + S
Print	CTRL + P
<b>RAPID Editor Text Commands</b>	
Copy	CTRL + Insert or, CTRL + C
Cut	SHIFT + Delete or, CTRL + X
Cut line	CTRL + L
Delete line	CTRL + SHIFT + L
Delete to beginning of word	CTRL + BACKSPACE
Delete to end of word	CTRL + Delete
Find next occurrence	F3
Indent	Tab
Make the selected text lowercase	CTRL + U
Make the selected text uppercase	CTRL + SHIFT + U
Move to beginning of document	CTRL + Home
Move to beginning of line	Home
Move to end of document	CTRL + End
Move to end of line	End
Move to next word	CTRL + Right
Move to previous word	CTRL + Left
Move to visible bottom	CTRL + Page Down
Move to visible top	CTRL + Page Up
Open line above	CTRL + Enter
Open line below	CTRL + SHIFT + Enter
Outdent	SHIFT + TAB
Paste	SHIFT + Insert or, CTRL + V
Redo	CTRL + SHIFT + Z or, CTRL + Y
Scroll down	CTRL + Down
Scroll up	CTRL + Up

*Continues on next page*

# 1 Introduction to RobotStudio

## 1.4.15 Keyboard shortcuts

*Continued*

Command	Key Combination
Select block down	ALT + SHIFT + Down
Select block left	ALT + SHIFT + Left
Select block right	ALT + SHIFT + Right
Select block up	ALT + SHIFT + Up
Select down	SHIFT + Down
Select left	SHIFT + Left
Select page down	SHIFT + Page Down
Select page up	SHIFT + Page Up
Select right	SHIFT + Right
Select to beginning of document	CTRL + SHIFT + Home
Select to beginning of line	SHIFT + Home
Select to end of document	CTRL + SHIFT + End
Select to end of line	SHIFT + End
Select to next word	CTRL + SHIFT + Right
Select to previous word	CTRL + SHIFT + Left
Select to visible bottom	CTRL + SHIFT + Page Down
Select to visible top	CTRL + SHIFT + Page Up
Select up	SHIFT + Up
Select word	CTRL + SHIFT + W
Toggle overwrite mode	Insert
Transpose characters	CTRL + T
Transpose lines	CTRL + ALT + SHIFT + T
Transpose words	CTRL + SHIFT + T



### Note

When the user presses the ALT key, shortcut keys are displayed on the RobotStudio ribbon. Use these shortcut keys with ALT key to access the corresponding menu item.

## 2 Building stations

### 2.1 Workflow of building a station

#### Overview

The following sections outline the workflow for building a new station. It also includes the prerequisites for creating and simulating robot programs. The workflow includes:

- Options for creating a station with a system.
- Importing or creating the objects to work with
- Optimizing the station layout by finding the best placement of robots and other equipment.



#### Note

For most scenarios, you are recommended to follow the workflows from start to finish, even though other sequences maybe possible.

#### Creating a station with a system

The table below shows the options for creating a station with a system.

For the exact procedures, see [New on page 190](#).

Activity	Description
Create a station with a template system	This is the simplest way to create a new station containing a robot and a link to a rudimentary system template.
Create a station with an existing system	This creates a new station containing one or more robots in accordance with an existing, built system.
Create a station with no system	An advanced user can build a station from scratch and then add a new or existing system to it.

#### Manually starting the VC

The table below shows the alternatives for manually starting with a system. Perform only those steps applicable to your station.

Activity	Description
Manually connecting a library to the VC	See <a href="#">Starting a VC on page 85</a> .
Restarting the VC	See <a href="#">Restarting a VC on page 86</a> .

#### Importing station components

The table below shows the workflow for importing station components. Perform only those steps applicable to your station.

For procedures, see [Importing a station component on page 87](#).

Activity	Description
Import a robot model	See <a href="#">Robot System on page 206</a> .
Import a tool	See <a href="#">Import Library on page 205</a> .

*Continues on next page*

## 2 Building stations

### 2.1 Workflow of building a station

*Continued*

Activity	Description
Import a positioner	See <a href="#">ABB Library on page 204</a> .
Import a track	See <a href="#">Import Library on page 205</a> .
Import other equipment	If you have CAD models of the equipment, you can import them, see <a href="#">Import Library on page 205</a> . Otherwise, you can create models in RobotStudio, see <a href="#">Mechanisms on page 94</a> .
Add work piece	If you have CAD models of the work piece, you can import them, see <a href="#">Workobject on page 216</a> . Otherwise, you can create models in RobotStudio, see <a href="#">Objects on page 92</a> .

### Placing objects and mechanisms

The table below shows the workflow for placing the objects in the station.

Activity	Description
Place objects	If you are building a model of a real station, start by placing all objects with known positions. For objects without known positions, find a suitable placement, see <a href="#">Placing objects on page 97</a> and <a href="#">Placing external axes on page 98</a> .
Attach tools	Attach the tools to the robot, see <a href="#">Attach to on page 450</a> .
Attach robots to tracks	If track external axes are used, attach the robots to the tracks, see <a href="#">Attach to on page 450</a> .
Attach work pieces to positioners	If positioner external axes are used, attach the work pieces to the positioners, see <a href="#">Attach to on page 450</a> .
Test reachability	Test if the robot can reach critical positions on the work piece. If you are satisfied with how the robot reaches the positions, your station is ready for programming. Otherwise, continue adjusting the placement or trying other equipment as described below, see <a href="#">Testing positions and motions on page 122</a> .

## 2.2 Conveyor tracking station with two robots

### 2.2.1 Two robot systems in same task frame position

---

#### Overview

This section describes what happens when both the robot systems share the same task frame position. The baseframes of the mechanical units in both the robot systems have the same task frame position.

---

#### Prerequisites

- Two robot systems with conveyor tracking option (system 1 and system 2)
- A conveyor mechanism saved as library

See [Create Conveyor mechanism on page 317](#) to create conveyor tracking systems.

---

#### Setting up the conveyor tracking station

- 1 Add the existing system (system 1) to the station. See [Robot System on page 206](#).

**Note**

After starting the system, when asked to select the library, browse and select the already saved conveyor mechanism library.

- 2 Modify the baseframe positions of conveyor and robot.
  - a Move the mechanical unit (conveyor/robot) to its new location.
  - b See [Updating the baseframe position on page 409](#) to update the baseframe position of the conveyor/robot.
  - c Repeat steps 1 and 2 to modify the baseframe position of the robot.
  - d In the **System Configuration** window, click **OK**. When asked if you want to restart the system, answer **Yes**. Close the **System Configuration** window.
- 3 Add the existing system (system 2) to the station. See [Robot System on page 206](#).

**Note**

After starting the system, when asked to select the library, browse and select the same library as the one selected for system 1 or any other library. Later, this conveyor library will be removed from the station since system 2 shall use the same conveyor library as system 1.

- 4 Refer both systems (system 1 and system 2) to the same conveyor library.
  - a On the **Controller** tab, in the **Virtual Controller** group, click **Edit System**. This opens the **System Configuration** dialog for system 2.
  - b Select the library node in the hierarchical tree.

*Continues on next page*

## 2 Building stations

---

### 2.2.1 Two robot systems in same task frame position

*Continued*

- c Select the option **Select from Station**. Click **Change**. The **Select Library** dialog box appears.
- d Select the same conveyor library as the one selected for system 1. Click **OK**.



#### Note

Now both systems (system 1 and system 2) use the same conveyor library and the library previously referenced by system 2 is removed from the station.

- 5 Modify the baseframe positions of robot (system 2).
  - a Move the mechanical unit (robot) to its new location.
  - b See [Updating the baseframe position on page 409](#) to update the baseframe position of the robot.
  - c Repeat steps 1 and 2 to modify the baseframe position of the robot.
  - d In the **System Configuration** window, click **OK**. When asked if you want to restart the system, answer **Yes**. Close the **System Configuration** window.

## 2.2.2 Two robot systems in different task frame positions

### Overview

This section describes what happens when two robot systems have different task frame positions but uses the same sync switch. This means the Baseframes of the conveyor mechanical units in both the robot systems have different values.

### Prerequisites

Two robot systems with conveyor tracking option (system 1 and system 2)

See [Create Conveyor mechanism on page 317](#) to create conveyor tracking systems.

### Setting up the conveyor tracking station

- 1 Add the existing system (system 1) to the station. See [Robot System on page 206](#).



#### Note

After starting the system, when asked to select the library, browse and select the already saved conveyor mechanism library.

- 2 Modify the baseframe positions of conveyor and robot.
  - a Move the mechanical unit (conveyor/robot) to its new location.
  - b See [Updating the baseframe position on page 409](#) to update the baseframe position of the conveyor/robot.
  - c Repeat steps 1 and 2 to modify the baseframe position of the robot.
  - d In the **System Configuration** window, click **OK**. When asked if you want to restart the system, answer **Yes**. Close the **System Configuration** window.
- 3 Add the existing system (system 2) to the station. See [Robot System on page 206](#).



#### Note

After starting the system, when asked to select the library, browse and select the same library as the one selected for system 1 or any other library. Later, this conveyor library will be removed from the station since system 2 shall use the same conveyor library as system 1.

- 4 Update both systems (system 1 and system 2) to use the same conveyor library.
  - a On the **Controller** tab, in the **Virtual Controller** group, click **Edit System**. This opens the **System Configuration** dialog for system 2.
  - b Select the library node in the hierarchical tree.
  - c Select the option **Select from Station**. Click **Change**. The **Select Library** dialog box appears.

*Continues on next page*

## 2 Building stations

### 2.2.2 Two robot systems in different task frame positions

*Continued*

- d Select the same conveyor library as the one selected for system 1. Click **OK**.



#### Note

Now both systems (system 1 and system 2) use the same conveyor library and the library previously referenced by system 2 is removed from the station.

- 5 Modify the task frame position of the conveyor mechanism. See [Set Task Frames on page 408](#).



#### Note

Before modifying the task frame, make a note of the current conveyor position in world coordinates. After modifying the task frame, move the conveyor back to the position it was before modifying the task frame.

- 6 Modify the baseframe positions of robot (system 2).  
Repeat step 2 to modify the baseframe position of the robot (system 2)
  - a Move the mechanical unit (robot) to its new location.
  - b See [Updating the baseframe position on page 409](#) to update the baseframe position of the robot.
  - c Repeat steps 1 and 2 to modify the baseframe position of the robot.
  - d In the **System Configuration** window, click **OK**. When asked if you want to restart the system, answer **Yes**. Close the **System Configuration** window.
- 7 Modify the baseframe position of the conveyor (system 2).
  - a On the **Controller** tab, in the **Virtual Controller** group, click **Edit System**. This opens the **System Configuration** dialog for system 2.
  - b Select the conveyor in the hierarchical tree. The **BaseFrame** property list for the conveyor is now displayed.
  - c Select the option **Use Current Station Values** to update the baseframe value of the robot in the controller.
  - d Deselect the option **Check BaseFrame on Startup**.
  - e In the **System Configuration** window, click **OK**. When asked if you want to restart the system, answer **Yes**.



#### Note

By deselecting the option **Check BaseFrame on Startup**, RobotStudio will not compare the **BaseFrame** values in the station and the controller every time the controller is started. This avoids repositioning the conveyor library. If the two robot systems use the same part on the conveyor, the relation between the part and the two conveyor workobjects should be the same.

## 2.3 Creating a system with external axes automatically

### Automatically create a system with external axes

- 1 Import the desired robots, positioners, and track libraries into the RobotStudio station. See [Import Library on page 205](#).

If a robot and track are selected, attach the robot to the track. See [Attach to on page 450](#).



#### Note

Robot system supports the following tracks with lengths 1.7 m to 19.7 m in a separate task or same robot task. Depending on the manipulator type, the system allows one to three tracks per task. However with IRBTx004, only one track of this type can be used per system.

- IRBT4003
- IRBT4004
- IRBT6003
- IRBT6004
- IRBT7003
- IRBT7004
- RTT\_Bobin
- RTT\_Marathon
- Paint Rail

- 2 Create a robot system from layout. See [Robot System on page 206](#).



#### Note

To create a robot system with IRBT4004, IRBT6004, or IRBT7004, the TrackMotion mediapool must be installed. For more information, see [Installing and licensing RobotStudio on page 40](#).

### Supported external axes configuration

The following table shows a combination of different external axes configurations:

Combination	Positioner type							
	A	B	C	D	K	L	2xL	R
One IRB (Positioner in same task)	Y	Y	Y	Y	Y	Y	Y	Y
One IRB (Positioner in separate task)	Y	Y	Y	Y	Y	Y	Y	Y
Two IRB (Positioner in separate task)	Y	Y	Y	Y	Y	Y	N	Y
One IRB on Track Motion (Positioner in same task)	Y	N	N	N	YX	Y	Y	N

*Continues on next page*

## 2 Building stations

### 2.3 Creating a system with external axes automatically

*Continued*

Combination	Positioner type							
	A	B	C	D	K	L	2xL	R
One IRB on Track Motion (Positioner in separate task)	N	N	N	N	N	N	Y	N

- Y - Combination is supported
- N - Combination is not supported
- YX - Combination is supported and manual mapping of mechanical units and joints required



#### Note

Creating a system from layout only supports tracks of type RTT and IRBTx003 in combination with positioners. i.e. IRBTx004 is not supported in combination with the positioners.

### Manual mapping of mechanical units and joints

If the system contains more than one mechanical unit, the number of tasks and base frame positions of the mechanism should be verified in the System Configuration.

- 1 On the **Controller** tab, in the **Virtual Controller** group, click **Edit System**. This opens the **System Configuration** dialog.
- 2 Select the robot from the node in the hierarchical tree.  
The property page of this node contains controls for mapping and setting axes and joints.
- 3 Click **Change** to open a dialog box.
- 4 Manually map the mechanical unit and mechanism joints. Click **Apply**.
- 5 Modify the baseframe positions of the mechanical unit. See [Updating the baseframe position on page 409](#).

## 2.4 Manually setting up system with track motion

### 2.4.1 Track motion of type RTT or IRBTx003

#### Manually setting up a system with track motion of type RTT or IRBTx003

Use this procedure to manually set up a system with track motion type RTT Bobin, RTT Marathon or IRBT4003, IRBT6003, or IRBT7003.

- 1 Build and start a new system. See [Building a new system on page 161](#).

	Action	Description
1	Select the desired robot variant (IRB6600).	In the <b>New Controller System</b> wizard of the <b>System Builder</b> , navigate to <b>Modify Options</b> page and scroll down to <b>Drive Module 1 &gt; Drive module application</b> group and expand <b>ABB Standard manipulator</b> option and select <b>Manipulator type (IRB6600)</b> .
2	Select the Additional axes configuration.	In the <b>New Controller System</b> wizard of the <b>System Builder</b> , navigate to <b>Modify Options</b> page of the <b>System Builder</b> , scroll down to <b>Drive Module 1&gt; Additional axes configuration</b> group and expand the <b>Add axes IRB/drive module 6600</b> option and select the <b>770-4 Drive W in pos Y2</b> option.  The option <b>770-4 Drive W in pos Y2</b> , the <b>Drive module</b> , and the <b>Position</b> varies depending on the <b>Additional axes configuration</b> selected. Make sure to select at least one drive in any position.
3	Click <b>Finish</b> .	Close the <b>Modify Options</b> page.

- 2 Add the system to the station. See [Adding a system on page 85](#)
- 3 Add the corresponding track configuration file of the desired robot variant (IRB 6600) and the desired track model to the station. See [Adding the track to the system on page 87](#).



#### Note

In the **Select Library** group, select either the existing track or import a different track.

The system may fail unless the correct additional axes configuration is selected.

- 4 Specify whether the baseframe is moved by another mechanism.
  - a On the **Controller** tab, in the **Virtual Controller** group, click **Edit System**. This opens the **System Configuration** dialog.
  - b Select **ROB\_1** node from the hierarchical tree.
  - c Select the option **Track** from the **BaseFrame moved by** list.
  - d Click **OK**. When asked if you want to restart the system, answer **Yes**. Close the **System Configuration** window.

## 2 Building stations

### 2.4.2 Track motion of type IRBTx004

### 2.4.2 Track motion of type IRBTx004

#### Overview

For configuration of tracks of type IRBT4004, IRBT6004, or IRBT7004, the TrackMotion mediapool must be installed. For more information, see [Installing and licensing RobotStudio on page 40](#).

#### Manually setting up a system with track motion of type IRBTx004

- 1 Build and start a new system. See [Building a new system on page 161](#).

	Action	Description
1	Add additional options for IRB-Tx004.	See <a href="#">Adding additional options on page 162</a> . Browse and select the key file (.kxt) located in the mediapool Track 5.XX.YYYY where 5.XX indicates the latest RobotWare version being used.
2	Select the desired robot variant (IRB6600).	On the <b>Modify Options</b> page of the <b>System Builder</b> , scroll down to <b>Drive Module 1 &gt; Drive module application</b> group and expand <b>ABB Standard manipulator</b> option and select <b>Manipulator type (IRB6600)</b> .
3	Select <b>Additional axes configuration</b> .	On the <b>Modify Options</b> page of the <b>System Builder</b> , scroll down to <b>Drive Module 1 &gt; Additional axes configuration</b> group and expand the <b>Add axes IRB/drive module 6600</b> option and select the <b>770-4 Drive W in pos Y2</b> option. The option <b>770-4 Drive W in pos Y2</b> , the <b>Drive module</b> , and the <b>Position</b> varies depending on the <b>Additional axes configuration</b> selected. Make sure to select at least one drive in any position.
4	Select the desired track motion (IRBT 6004).	On the <b>Modify Options</b> page of the <b>System Builder</b> , scroll down to the <b>TRACK</b> and expand the <b>Drive module for Track motion</b> group. Select <b>Drive Module 1 &gt; Track Motion type &gt; IRBT 6004 &gt; Irb Orientation on Track &gt; Standard carriage In Line &gt; Select Track Motion Length &gt; 1.7m (or any other variant)</b> .
5	Click <b>Finish</b> .	Close the <b>Modify Options</b> page.

- 2 Add the system to the station. See [Adding a system on page 85](#).
- 3 Add the desired track model to the station using the following procedure. See [Adding the track to the system on page 87](#).
  - a In the **Select Library** group, click **Other** to import a different track motion library.
  - b Click **OK**. When asked if you want to restart the system, answer **Yes**. Close the **System Configuration** window.

## 2.5 Virtual Controller

### 2.5.1 Starting a VC

#### Overview

RobotStudio uses virtual controllers for running the robots. Virtual controllers can run both systems for real robots and specific virtual systems for testing and evaluation purposes. A virtual controller uses the same software as the controller to execute the RAPID program, to calculate robot motions and to handle I/O signals. When starting a virtual controller, point out which system to run on it. Since the system contains information about the robots to use and important data such as robot programs and configurations, it is important to select the right system for the station.



#### Note

You can start and stop a virtual controller, using a given system path and without needing a station. For more information, see [Start Virtual Controller on page 359](#).

#### Starting a VC

The table below describes the different ways a virtual controller may start:

Startup	Description
Automatic, when creating a station	In most cases, a VC is automatically started when you create a new station. Library files for the robots used by the system are then imported to the station.
Automatic, when adding a system to an existing station	If your station uses several systems or if you started with an empty station, you can add systems to an open station. Library files for the robots used by the the systems are then imported to the station.
Manually, when connecting to an imported library	If you have manually imported a robot library you want to use with a system, instead of importing a new library at startup, you can connect this library to a controller. If you have manually imported a robot library you want to use with a system, instead of importing a new library at startup, you can connect this library to a controller. A library may only be connected to a single-robot system and may not be already connected to another VC.
Manually, when starting a controller from the Controller tab.	The Start Virtual Controller commands allows you start and stop a virtual controller, using a given system path and without needing a station.

#### Adding a system

To add a system to a new station, see [New on page 190](#).

To add a system to an existing station, see [Robot System on page 206](#).

For information how to create a system with specific options, see [The System Builder on page 158](#).

To start or add a virtual controller which is not part of a station, see [Add Controller on page 358](#).

## 2 Building stations

---

### 2.5.2 Restarting a VC

### 2.5.2 Restarting a VC

---

For information on when and how to restart a VC in RobotStudio, see [Restart a controller on page 365](#).

## 2.6 Station components

### 2.6.1 Importing a station component

#### Importing a robot model

This is how to import a robot model without a controller to your station.

A robot which is not connected to a controller cannot be programed. To import a robot connected to a virtual controller, configure a system for the robot and start it in a virtual controller, see [Building a new system on page 161](#) and [Starting a VC on page 85](#), respectively.

To import a robot model, in the **Home** tab, click **Robot System** and then select a robot model from the gallery.

#### Importing a tool

A tool is a special object, for example, an arc weld gun or a gripper, that operates on the work piece. For achieving correct motions in robot programs, the parameters of the tool have to be specified in the tool data. The most essential part of the tool data is the TCP, which is the position of the tool center point relative to the wrist of the robot (which is the same as the default tool, *tool0*).

When imported, the tool will not be related to the robot. So in order for the tool to move with the robot, you must attach it to the robot.

To import a tool, in the **Home** tab, click **Tool** and then select a tool from the gallery.

#### Importing a positioner

To import a tool, in the **Home** tab, click **Positioner** and then select a positioner from the gallery.

#### Adding the track to the system

To select the model of the external axis to use, follow these steps:



#### Note

This procedure is not applicable for a robot system with track motions IRBT4004, IRBT6004, or IRBT7004. They are configured by the TrackMotion mediapool and not by adding separate configuration files. For information on installation instructions, see [Installing and licensing RobotStudio on page 40](#).

- 1 Start the system in a virtual controller, either in a new empty station or in an existing station, see [Robot System on page 206](#).
- 2 In the **Layout** browser, select the system to add the track to.
- 3 On the **Controller** tab, click **System Configuration**.
- 4 Click **Add** to add parameters for the track to the system. Browse to the parameter file (.cfg) for the track to add and click **Open**.

If you have a specific parameter file for you track, use that one. Otherwise, parameter files for some standard tracks are delivered with the RobotStudio installation. These can be found in the folder *ABB Library/ Tracks* in

*Continues on next page*

## 2 Building stations

---

### 2.6.1 Importing a station component

*Continued*

RobotStudio's installation folder. The *ABB Library* folder can also be opened from the Quick access pane at the left of the Open dialog box used for adding parameter files.

The file name of each parameter file tells which tracks it supports. The first part tells the length of the track and the second the number of tasks.

For example, the file TRACK\_1\_7.cfg supports all tracks with the length 1.7 meters in systems with one single task. For Multimove systems or other systems with several tasks, use the configuration file with the matching number of tasks.

For example, if the track length is 19.9 m and the robot attached to the track is connected to task 4 of the MultiMove system, then select TRACK\_19\_9\_Task4.cfg file.

- 5 In the **System Configuration** window, click **OK**. When asked if you want to restart the system, answer **Yes**.
- 6 During the restart a list of all tracks compatible with the configuration file is displayed. Select the one to use and click **OK**.

After the restart the track appears in the station. Continue with attaching the robot to the track.

---

### Importing a library, geometry or piece of equipment

A library component is a RobotStudio object that has been saved separately. Normally, components in a library are locked for editing.

A geometry is CAD data which you can import to use in RobotStudio. For a list of importable CAD formats, see [Libraries, geometries and CAD files on page 37](#).

To import a library, geometry or piece of equipment, see [Import Library on page 205](#).

## 2.6.2 Converting CAD formats

### Overview

A CAD converter is installed together with RobotStudio by default. In most cases you do not have to convert CAD files before importing them to RobotStudio, but the CAD converter might be useful for converting several files at once, or for converting with custom settings.

### Prerequisites

Most of the file formats require separate licenses, see [Libraries, geometries and CAD files on page 37](#) for more information.

### Starting the CAD converter

Click **Start** menu, point to **Programs, ABB Industrial IT, Robotics IT, RobotStudio 5.xx** and click **CAD Converter**.

### Converting CAD files

To convert CAD files, follow these steps:

- 1 Click **Add files** and select the files to convert. Optionally, click **Add files**, again to add more files from another location.  
Each file is now added to a row in the grid.
- 2 Optionally, change the suggested file name or target format by clicking that column for the file to change.
- 3 In the **Target directory** box, specify the folder in which to save the new files.
- 4 Optionally, click **Settings** and change the settings for the conversion. For details about the conversion settings, see [Conversion settings on page 89](#).
- 5 Click **Convert Files**.

### Conversion settings

The table below describes the settings for the conversion:

Setting	Description
<b>Acis save file format</b>	Select which version of ACIS to save to when using ACIS as target format.
<b>Enable Healing</b>	Controls whether the conversion engine attempts to heal geometric entities. Only supported for specified formats.
<b>Translate hidden/no-show entities</b>	Controls whether the hidden entities are translated or discarded. Only supported for specified formats.
<b>VRML/STL Scale factors</b>	VRML and STL are often created in units that RobotStudio does not expect; they thus need to be resized.
<b>Delete all generated log files on exit</b>	Makes the CAD converter delete log files when exiting.

## 2 Building stations

### 2.6.3 Troubleshooting and optimizing geometries

### 2.6.3 Troubleshooting and optimizing geometries

#### Overview

The characteristics of the geometries and CAD models in the station may have great effect on your work in RobotStudio, both in aspects of making the objects easier to program as well as enhancing simulation performance.

Below are some guidelines for troubleshooting geometries.

Trouble	Information
The pointer snaps to the wrong parts of the objects when selecting in the graphics window	<p>This problem might be caused by wrong snap mode settings, imprecise selecting, hidden or lack of geometrical information. To resolve these problems, do the following:</p> <ul style="list-style-type: none"><li>• Check the selection level and snap mode settings. For more information, see <a href="#">Selecting an item on page 70</a>.</li><li>• When making the selection, zoom and rotate the object so that you are sure to click <i>inside</i> the object.</li><li>• Check if the object has hidden details that might affect the snapping. Remove details that are not necessary for your programming or simulation. For more information, see <a href="#">Modifying a part on page 93</a>.</li><li>• Some file formats only contain a graphical representation and no geometrical data. Import the geometry from a file format that also contains geometrical data. For more information, see <a href="#">Libraries, geometries and CAD files on page 37</a>.</li></ul>
The graphics window re-draws or updates slowly	<p>This might be due to the performance of your computer not being high enough for the size of the geometry files in your station.</p> <p>To reduce the size of the geometry files, do any of the following:</p> <ul style="list-style-type: none"><li>• Use a lower detail level for rendering the geometry. For more information, see <a href="#">Graphic Appearance on page 461</a>.</li><li>• Check if the object has unnecessary details. Remove details that are not necessary for your programming or simulation. For more information, see <a href="#">Modifying a part on page 93</a>.</li></ul>

*Continues on next page*

Trouble	Information
Parts of the geometry are not visible	<p>If parts of the geometry are not visible from some views, a probable cause is that the object is made up of 2D surfaces and the option <i>Backface culling</i> is on.</p> <p>Backface culling means that the faces of the object are only visible from the front, and if the object (or any of its faces) is oriented differently, they will not be visible.</p> <p>To correct the problem, do one of the following:</p> <ul style="list-style-type: none"> <li>• Switch to modeling mode and invert the direction of the face that is not displayed correctly. This not only corrects the display, it also decreases the chance of faulty orientations during graphical programming. For more information, see <a href="#">Invert on page 465</a> or <a href="#">To invert the direction of all faces of a part on page 91</a>.</li> <li>• Turn backface culling off for the specific object. This makes the object display correctly, but does not affect the direction of the face, which might cause problems if the face will be used for graphical programming. For more information, see <a href="#">To deactivate backface culling for a single object on page 91</a>.</li> <li>• Turn backface culling off for all objects in the station. This makes the objects display correctly, but does not affect the direction of the face, which might cause problems if the face will be used for graphical programming. It also decreases the performance of the graphic handling. For more information, see <a href="#">To change the generic setting for backface culling on page 91</a>.</li> </ul>

#### To invert the direction of all faces of a part

To invert the direction of all faces of a part, follow these steps:

- 1 Select the part on which faces you want to invert the directions.
- 2 On the **Modify** menu, click **Graphic Appearance**.
- 3 On the **Rendering** tab, click **Flip normals** and then click **OK**.

#### To deactivate backface culling for a single object

To change the backface culling setting for a single object, follow these steps:

- 1 Select the part for which you want to change the backface culling setting.
- 2 On the **Modify** menu, click **Graphic Appearance**.
- 3 On the **Rendering** tab, clear the **Backface culling** check box and then click **OK**. The faces of the object will now be displayed even if the generic setting for backface culling is on.

#### To change the generic setting for backface culling

The generic setting for backface culling affects all new objects and existing objects that do not have backface culling specifically deactivated.

- 1 On the **File** menu, click **Options**.
- 2 On the Navigation pane to the left, select **Graphics: Performance**.
- 3 On the **Performance** page to the right, select or clear the **Cull back-facing triangles** check box and then click **OK**.

## 2 Building stations

---

### 2.7.1 Objects

## 2.7 Modeling

### 2.7.1 Objects

---

#### Overview

This section describes how to create or modify geometrical objects.

---

#### Creating a frame

A frame is a generic coordinate system that you can use as reference when positioning objects. Generic frames can also be converted to special kinds of coordinate systems, like workobjects or tool center points.

For procedures, see [Frame on page 213](#) and [Frame from Three Points on page 214](#).

---

#### Creating a solid

With the create solids commands you can create and build models of objects you do not have CAD files or libraries for. With the create solids commands you create primitive solid bodies; these can later be combined to more complex bodies.

For procedures, see [Solid on page 297](#).

---

#### Creating a surface

For procedures, see [Surface on page 301](#).

---

#### Creating a curve

When creating paths with targets based on the object geometries, curves are the geometrical objects that RobotStudio uses. For example, if you want the robot to run along the edge of an object, you can first create a curve along the border and then generate a complete path along that curve, instead of manually finding and creating the necessary targets.

If the CAD model/geometry of the work piece does not already contain curves, you can create the curves in RobotStudio.

For procedures, see [Curve on page 303](#).

---

#### Modifying a curve

When creating paths with targets based on the objects geometries, curves are the geometrical objects that RobotStudio uses. By optimizing the curves before starting programming, you reduce the touch-up of the generated paths.

For procedures, see [Modify Curve on page 474](#).

---

#### Creating a border

For procedures, see [Border on page 308](#).

---

#### Creating a line from normal

A line can be created as a new part and body perpendicular to a surface.

For a procedure, see [Line from Normal on page 315](#).

*Continues on next page*

---

### Extruding a surface or curve

Curves and surfaces and curves can also be extruded to 3D objects, which may then be converted to solids. You can extrude along either a vector or a curve.

For procedures, see [Extrude Surface or Curve on page 313](#).

---

### Modifying a part

When you import a geometry or create an object, it will be one part. A part can, however, contain several bodies. In RobotStudio's modeling mode you can modify the parts by adding, moving and deleting the bodies.

To modify a part, follow this step:

- 1 In the **Modeling** browser, expand the node for the part to modify. Then modify the part by doing any of the following:

To	Do this
Delete a body	Select the body and press the DEL key.
Move a body from one part to another	Drag the body or use the <b>Copy</b> and <b>Paste</b> commands on the <b>Edit</b> menu.
Move one body relative to the others	Select the body and then move it using any of the ordinary commands for moving objects, see <a href="#">Placing objects on page 97</a> .

---

### Modifying a library component

As external files, libraries are merely linked from a station. Therefore, to modify an imported library component, the link must first be broken and later reestablished. For procedures, see [Modify Library Component on page 468](#).

## 2 Building stations

---

### 2.7.2 Mechanisms

### 2.7.2 Mechanisms

---

#### Workflow

This information topic describes how to create a new mechanism, that is, a graphical representation of a robot, tool, external axis or device. The various parts of a mechanism move along or around axes.

Creating a mechanism is dependent upon skillful construction of the main nodes of the tree structure. Four of these—links, joints, frames/tools and calibration—are initially marked red. As each node is configured with enough subnodes to make it valid, the marking turns to green. As soon as all nodes have become valid, the mechanism will be considered compilable and can be created. For additional validity criteria, see the table below.

Node	Validity criteria
Links	<ul style="list-style-type: none"><li>• It contains more than one subnode.</li><li>• The BaseLink is set.</li><li>• All link parts are still in the station.</li></ul>
Joints	<ul style="list-style-type: none"><li>• At least one joint must be active and valid.</li></ul>
Frame/tool Data	<ul style="list-style-type: none"><li>• At least one frame/tool data exists.</li><li>• For a device, no frames are needed.</li></ul>
Calibration	<ul style="list-style-type: none"><li>• For a robot, exactly one calibration is required.</li><li>• For an external axis, one calibration is required for each joint.</li><li>• For a tool or device, calibrations are accepted, but not required.</li></ul>
Dependencies	<ul style="list-style-type: none"><li>• None.</li></ul>

The modify mode of the Mechanism Modeler has two purposes: to enable modification of an editable mechanism in its tree structure, and to complete the modeling of a new or modified mechanism.

It is recommended to configure each main node in the tree structure from the top down. Depending on its current status, right-click or double-click a node or subnode to add, edit or remove it.

For procedures, see [Create Mechanism on page 317](#).

## 2.7.3 Tools and tooldata

---

### Overview

To simulate the robot tool, you need tooldata for the tool. If you import a predefined tool or if you create a tool using the **Create Tool Wizard**, the tooldata is automatically created; otherwise, you have to create the tooldata yourself.

The tooldata simplifies the programming work with respect to the different tools that may come in use. Defining separate sets of tooldata for different tools makes it possible to run the same robot program with different tools: only the new tooldata has to be defined. The tooldata contains the information required for moving and simulating the tool.

Two methods for manipulating tooldata in RobotStudio are as follows:

- Create or modify tooldata, see [Tooldata on page 217](#) and [Modify Tooldata on page 479](#), respectively. This will create all data necessary for programming, but there will be no visual tool during the simulation.
- Create tooldata for an existing geometry, [Create Tool on page 324](#).

---

### Creating and setting up a stationary tool

This information topic describes how to create a stationary tool. For information about creating a robot hold tool, see [Create Tool on page 324](#).

Using a stationary tool, the robot holds and moves the work piece in relation to the tool. Thus, both the tooldata and the workobject must be set up correctly.

To create the tooldata for a stationary tool, follow these steps:

- 1 Import the geometry or library that represents the tool, see [Import Geometry on page 212](#).  
If you do not have the geometry or library at hand but know the position, you can skip this step. The tool will be programable, but not visible in the station.
- 2 Create the tooldata for the tool, see [Tooldata on page 217](#).  
Make sure to set the **Robot holds tool** option to **false**.
- 3 Create a workobject that is moved by the robot. see [Workobject on page 216](#).  
Make sure to set the **Robot holds workobject** option to **true**.
- 4 If you have a geometry or library component for the work piece, attach it to the robot, see [Attach to on page 450](#).

## 2 Building stations

---

### 2.7.4 Setting the local origin of an object

### 2.7.4 Setting the local origin of an object

---

#### Overview

Each object has a coordinate system of its own called local coordinate system in which the object dimensions are defined. When the object's position is referred from other coordinate system, it is the origin of this coordinate system that is used. With the Set Local Origin command you reposition the object's local coordinate system, not the object itself.

For a procedure, see [Set Local Origin on page 492](#).

## 2.8 Placement

### 2.8.1 Placing objects

#### Overview

To achieve the required layout of your station, you need to import or create objects, place them accordingly and, if applicable, attach them to other objects.

Placing objects means setting their position and rotation. If the objects are to be attached to robots or other mechanisms, they will be placed at their attachment point automatically.

The following table describes the actions relating to placement:

Actions	Description
Placing an object	To place an object is to put the object in the required position in the station, see <a href="#">Place on page 484</a> and <a href="#">Set Position on page 494</a> .
Rotating an object	The objects in the station can be rotated to achieve the required layout, see <a href="#">Rotate on page 490</a> .
Measuring distance or angles	The measurement functions calculates distances, angles and diameters between points you select from the graphics window. When using measurements, results and instructions on how to proceed are displayed in the <b>Output</b> window, see <a href="#">The Measure Group on page 316</a> .
Creating a component group	A component group groups related object in the browser, see <a href="#">Component Group on page 262</a> .
Attaching or detaching an object	Objects that are to be used by the robots in any way, such as tools, need to be attached to the robot, see <a href="#">Attach to on page 450</a> and <a href="#">Detach on page 458</a> .
Jogging a robot	Robots can be placed by jogging. The robot axes can also be positioned by jogging, see <a href="#">Jogging mechanisms on page 105</a> .
Modifying the task frame	Modifying the task frame repositions a controller and all its robots and equipment in the station. By default the controller world and the station world coordinate system coincide. This is convenient when building a station with one single controller. For a procedure, see <a href="#">Set Task Frames on page 408</a> . However, when you have several controllers in one station, or need to reposition a controller in an existing station, you need to modify the <a href="#">Edit System on page 409</a> .
Modifying the baseframe position	Modifying the baseframe position sets an offset between the controller's world coordinate system and the baseframe of the mechanical unit. This is necessary when having several mechanical units belonging to one controller, for example, several robots in MultiMove systems or when using positioner external axes. For a procedure, see <a href="#">Edit System on page 409</a> .

## 2 Building stations

---

### 2.8.2 Placing external axes

### 2.8.2 Placing external axes

---

#### Overview

When starting a system with a track or positioner external axis in a RobotStudio station, you have to set up the system to load a model for the track or positioner and get the motions to work properly.

---

#### Prerequisites

The system shall be created with support for track or positioner external axes, see [A system with support for one robot and one positioner external axis on page 175](#).

---

#### Attaching the robot to the track

To attach the robot to the track, follow these steps:

- 1 In the **Layout** browser, drag the robot icon and drop it on the track icon.
- 2 To the question **Should the robot be coordinated with the track?**, answer **Yes** to be able to coordinate the track's position with that of the robot in robot programs. To program the track and the robot independently, answer **No**.
- 3 When asked if you want to restart the system, answer **Yes**.  
The track is now added to the system and ready to be programmed, see [Programming external axes on page 131](#) for more information on how to program the track.



#### CAUTION

If the system is I-started, the setup is deleted and the procedures described here must be performed again.

---

#### Placing the positioner in the station

To place the positioner in the station, follow these steps:

- 1 Move the positioner to the desired position using any of the ordinary functions for placing and moving objects, see [Placing objects on page 97](#).
- 2 Modify the baseframe position of each mechanical unit of the positioner except the INTERCH unit, if it exists. When asked if you want to restart the system, answer **Yes**.  
After the restart the system is updated with the positioner's new location. Continue attaching fixtures and workobjects to the positioner.

---

#### Attaching objects to the positioner

To program robot motions on an object that is held by the positioner, the targets must be created in a workobject that is attached to the positioner. For a complete visual simulation, CAD models that are moved by the positioner should also be attached. To attach the objects, follow these steps:

- 1 Import the models of the fixture and the work piece if you do not have them in the station already, see [Importing a station component on page 87](#).

*Continues on next page*

- 2 Attach the fixture to the positioner, see [Attaching and detaching objects on page 71](#). When asked whether to keep the current position, answer **No**.

If the positioner has several stations, you will be asked which one to attach the object to.

- 3 Attach the work piece to the fixture. When asked whether to keep the current position, answer **No**.
- 4 Attach the workobject in which you will program the work piece to either the fixture, the work piece or the positioner. If you have defined calibration positions on either the work piece or the fixture, it is a good practice to use that object. When asked whether to keep the current position, answer **No**.

The positioner is now set up and ready to be programed, see [Programming external axes on page 131](#) for more information.



#### Tip

If the positioner is of an interchangeable type with several stations, you can either attach individual fixtures, work pieces and workobjects to each station flange, or you can use one set of objects that you attach and detach to the different flanges by events.



#### CAUTION

If the system is I-started, the setup is deleted and the procedures described here must be performed again.

## 2 Building stations

---

### 2.8.3 Placing robots

### 2.8.3 Placing robots

---

#### Overview

When modifying the position of a robot connected to a VC there is a possibility to modify the related task frame or any stationary RAPID objects (tooldata, workobjects) connected to the robot.

---

#### Prerequisites

A robot library must be present in the station and connected to a VC, see [Creating a station with a system on page 75](#).

---

#### Modifying the robot position using a positioning tool

- 1 Modify the baseframe position of a robot connected to a VC using any of the following options:
  - Set Position. See [Positioning an item on page 494](#).
  - Place object by One Point, Two Points, Three Points, Frame, and Two Frames. See [Placing an item on page 484](#).
  - Rotate. See [Rotating an item on page 490](#).
- 2 Click **Apply**.

To the question, **Do you also want to move the Task Frame?**. Click **Yes** or **No**.

  - Click **Yes** to move the task frame, but the base frame keeps its relative placement to the task frame.
  - Click **No** to move the base frame and the placement relative to the task frame will change.



#### Note

If there are any stationary RAPID objects (tooldata, workobjects) in the corresponding task, the following question appears **Do you want to keep the positioning of all stationary RAPID objects?**

- Click **Yes** to keep all the stationary RAPID objects in their global coordinates.
- Click **No** to move all the stationary RAPID objects along with the base frame (same coordinates relative to base frame). Workobjects attached to any other object in the station will not be affected. Workobjects attached to any other object in the station will not be affected.

If the base frame configuration of the VC is updated, the VC has to be restarted for the changes to take effect. i.e. if the base frame changes its placement relative to task frame, the following question appears **Do you want to update the controller configuration and restart?**

- Click **Yes** to restart the controller and update the base frame configuration of the connected VC.
- Click **No** if the base frame is not in accordance with the controller.

*Continues on next page*

---

### Modifying the robot position using Freehand move or rotate

- 1 Modify the baseframe position of a robot connected to a VC using the following Freehand options:

- Move. See [The Freehand Group on page 244](#).
- Rotate. See [Rotating an item on page 245](#).

For information on updating the robot baseframe, see [Updating the baseframe position on page 409](#).

- 2 A warning message is displayed in the Output window.

This page is intentionally left blank

## 3 Programming robots

### 3.1 Workflow for programming a robot

#### Overview

In most cases, going through the workflow from start to finish is recommended, even if it is possible to work in other sequences as well.

Synchronizing will save and load text files containing RAPID modules, and create RAPID programs from your station.

#### Prerequisites

Before creating a program for your robot, you should set up the station, including robots, work pieces and fixtures, in which your robot will work.

#### Programming a robot

The table below describes the workflow for programming a robot to perform the task you require.

Task	Description
Create targets and paths	<p>Create the targets and paths the robot requires to perform the work tasks.</p> <p>To create targets and paths, do one of the following:</p> <ul style="list-style-type: none"> <li>Create a curve to match your required shape. Then use the <b>Create path from curve</b> command to generate a path, complete with targets, along the shape you have created. See <a href="#">Curve on page 303</a> and <a href="#">AutoPath on page 225</a>.</li> <li>Create targets at the requested positions, then create a path and insert the created targets into it. See <a href="#">Create Target on page 219</a>, <a href="#">Teach Target on page 218</a> and <a href="#">Empty Path on page 224</a>.</li> </ul>
Check the target orientations	Make sure that the targets are oriented in the most efficient way for the tasks to be performed. If not, reorient the targets until you are satisfied. See <a href="#">Orientations on page 111</a> .
Check reachability	Check that the robot and tool reach all targets in the path. See <a href="#">Testing positions and motions on page 122</a> .
Synchronize the program to the virtual controller	Generates RAPID code from the RobotStudio items and enables the program to be simulated.
Perform text-based editing	If you need to edit the instructions or data created by RobotStudio, you can start the RAPID Editor. See <a href="#">Examples of using the RAPID editor on page 441</a> .
Collision detection	Check that the robot or tool does not collide with the surrounding equipment or the fixtures. If it does, adjust the placements or orientations until no collisions occur. See <a href="#">Detecting collisions on page 137</a> .
Test the program	Test the program by moving along the paths. See <a href="#">Testing positions and motions on page 122</a> .

## 3 Programming robots

---

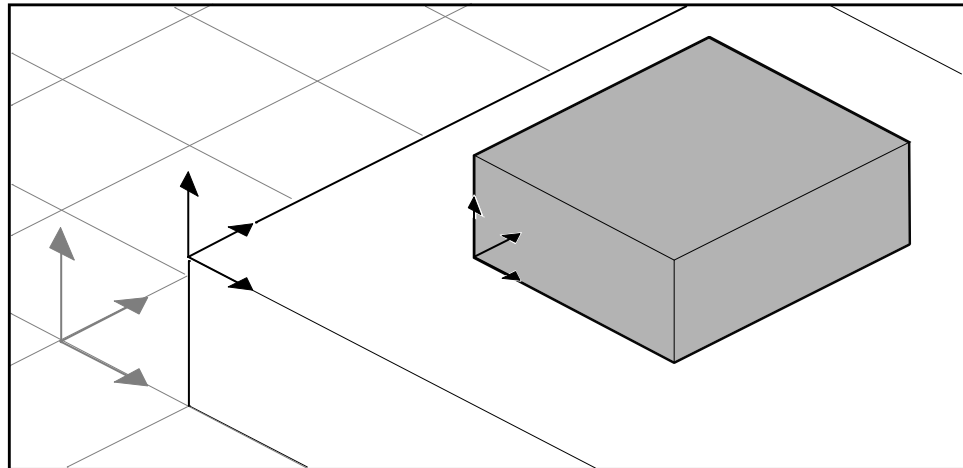
### 3.2 Workobjects

### 3.2 Workobjects

---

#### Creating a workobject

A workobject is a coordinate system used to describe the position of a work piece. The workobject consists of two frames: a user frame and an object frame. All programmed positions will be related to the object frame, which is related to the user frame, which is related to the world coordinate system.



xx050000

For creating a workobject, see [Workobject on page 216](#).

---

#### Modifying a workobject

For a procedure, see [Modify Workobject on page 480](#).

---

#### Converting a frame to a workobject

You can create a new workobject from an existing frame. The converted workobject gets the same name and position as the selected frame.

For a procedure, see [Convert Frame to Workobject on page 455](#).

---

#### Creating a frame by points

You can create a frame by specifying points on the axes of the coordinate system and letting RobotStudio calculate the placement and orientation of the frame's origin.

For a procedure, see [Frame from Three Points on page 214](#).

### 3.3 Jogging mechanisms

#### Jogging a robot

To check if the robot can reach all positions on the work piece, you can jog the TCP or the joints of the robot, either with the freehand commands or through dialog boxes. Jogging the robot close to its boundaries is best done with the latter method.

To	Procedure
Jog the joints of a robot	For freehand, see <a href="#">Jog Joint on page 246</a> . For a dialog box, see <a href="#">Mechanism Joint Jog on page 469</a> .
Jog the TCP of a robot	For freehand, see <a href="#">Jog Linear on page 247</a> . For a dialog box, see <a href="#">Mechanism Linear Jog on page 471</a> .

#### Prerequisites

To jog the TCP of a robot, the robot's VC must be running.

#### Jogging several mechanisms

Function	Description
Multirobot jog	When using multirobot jog, all selected mechanisms will follow the TCP of the one being jogged. Multirobot jog is available for all kinds of jogging. See <a href="#">MultiRobot Jog on page 249</a> .
Jogging with locked TCP	When jogging a mechanism that moves a robot (like a track external axis) with locked TCP, the robot will reposition so that the position of the TCP does not change, even though its baseframe is moved. When jogging an external axis that moves the work object with locked TCP, the robot will reposition so that its TCP follows the work object in the same way as when using multirobot jog. Locked TCP is available when jogging a mechanism that belongs to the same task as a robot. See <a href="#">Mechanism Joint Jog on page 469</a> .

## 3 Programming robots

---

### 3.4 Targets

### 3.4 Targets

---

#### Creating a target

You can create a new target manually either by entering the position for the target in the **Create Target** dialog box or by clicking in the graphics window.

The target will be created in the active workobject.

For a procedure, see [Create Target on page 219](#).

---

#### Creating a jointtarget

A jointtarget is a specification of the position for the robot axes.

For a procedure, see [Create Jointtarget on page 221](#).

---

#### Teaching targets

You can create a new target by jogging the robot and teaching a target at the active TCP. Taught targets will be created with the axis configuration used when jogged to the target.

The target will be created in the active workobject.

For a procedure, see [Teach Target on page 218](#).

---

#### Modifying a target position

By using the modify position command you can modify the position and rotation of a target.

For procedures, see [Set Position on page 494](#) and [Rotate on page 490](#), respectively.

---

#### Modifying a target with ModPos

The position of an existing target can be modified by jogging the robot to a new, preferred position. By selecting a move instruction for the target in a path, the ModPos command can be used to move the target to the TCP of the active tool.

When ModPos is executed, the target, referenced to by the move instruction, will be updated with the following information:

- position and orientation corresponding to the TCP of the active tool
- the current configuration of the active robot
- the current position and orientation values of all active external axes for the active robot



#### Note

To jog a robot linearly, a virtual controller must be running for that robot. For detailed information, see [Starting a VC on page 85](#).

---

#### Renaming targets

With this command you can change the name of several targets at once. You can either rename targets individually, or you can rename all targets in one or several paths at once.

*Continues on next page*

The new target names will consist of an optional prefix, an incremental number and an optional suffix.

For a procedure, see [Rename Targets on page 488](#).

When renaming targets, make sure that the new targets conform to the naming rules. The target names must:

- start with an alphabetical character in the ISO 8859-1 encoding (that is, an ordinary letter from the English alphabet)
- be shorter than 16 characters
- not be empty strings
- not contain any characters illegal in RAPID. See *RAPID reference manual* for details.

---

#### Removing unused targets

If deleting or changing paths or move instructions during programming, you might end up with large numbers of targets that are no longer used in any instructions. To make the workobjects and their targets easier to grasp, you can delete all unused targets.

For a procedure, see [Remove Unused Targets on page 487](#).

### 3.5 Paths

---

#### Creating an empty path

A path is a sequence of targets with move instructions that the robot follows. An empty path will be created in the active task.

For a procedure, see [Empty Path on page 224](#).

---

#### Creating a path from curve

If the work piece has curves or contours that correspond to the path to be created, you can create the paths automatically. The create path from curve command generates paths, complete with targets and instructions along existing curves.

The path will be created in the active task.

The orientation of the targets that will be created will be according to the settings of the approach/travel vectors in the **Options** dialog box.

To create a path from a curve, the curve must have first been created in the station. See [AutoPath on page 225](#).

---

#### Setting robot axis configuration for paths

The robot axis configuration specifies the position of the axes as the robot moves from target to target, when multiple solutions are possible. This is necessary for executing move instructions using configuration monitoring.

Taught targets have validated configurations, but targets created in any other way do not. Also, targets that are repositioned lose their configuration. In RobotStudio, targets without a valid configuration are marked with a yellow warning symbol. See [Robot axis configurations on page 35](#) for more information about configurations.

To set a configuration for all targets in a path, see [Configurations on page 451](#).

To set a configuration for a single target, see [Configurations on page 454](#).

---

#### Reversing paths

The reverse path commands change the sequence of targets in the path so that the robot moves from the last target to the first. When reversing paths, you can reverse either the target sequence alone or the entire motion process.

For procedures, see [Reverse Path on page 489](#).



#### Note

When reversing paths, the original paths are deleted. If you want to keep them, make copies before reversal.



#### Note

When reversing paths, only move instructions are handled. Action instructions, if any exist, have to be inserted manually after the reversal.

*Continues on next page*

---

### Rotating paths

With the rotate path command you can rotate complete paths and move the targets used by the paths accordingly. When rotating paths, the included targets will lose their axis configurations, if any have been assigned.

A frame or target must exist at the position to rotate around before starting the rotate path command.

For a procedure, see [Rotate Path on page 491](#).

---

### Translating a path

The translate path function moves a path and all included targets.

For a procedure, see [Translate Path on page 496](#).

---

### Compensating paths for tool radius

You can offset a path so that it compensates for the radius of a rotating tool. Since the targets in the path are moved, they will lose their axis configurations, if any have been assigned.

For a procedure, see [Tool Compensation on page 495](#).

---

### Interpolating a path

The interpolate functions reorient the targets in a path so that the difference in orientation between the start and end targets is distributed evenly among the targets in between. The interpolation can be either linear or absolute.

Linear interpolation distributes the difference in orientation evenly, based on the targets' positions along the length of the path.

Absolute interpolation distributes the difference in orientation evenly, based on the targets' sequence in the path.

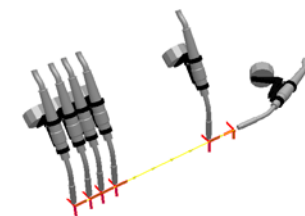
Below are examples of the difference between linear and absolute interpolation.

The interpolate functions reorient the targets in a path so that the difference in orientation between the start and end targets is distributed evenly among the targets in between. The interpolation can be either linear or absolute.

For a procedure, see [Interpolate Path on page 464](#).

### No interpolation

This is the path before any interpolation. Note that the last target is oriented differently than the others.



xx050026

*Continues on next page*

## 3 Programming robots

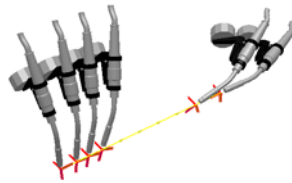
---

### 3.5 Paths

#### Continued

#### Linear interpolation

This is the same path after linear interpolation.



xx050027

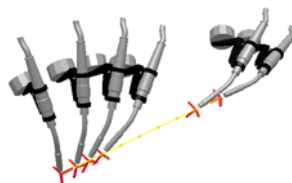
Note that the targets are oriented based on their placement relative to the start and end targets.

If a target were moved and you reran the linear interpolation, it would be reoriented according to its new position.

If new targets were inserted between the existing ones and you reran the linear interpolation, it would not affect the orientation of the existing targets.

#### Absolute interpolation

This is the same path after absolute interpolation



xx050028

Note that the targets are orientated based on their sequence in the path: each target has been reoriented equally, regardless of its place.

If a target were moved and you reran the absolute interpolation, it would not affect the orientation.

If new targets were inserted between the existing ones and you reran the absolute interpolation, it would change the orientation of all targets.

---

#### Mirroring a path

The mirror path function mirrors all motions instructions and their targets to a new path.

For a procedure, see [Mirror Path on page 472](#).

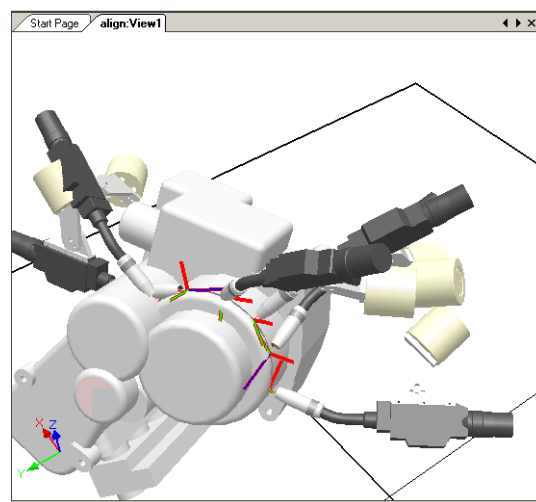
## 3.6 Orientations

### Overview

This is an overview of the tools for automating the modification of target orientations. When creating paths from curves in RobotStudio, the orientation of the targets depends on the characteristics of the curves and the surrounding surfaces. Below is an example of a path with unordered target orientations and examples of how the different tools have affected the targets.

### Unordered orientations

In the path below, the target orientations are unordered. The function View tool at target has been used for illustrating how the targets point in different directions.



xx050029

*Continues on next page*

## 3 Programming robots

---

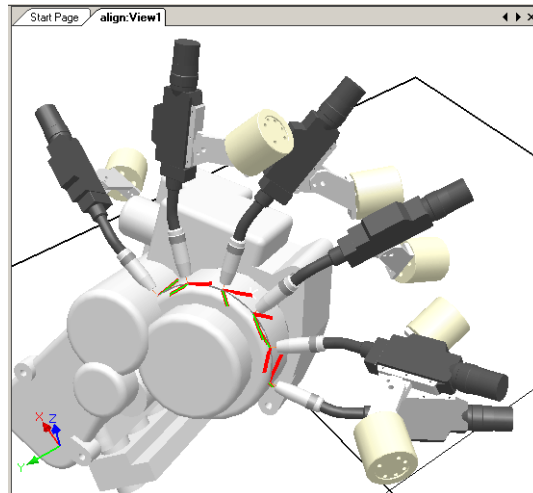
### 3.6 Orientations

*Continued*

---

#### Effect of target normal to surface

In the picture below, the targets, which previously were orientated randomly, have been set normal to the flat round surface at the right side of the path. Note how the targets' Z axis has been orientated normal to the surface; the targets have not been rotated in the other directions.



xx050030

---

#### Setting a target normal to surface

To set a target orientation normal to a surface is to make it perpendicular to the surface. The target can be oriented normal to the surface in two different ways:

- The entire surface can be used as a reference for the normal. The target will be oriented as the normal to the closest point at the surface. The entire surface is the default surface reference.
- A specific point on the surface can be used as the reference for the normal. The target will be orientated as the normal to this point, regardless of whether the normal to the closest point at the surface has another orientation.

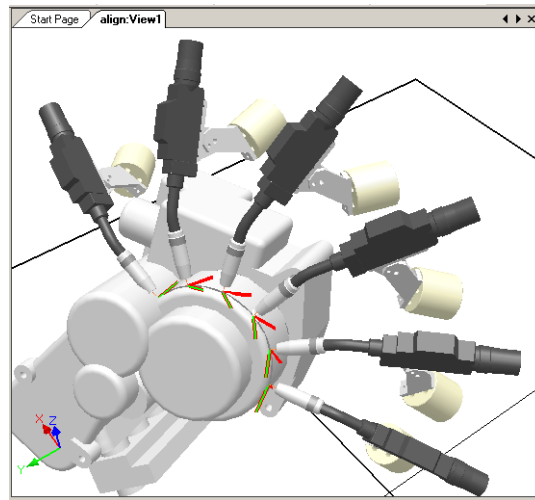
Objects imported without geometry (for example, .jt files) can only refer to specific points on the surface.

For a procedure, see [Set Normal to Surface on page 493](#).

*Continues on next page*

#### Effect of align target orientation

In the picture below, the targets, which were previously orientated with the Z axis normal to the surface but with the X and Y axes orientated randomly, have been organized by aligning the targets' orientation around the X axis with the Z axis locked. One of the targets in the path has been used as reference.



xx050031

#### Aligning a target orientation

With the align target orientation command you align the rotation of selected targets around one axis without changing the rotation around the others.

For a procedure, see [Align Target Orientation on page 449](#).



#### Tip

You can also align ordinary frames in the same way.

*Continues on next page*

## 3 Programming robots

---

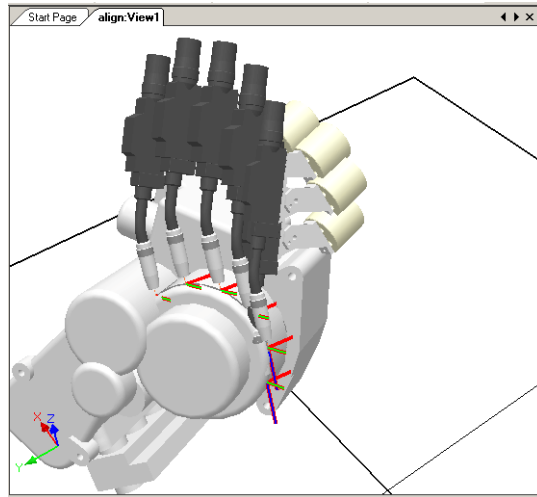
### 3.6 Orientations

*Continued*

---

#### Effects of copy and apply orientation

In the picture below, the targets, which were previously oriented randomly, have been organized by copying the exact orientation of one target to all the others. This is a quick way to fix workable orientations for processes where variations in approach, travel, or spin directions either do not matter or are not affected, due to the shape of the work piece.



xx050032

---

#### Copying and applying an orientation for objects

To transfer an orientation from one object to another is an easy way to align different frames for simplifying the programming of the robot. Target orientations may also be copied.

For procedures, see [Copy / Apply Orientation on page 457](#).

### 3.7 RAPID Instructions



#### Note

For information on the RAPID Editor, which is used for creating and modifying RAPID program code, see [RAPID tab on page 413](#).

#### Move and action instructions

For RAPID programming, RobotStudio's main advantage is in the area of motion programming.

A move instruction is an instruction for the robot to move to a specified target in a specified manner. With RobotStudio, you can create move instructions in three ways:

Method	Description
Create a move instruction based on an existing target	Creates move instructions based on one or several targets selected in the <b>Paths&amp;Targets</b> browser. For a procedure, see <a href="#">Add to Path on page 447</a> .
Create a move instruction and a corresponding target	Creates a move instruction and a corresponding target at once. The position of the target can either be selected from the graphics window or typed numerically. For a procedure, see <a href="#">Move Instruction on page 236</a> .
Teach a move instruction	Teaching a move instruction creates a move instruction and a corresponding target at the robot's current position. Teaching a move instruction also stores the current configuration with the target. For a procedure, see <a href="#">Teach Instruction on page 235</a> .

In addition to move instructions, you can also create and insert action instructions from RobotStudio. An action instruction is an instruction other than a move instruction that can, for example, set parameters, or activate or deactivate equipment and functions. The action instructions available in RobotStudio are limited to those commonly used for affecting the robot's motions. For inserting other action instructions or another kind of RAPID code in the program, use the RAPID Editor. For a procedure, see [Action Instruction on page 237](#).

The table below lists the action instructions that can be created. For details, see the RAPID Reference Manual.

Action instruction	Description
ConfL On/Off	ConfL specifies whether to monitor the robot's configurations during linear movements. When ConfL is set to Off, the robot may use another configuration than the programmed one for reaching the target during program execution.
ConfJ On/Off	ConfJ specifies whether to monitor the robot's configurations during joint movements. When ConfJ is set to Off, the robot may use another configuration than the programmed one for reaching the target during program execution.
Actunit <i>UnitName</i>	Actunit activates the mechanical unit specified by <i>UnitName</i> .
DeactUnit <i>UnitName</i>	Deactunit deactivates the mechanical unit specified by <i>UnitName</i> .

*Continues on next page*

## 3 Programming robots

### 3.7 RAPID Instructions

*Continued*

Action instruction	Description
ConfJ On/Off	ConfJ specifies whether to monitor the robot's configurations during joint movements. When ConfJ is set to Off, the robot may use another configuration than the programmed one for reaching the target during program execution.
Actunit <i>UnitName</i>	Actunit activates the mechanical unit specified by <i>UnitName</i> .
DeactUnit <i>UnitName</i>	Deactunit deactivates the mechanical unit specified by <i>Unit-Name</i> .

#### Modifying an instruction

Most instructions have arguments that specify how the instruction shall be carried out. For example, the MoveL instruction has arguments that specify the speed and accuracy with which the robot moves to the target.

For a procedure, see [Modify Instruction on page 477](#).



#### Note

Some arguments are read from the virtual controller. If the virtual controller has not been started, only the arguments stored in the station can be modified.

#### Converting to move circular

To create a circular motion to an instruction target, you must convert the motion type to circular motion (that is, MoveC in RAPID).

A circular motion is defined by two motion instructions, where the first is the via-point and the second contains the end point of the circular motion.

The circular motion can only be used for open circular arcs, not for closed circles. To create a path for a closed circle, use two circular motions.

For a procedure, see [Convert to Move Circular on page 456](#).

#### Creating RAPID instructions for setting I/O signals

For controlling I/O signals in the robot program, you use RAPID commands that set the signals. These require that you first create instruction templates for the instructions that set the signals. See *RAPID reference manual* for details about the instructions that control I/O signals.

To add RAPID instructions that set I/O signals, follow these steps:

- 1 Synchronize the system in which you want to add the instructions to the virtual controller, see [Synchronization on page 134](#).
- 2 In programming mode, select the module for editing, right-click it and then click **Edit program**.
- 3 In the RAPID editor, add the instructions for setting the signals.
- 4 When you are done adding instructions, synchronize the task and paths from the Virtual Controller back to the station.

*Continues on next page*

---

#### Using cross-connections and groups for setting I/O signals

You can also create cross-connections and signal groups, which make one signal set the value of several other signals. See *System parameters reference manual* for details about cross-connections and groups.

To make one signal set several others, follow these steps:

- 1 Request write access, and then open the configuration topic I/O in the configuration editor. Add configure instances for the cross-connections and groups to create.

---

#### Instruction templates

Instruction templates contain predefined sets of argument values that are applied to the instructions you create using the template. You can create templates for all instructions in the system running on the virtual controller. To see which instructions are available and what their arguments do, see the RAPID reference manual for your RobotWare version and the reference sections in manuals for software options, if you have any installed on the system.

*Move instruction templates* are always part of *process templates*. The process templates contain one instruction template for each type of move instruction that might be used by the process.

The *process templates* are instances of *process definitions*, which define the types of move instructions (*move instruction definitions*) that might be used by the process.

To create new move instruction templates, start by creating a new process template for a process that uses the type move instructions you want to create templates for. If such a process does not exist, you first have to create a new process definition.

If no *move instruction definition* for the type of instruction you want to create a template for exists, you must create it first.

When creating instruction descriptions, the virtual controller must be running, since the available instruction types are read from the system.

Templates can be imported and exported on four levels: tasks, move instruction descriptions, action instruction descriptions and process definitions. The default directory for imported and exported template files is *My Documents/RobotStudio*. Simply choosing another directory will then make that directory default. As a default .xml is the file format.

The validation procedure checks for duplicate names, incomplete process definitions and virtual controller equality. It is performed automatically, after a template file has been imported or a node renamed or deleted.

For procedures, see [Instruction Template Manager on page 238](#).

---

#### Instruction Template Manager

The Instruction Template Manager is used to add support for instructions other than the default set that comes with the RobotStudio.

For example, a robot controller system with the RobotWare Dispense option has specialized move instructions related to glueing like DispL and DispC. You can

*Continues on next page*

### 3 Programming robots

---

#### 3.7 RAPID Instructions

*Continued*

manually define the instruction templates for these using the Instruction Template Manager. The instruction templates are exported to XML format and reused later.

The instruction template supports the following Robotware options:

- Cap (Continuous Application Process)
- Disp (Dispense)
- Trigg (Fixed Position Events)
- Spot Pneumatic
- Spot Servo
- Spot Servo Equalizing
- Paint

RobotStudio has pre-defined XML files that are imported and used for robot controller systems with the appropriate RobotWare options. These XML files have both the Move and Action instructions.



#### Note

Use RobotStudio ArcWelding PowerPac while using RobotWare Arc.

---

#### Creating a process template with move instruction templates



#### Note

Move instructions are always related to processes.

- 1 On the **Home** tab, from the active **Task** list, select the task for the robot for which you want to create the instruction template.
- 2 On the **Create** menu, click **Instruction Template Manager**. This opens the Instruction templates page in the work space.
- 3 In the **Instruction Templates** tree to the left, make sure there are move instruction definitions for the types of move instructions you want to create templates for. If not, follow the procedure in [Creating a move instruction description on page 119](#) for creating them.
- 4 Make sure there is a process definition that uses the types of move instructions you want to create templates for. If not, follow the procedure in [Creating a process definition on page 119](#) for creating it.
- 5 Right-click the *process definition* for which you want to create a new templates and click **Create Process Definition**.
- 6 In the **Create Process Definition** dialog box, enter a name, with characters from ASCII set, for the new template and click **Create**. A new process template node with a set of move instruction templates is now created.
- 7 Select each new template one at a time, and in the arguments grid to the right of the tree view, set the argument values that shall be applied when you create new instructions based on the template. Finish for each template by clicking **Apply changes** at the bottom of the grid.

*Continues on next page*

For details about available arguments and what they do, see the RAPID reference manual for ordinary RAPID instructions and the option manual for software option instructions.

---

#### Creating a move instruction description

To create templates for other instructions than the one that already exists in the tree view, you first have to create an instruction description that defines the arguments that belong to the instruction. To create the instruction description, follow these steps:

- 1 Right-click the **Move Instructions** node and click **Create Move Instruction Description**. This opens the **Create Move Instruction Description** dialog box.
- 2 In the **Controller Description** list, select the instruction you want to make a description for. All action instructions installed on the controller, both through RobotWare and software options, are available.
- 3 In the **Motion type** list, select the motion type for the instruction.
- 4 Optionally, in the **Information text** box, enter a comment to the instruction.
- 5 Click **Create**. The instruction description appears in the tree view and its settings are displayed in the grid.
- 6 In the instruction grid, set the **point type** values. If necessary, also change the other settings.
- 7 After changing the settings, click **Apply Changes** in the bottom of the grid view.

---

#### Editing an instruction template

To edit an instruction template, follow these steps:

- 1 In the active task list, select the task for the robot for which you want to edit the instruction template.
- 2 On the **Create** menu, click **Instruction Template Manager**. This opens the Instruction templates page in the work space.
- 3 In the **Instruction Templates** tree to the left, browse to and select the template to edit.
- 4 In the arguments grid to the right of the tree view, set the argument values that shall be applied when you create new instructions based on the template. Finish by clicking **Apply changes** at the bottom of the grid.

For details about available arguments and what they do, see the RAPID reference manual for ordinary RAPID instructions and the option manual for software option instructions.

---

#### Creating a process definition

To create a process definition, follow these steps:

- 1 Make sure there are move instruction definitions for the types of move instructions you want to create templates for. If not, follow the procedure above for creating them.

*Continues on next page*

## 3 Programming robots

---

### 3.7 RAPID Instructions

*Continued*

- 2 Right-click the **Process definitions** node and click **Create Process Definition**. This opens the Create Process Description dialog box.
- 3 In the **Create Process Definitions** dialog box, enter a name for the process definition, a name for its first process template and then select the move instruction types to use. Finish by clicking **Create**.

---

#### Creating an action instruction template

To create a template for an action instruction, follow these steps:

- 1 In the active task list, select the task for the robot for which you want to create the instruction template.
- 2 On the **Home** menu, click **Instruction Template Manager**.
- 3 In the **Instruction Templates** tree to the left, right-click the instruction description (corresponding to the instructions as described in the RAPID reference manual) for which you want to create a new template and click **Create Action instruction Template**.

If the instruction description does not exist in the tree, create it by following the procedure described in [Creating an action instruction description on page 120](#).

- 4 In the **Create Action Instruction Template** dialog box, enter a name for the new template and click **Create**. The new template is now created under the instruction description node it belongs to.
- 5 Select the new template and in the arguments grid to the right of the tree view, set the argument values that shall be applied when you create new instructions based on the template. Finish by clicking **Apply changes** at the bottom of the grid.

For details about available arguments and what they do, see the RAPID reference manual for ordinary RAPID instructions and the option manual for software option instructions.

---

#### Creating an action instruction description

To create templates for other instructions than the one that already exists in the tree view, you first have to create an instruction description that defines the arguments that belong to the instruction.

To create the instruction description, follow these steps:

- 1 Right-click the **Action Instructions** node and click **Create Action Instruction Description**. This opens the **Create Instruction Description** dialog box.
- 2 In the **Controller Description** list, select the instruction you want to make a description for. All action instructions installed on the controller, both through RobotWare and software options, are available.
- 3 Optionally, in the **Information text** box, enter a comment to the instruction.
- 4 Click **Create**. The instruction description appears in the tree view, and its settings are displayed in the grid.

After changing the settings (if necessary), click **Apply Changes** in the bottom of the grid view.

*Continues on next page*

- 5 Continue with creating templates for the instruction description, as described in [Creating an action instruction template on page 120](#).

## 3 Programming robots

---

### 3.8 Testing positions and motions

## 3.8 Testing positions and motions

---

### Overview

RobotStudio has several functions for testing how robots reach and move to targets. They are useful both for finding the optimal layout when building a station and during programming.

Below are brief descriptions of the functions for testing reachability and motions.

---

### Checking reachability

The check reachability function displays whether the robot can reach selected targets and motion instructions by changing the frames' colors in the graphic view. Reachable frames are colored green, unreachable red, and frames with reachable positions but not with current orientation are colored yellow.

The reachability check is useful when building the station, since the reachability of several targets are displayed at once. For a procedure, see [Check Reachability on page 453](#).

---

### Jumping to target

Jump to target tests whether the robot can reach a specific position. This is useful when building the station: by creating targets at critical positions on the work piece and jumping the robot to them, you get an early indication of whether the items are positioned correctly or not. For a procedure, see [Jump to Target on page 466](#).

---

### Viewing a robot at target

When View robot at target is activated, the robot is automatically positioned with the tool at the target when one is selected. If several robot axis configurations are possible for reaching the target, the robot will use the one nearest the configuration it had before jumping to the target. For a procedure, see [View Robot at Target on page 497](#).

---

### Viewing tool at target

View tool at target displays the tool at target, without checking that the robot can reach it. This test is useful both when building the station and when programming the robot, since the orientation of targets both affects the reachability and the process performance. For a procedure, see [View Tool at Target on page 498](#).

---

### Executing move instructions

Execute move instruction tests if the robot can reach a specific position with the programmed motion properties. This is useful for testing motions during programming. For detailed information, see [Execute Move Instruction on page 459](#).

---

### Moving along path

Move along path executes all move instructions in a path. It is thereby a more complete test than Execute move instructions, but not as complete as a full simulation, since it ignores RAPID code that is not move instructions. For a procedure, see [Move Along Path on page 481](#).

*Continues on next page*

---

#### Moving to a pose

Moving to a pose moves a mechanism to a predefined joint value at a predefined time without using the Virtual Controller. This is useful when movement of the external equipment (such as a clamp or conveyer) must be simulated. For a procedure, see [Move to Pose on page 482](#).

---

#### Simulating programs

Simulating programs involves running a program on the virtual controller as it is run on a real controller. It is the most complete test whereby you can see how the robot interacts with external equipment through events and I/O signals. For a procedure, see [Simulation Setup on page 329](#).

---

#### Improving the reachability

If the robot cannot reach the target, or if you are not satisfied with the motions, try the following for improving the reachability:

- 1 Set ConfL or ConfJ to Off for enabling the robot to use new configurations for reaching the target.
- 2 Change the orientation of the target.
- 3 Change the position of either the robot or the work piece.
- 4 Use a system with a track external axis for increasing the robot's range.
- 5 Use a system with a positioner external axis for enabling different work piece positions for different targets.

## 3 Programming robots

---

### 3.9.1 About programming MultiMove

## 3.9 Programming MultiMove systems

### 3.9.1 About programming MultiMove

---

#### About MultiMove

The MultiMove functions help you create and optimize programs for MultiMove systems where one robot or positioner holds the work piece and other robots operate on it. Below is an outline of the main workflow for programming MultiMove systems with RobotStudio, with references to detailed instructions further down in the section.

---

#### Prerequisites

For using the MultiMove functions you must first have the following:

- A virtual controller running a MultiMove system started in RobotStudio, see [A MultiMove system with two coordinated robots on page 173](#) for an example.
- All coordinate systems and tools used by the system.
- The paths along which the tool shall move. The paths must be created in a workobject that belongs to a tool robot and is attached to the work piece robot. A wizard will guide you through attaching the workobjects if this has not been done before starting the MultiMove functions.

For detailed information about MultiMove in RobotWare systems and RAPID programs, see *MultiMove application manual*.

---

#### Normal workflow

This is the typical workflow for creating MultiMove programs using the MultiMove function:

Action	Description
Setting up the MultiMove	Select the robots and paths to use in the program, see <a href="#">Setting up the MultiMove on page 126</a> .
Testing the MultiMove	Execute the motion instructions along the paths, see <a href="#">Testing the MultiMove on page 127</a> .
Tuning the motion behavior	Tune motion behavior, such as tolerances and constraints for TCP motions, see <a href="#">Tuning the motion behavior on page 128</a> .
Creating the program	Generate the tasks for the robots, see <a href="#">Creating paths on page 130</a> .

*Continues on next page*

#### Additional actions

In addition to using the functions that calculate and create optimized MultiMove paths, you can program MultiMove manually using a combination of the ordinary programming tools in RobotStudio and a set of tools specific for MultiMove programming.

The main actions for programming MultiMove manually are outlined below. Not all actions might be necessary, but the order in which they shall be carried out depends on the contents of the station and your goals.

Action	Description
Creating Tasklists and Syncidents	This data specifies the tasks and paths that shall be synchronized with each other. See <a href="#">The Create Tasklist tool on page 234</a> and <a href="#">The Create Syncident tool on page 234</a> , respectively.
Adding and updating ID arguments to the instructions to synchronize	To add IDs to the instructions, you can use one of the following methods: Using <a href="#">The Recalculate ID tool on page 233</a> to add and update IDs for instructions in paths that already are synchronized. Using <a href="#">The Convert path to MultiMove path tool on page 234</a> to add IDs to instructions in paths that have not yet been synchronized.
Adding and adjusting Sync instructions to the paths.	Add <code>SyncMoveon/Off</code> or <code>WaitSyncTask</code> instructions to the paths to synchronize and set their tasklist and Syncident parameters. See <a href="#">Creating an action instruction on page 237</a> .
Teaching MultiMove instructions	It is also possible to jog all robots to the desired positions and then teach instructions to new synchronized paths. See <a href="#">MultiTeach tab on page 230</a> .

### 3.9.2 Setting up the MultiMove

---

#### Selecting robots and paths

This procedure is for selecting the robots and paths in the station that shall be used for the MultiMove program. All robots for the MultiMove program must belong to the same system.

- 1 On the **Home** tab, click **MultiMove**. Click the **Setup** tab below the MultiMove work area.
- 2 In the work area, click the **System config** bar for expanding the system config section.
- 3 In the **Select System** box, select the system that contains the robots to program.  
The robots of the selected system are now displayed in the System grid below the Select system box.
- 4 For each robot that shall be used in the program, select the check box in the **Enable** column.
- 5 For each robot that shall be used in the program, specify whether it carries the tool or the work piece using the options in the **Carrier** column.
- 6 In the work area, click the **Path config** bar for expanding the path config section.
- 7 Select the **Enable** check box for the tool robot and click the expand button. This displays the paths of the robot.
- 8 Select the order of the paths to execute by specifying them in right order using **Path name** column.
- 9 For each path that shall be included in the program, select the check box in the **Enable** column.
- 10 When you have set up the robots and paths, continue testing the Multimove and then tune the motion properties, if necessary.

### 3.9.3 Testing the MultiMove

#### Overview

Testing the MultiMove executes the motion instructions along the paths according to the current settings on the setup and motions properties pages.

#### Testing Paths

This procedure is for setting the robots start position and testing the resulting movements along the path sequence.

- 1 Jog the robots to what seems to be a good start position.
- 2 On the **Home** tab, click **MultiMove**. Click the **Test** tab at the bottom of the MultiMove work area for displaying the test area.
- 3 Optionally, select the **Stop at end** check box to make the simulation stop after moving along the paths. Clearing this check box makes the simulation continue in a loop until you click **Pause**.
- 4 Click **Play** to simulate the motions along the paths based on the current start position.

If you are satisfied with the motions, continue generating multimove paths. If the simulation cannot complete or if you are not satisfied with the motions, pause the simulation and perform any of the actions below to adjust the motions:

Action	Description
Examine the robots' positions for critical targets.	Click <b>Pause</b> and then use the arrow buttons to move to one target at a time.
Jog the robots to new start positions.	New start positions might result in changed motions, since the robots will use different configurations. In most cases, positions near the robots' joint limits shall be avoided.
Go to the <b>Motion Behavior</b> tab and remove constraints.	The default setting for the motion properties is no constraints. If this has been changed, constraints might exist that limit motions more than necessary.

## 3 Programming robots

---

### 3.9.4 Tuning the motion behavior

### 3.9.4 Tuning the motion behavior

---

#### Overview

Tuning the motion behavior means to set up rules for the robot's motions, for example, constraints on the position or orientation of the tool. Generally, the MultiMove program will obtain the smoothest motions with the fastest cycle and process times with as few constraints as possible.

For procedures, see [Motion Behavior tab on page 227](#).

---

#### Modifying the joint influences

The joint influence controls the balance of how much the robots will use their joints. Decreasing the weight value for one axis will restrict the motion for this axis, while increasing it will promote motion on this axis relative to alternative axes.

- 1 On the **Home** tab, click the **Motion Behavior** tab.
- 2 Expand the **Joint Influence** group by clicking its title bar.
- 3 In the Select Robot box, select the robot whose joint influence you want to modify.

The weight values for the robot axes are now displayed in the grid.

- 4 For each axis whose motion you want to restrict or promote, adjust the Weight value. A lower value restricts, and a higher value promotes, motions on that axis.

---

#### Modifying the TCP constraints

The joint influence controls the balance of how much the robots will use their joints. Decreasing the weight value for one axis will restrict the motion for this axis, while increasing it will promote motion on this axis relative to alternative axes.

- 1 On the **Simulation** tab, click the **Motion Behavior** tab.
- 2 Expand the **TCP Constraints** group by clicking its title bar.  
The directions and rotations in which you can constrain the TCP's motion are now displayed in the grid
- 3 For each pose you want to constrain, select the **Enable** check box and specify the constraint values (location in the TCP coordinate system). To use the values from the current TCP position, click **Pick from TCP**.
- 4 Optionally, adjust the **Weight** value for the constraint. A low value results in a harder constraint, while a high value allows a larger deviation.

---

#### Modifying the tool tolerance

The joint influence controls the balance of how much the robots will use their joints. Decreasing the weight value for one axis will restrict the motion for this axis, while increasing it will promote motion on this axis relative to alternative axes.

- 1 On the **Simulation** tab, click the **Motion Behavior** tab.
- 2 Expand the **Tool Tolerance** group by clicking its title bar.

The directions and rotations in which you can enable tolerances are now displayed in the grid

*Continues on next page*

- 3 For each offset you want to set, select the **Enable** check box.
- 4 In the **Value** column, specify the allowed deviation.
- 5 Optionally, adjust the **Weight** value for the tolerance. A low value increases the use of the tolerance, while a high value promotes motions that do not use the tolerance.

---

#### Modifying the tool offset

The tool offset sets a fixed distance between the tool and the paths.

- 1 On the **Simulation** tab, click the **Motion Behavior** tab.
- 2 Expand the **Tool Offset** group by clicking its title bar.

The directions and rotations in which you can set offsets are now displayed in the grid.
- 3 For each offset you want to set, select the **Enable** check box.
- 4 In the **Offset** column, specify the offset distance.

## 3 Programming robots

### 3.9.5 Creating paths

### 3.9.5 Creating paths

#### Overview

When you are satisfied with the motions displayed when testing the Multimove program, the next step is to convert the temporary move instructions used by the MultiMove function to ordinary paths in RobotStudio.

#### Creating the paths

To create paths for the MultiMove program in RobotStudio, follow these steps:

- 1 On the **Home** tab, click **Create Paths** tab.
- 2 Expand the **Settings** group by clicking on the its title bar.
- 3 Optionally, change the naming settings in the following boxes:

Box	Description
Start ID	Specify the first ID number for the synchronization of the instructions for the robots.
ID step index	Specify the increment between ID numbers.
Sync ident prefix	Specify a prefix for the syncident variable, which connects the sync instructions in the tasks for the tool robot and the work piece robot with each other.
Task list prefix	Specify a prefix for the tasklist variable, which identifies the tasks for the tool robot and the work piece robot to synchronize.

- 4 Expand the **WP Robot Settings** group by clicking on its title bar and then check the settings in the following boxes:

Box	Description
WP Workobject	Specify the workobject to which the targets generated for the workpiece robot shall belong.
WP TCP	Specify which tooldata the workpiece shall use when reaching its targets.
Path prefix	Specify a prefix for the generated paths.
Target prefix	Specify a prefix for the generated targets.

- 5 Expand the **Generate path** group by clicking on its title bar and then click **Create Paths**.

## 3.10 Programming external axes

### Overview

This is a brief overview of the functions and commands for programming external axes in RobotStudio. For a more detailed description of external axes and how to program them, see the product manual for the external axis to use and the *RAPID reference manual*.

### Coordinated motions

Normally, external axes are used to move the workpiece, the robot or any other mechanism. The motions of an external axis can be coordinated with those of a robot in two ways, depending on the task in which the external axis is defined.

Task for external axis	Coordination method
Same task as the robot's	<p>If the external axis is in the same task as the robot, the current position of active external axes is stored with each target that is created. When the robot then moves to the target, the external axis will move to the stored position as well.</p> <p>Modifying and optimizing the position of positioner external axes can be automated using the MultiMove function, or be performed manually for selected targets. Positions of track external axes can only be modified manually.</p> <p>For information about using the MultiMove function, see <a href="#">About programming MultiMove on page 124</a>. For information about how to modify the position of external axes manually, see below.</p>
Other task than the robot's	<p>If the external axis is in another task than that of the robot it shall be coordinated with, the motions of the external axis are created by <i>MoveExt</i> instructions, and the coordination is made by <i>sync</i> instructions.</p> <p>For positioner external axes creating or optimizing <i>MoveExt</i> and <i>sync</i> instructions can be automated way using the MultiMove function, or be performed manually by creating a path with <i>MoveExt</i> instructions for the positioner and then adding <i>sync</i> instructions to the path for the robot and the external axis. Track external axes can only be programmed manually.</p> <p>For information about using the MultiMove function see <a href="#">About programming MultiMove on page 124</a>. For information about how to use the <i>sync</i> instructions, see <i>RAPID reference manual</i> and <i>MultiMove application manual</i>.</p>

### Modifying positions of external axes

When programming external axes, you often need to adjust the position of the external axis for some targets. For example, if you create a path from curves on a work piece that is attached to a positioner, the positioner will initially have the same position for all targets. By repositioning the work piece for some of the targets you might improve process time and reachability.

When targets are created in stations with a coordinated external axis, the position values of the external axis are stored in the target. With the Modify External Axis function you can reposition the external axis, thus making it possible for the robot to reach the target in new ways. For a procedure, see [Modify External Axis on page 476](#).

*Continues on next page*

## 3 Programming robots

---

### 3.10 Programming external axes

*Continued*

To modify the external axis values for a target, the following conditions must be met:

- The external axis must be added to the system and set up correctly. For examples of how to add support for an external axis to a system, see [A system with support for one robot and one positioner external axis on page 175](#). For information about how to set up an external axis in a RobotStudio station, see [Placing external axes on page 98](#).
- The external axis must be defined in the same task as the robot.
- The external axis must be activated.

---

#### Activation and deactivation

Activating a mechanical unit makes it controlled and monitored by the controller. Consequently, the mechanical unit must be activated before programming or running programs. If a system uses several external axes or interchangeable models with several work stations, several mechanical units might share common drive units. If this is the case, you must make sure to set the mechanical unit as active.

For more information about activating and deactivating mechanical units, see *RAPID reference manual* on the instructions *ActUnit* and *DeactUnit*.

Activating and deactivating mechanical units can be done either manually, see [Activate Mechanical Units on page 339](#), or programmatically by RAPID instructions, see below.

---

#### To activate or deactivate mechanical units programmatically

To set the mechanical units to be active programmatically by RAPID instructions, follow these steps:

- 1 In the **Paths&Targets** browser, browse down to the path in which you want to insert the activation or deactivation instruction. To insert it as the first instruction in the path, select the path node and to insert it between existing instructions, select the instruction before the intended insertion point.
- 2 On the **Home** tab, click **Action Instruction** to bring up a dialog box.
- 3 In the **Instruction Templates** list, select one of the **ActUnit** or **DeactUnit** instructions.
- 4 In the **Instruction Arguments** grid and the **MechUnit** list, select the unit to activate or deactivate.
- 5 Click **Create**. When the path is executed either through the Move along path command, or running the RAPID program, the instruction will be carried out.

## 3.11 Loading and saving programs and modules

---

### Overview

RAPID programs and modules are normally stored in the RobotWare systems, as they are created. You can also save the programs to files on the PC, which makes it possible to load them to other controllers, either other virtual controllers or real IRC5 controllers.

### Programs are saved from the VC

When saving a program to files on the PC from RobotStudio, it is the RAPID program stored in the system of the VC that is saved. This program is created and updated by synchronizing the station to the VC, see [Synchronize to VC on page 415](#).

### Procedures

To create or load a module or load a program, see:

- [Creating a new RAPID module on page 423](#)
- [Loading a RAPID module on page 423](#)
- [Loading a RAPID program on page 427](#)

To save a module or program, see:

- [Saving a RAPID module as another on page 423](#)
- [Saving a program on page 427](#)

## 3.12 Synchronization

---

### Overview

To synchronize is to make sure that the RAPID program in the system running on the virtual controller corresponds to the programs in RobotStudio. You can synchronize both from RobotStudio to the virtual controller and from the virtual controller to RobotStudio.

In a RobotStudio station, robot positions and movements are defined by targets and move instructions in paths. These correspond to data declarations and RAPID instructions in the modules of the RAPID program. By synchronizing the station to the virtual controller, you create RAPID code out of the data in the station. By synchronizing the virtual controller to the station, you create paths and targets out of the RAPID program in the system running on the virtual controller.

---

### When to synchronize the station to the VC

Synchronizing the station to the VC updates the RAPID program of the virtual controller with the latest changes in the station. This is useful to do before:

- Performing a simulation.
- Saving a program to files on the PC.
- Copying or loading RobotWare systems.

To synchronize a station to the VC, see [Synchronize to VC on page 415](#).

---

### When to synchronize the VC to the station

Synchronizing the VC to the station creates paths, targets and instructions that correspond to the RAPID program in the system running on the virtual controller. This is useful to do when you have:

- Started a new virtual controller which system contains existing programs.
- Loaded a program from a file.
- Text-edited the program.

To synchronize the VC to a station, see [Synchronize to Station on page 414](#).

## 4 Simulating programs

### 4.1 Simulation Overview

#### About this chapter

This chapter describes how to simulate and validate robot programs. Below are short introductions to the simulation functions in RobotStudio.

Function	Description
Play simulations	Simulations run entire robot programs on a virtual controller. Before you run a simulation you need to decide which paths are to be simulated. To set up a simulation, see <a href="#">Simulation Setup on page 329</a> . To run a simulation, see <a href="#">Simulation Control on page 340</a> .
Collision detection	Collision detection displays and logs collisions and near-misses for specified objects in the station. Normally used during simulation of robot programs, it can also be used when building the station. For more information, see <a href="#">Detecting collisions on page 137</a> .
Event handling	Events can be used to connect an action to a trigger. For example, you can attach one object to another when they collide or a signal is set. For more information, see <a href="#">Creating an event on page 140</a> .
I/O Simulation	In simulations I/O signals are normally set either by the robot program or by events. With the I/O simulator you can set signals manually, which provides a quick test of specific conditions. For more information, see <a href="#">Simulating I/O signals on page 141</a> .
Simulation Monitoring	With the simulation monitoring functions you enhance the simulation by adding traces along the TCP movements or alerts triggered by defined speeds or motions. For more information, see <a href="#">Enabling simulation monitoring on page 142</a> .
Process time measurement	With the process timer you measure the time for a process to complete. For more information, see <a href="#">Measuring simulation time on page 143</a> .

#### Time handling during simulation

When simulating stations with events or several controllers, or other time managing equipment, time can be managed in two modes: either as *free runtime* or as *time slices*. RobotStudio uses time slice mode by default, but you can switch to free runtime, if required.

#### Free runtime

Since all controllers use the same computer resources, their synchronization might not be exactly as in the real world if they run independently of each other (called *free run* mode). The cycle time will be correct, but the timing for setting signals and triggering events might be inaccurate.

*Continues on next page*

## 4 Simulating programs

---

### 4.1 Simulation Overview

#### *Continued*

#### Time Slice

Time slices can be used to ensure that the timing for signals and other interaction between controllers is accurate. In this mode, RobotStudio synchronizes the controllers by dividing a time segment into small slices and waiting for all controllers to complete a current time slice before any controller can start anew. Thus, the controllers are synchronized, and the cycle time will be calculated correctly. The drawbacks are that the virtual FlexPendant cannot be open, and that the simulation might be somewhat slow and jerky, depending on the complexity of the simulation and the performance of the computer.



#### **Note**

If the simulation uses events or involves several different controllers, the virtual time mode **Time Slice** shall be used to make sure that the timing between the controllers is correctly simulated.

---

## 4.2 Detecting collisions

---

### Overview

With RobotStudio you can detect and log collisions between objects in the station. The basic concepts of collision detection are explained below.

---

### Collision sets

A collision set contains two groups, *Objects A* and *Objects B*, in which you place the objects to detect any collisions between them. When any object in *Objects A* collides with any object in *Objects B*, the collision is displayed in the graphical view and logged in the output window. You can have several collision sets in the station, but each collision set can only contain two groups.

A common use of collision sets is to create one collision set for each robot in the station. For each collision set you then put the robot and its tool in one group and all objects you do not want it to collide with in the other. If a robot has several tools, or holds other objects, you can either add these to the robot's group as well or create specific collision sets for these setups.

Each collision set can be activated and deactivated separately.

---

### Collisions and near-misses

In addition to collisions, the collision detection can also watch for near-misses, which is when an object in *Objects A* comes within a specified distance from an object in *Objects B*.

---

### Recommendations for collision detection

In general, the following principles are recommended to facilitate collision detection:

- Use as small collision sets as possible, splitting large parts and collecting in the collision sets only relevant parts.
- Enable coarse detail level when importing geometry.
- Limit the use of near-miss.
- Enable last collision detection, if the results are acceptable.

---

### Results of creating a collision set

After you have created a collision set, see [Create Collision Set on page 328](#), RobotStudio will check the positions of all objects and detect when any object in *ObjectsA* collides with any object in *ObjectsB*.

Activation of detection and display of collisions depend on how the collision detection is set up.

If the collision set is active, RobotStudio will check the positions of the objects in the groups, and indicate any collision between them according to the current color settings.

*Continues on next page*

## 4 Simulating programs

### 4.2 Detecting collisions

*Continued*

#### Collision detection

Collision detection checks whether robots or other moving parts collide with equipment in the station. In complex stations, you can use several collision sets for detecting collisions between several groups of objects.

After collision detection has been set up, it does not need to be started, but automatically detects collisions according to the setup.

#### Setting when to check for collisions

To set whether to detect collisions always or only during simulation, follow these steps:



- 1 On the **File** menu, click **Options**.
- 2 On the Navigation pane to the left, select **Simulation: Collision**.
- 3 On the Collision page to the right, select one of the following options from the **Perform collision detection**:

Option	Description
During simulation	Collision detection is active only during simulation (when running RAPID programs in the virtual controller).
Always	Collision detection is always active, even when moving objects manually or testing reachability.

#### Setting the objects for collision detection

To set the objects for collision detection, follow these steps:

- 1 Make sure that the objects for collision detection are placed correctly in collision sets.
- 2 Make sure that the collision set for the objects is activated, which is indicated by an icon in the Layout browser:

Icon	Description
 xx050033	Active. Collisions between objects in this set will be detected.
 xx050007	Not active. Collisions between objects in this set will not be detected.

To activate or deactivate collision sets, continue with the following steps:

- 3 Right-click the collision set to change and then click **Modify Collision set** to bring up a dialog box.
- 4 Select or clear the **Active** check box and then click **Apply**.

#### Setting near-miss detection

Near-misses occur when objects in collision sets are close to colliding. Each collision set has its own near-miss settings. For setting near-miss detection, follow these steps:

- 1 In the **Layout** browser, right-click the collision set to change and then click **Modify Collision set** to bring up a dialog box.

*Continues on next page*

- 2 In the **Near miss** box, specify the maximum distance between the objects to be considered a near-miss and then click **Apply**.

---

#### Setting logging options

In addition to the graphical display of collisions, you can also log the collisions to the output window or a separate log file:

- 1 On the **File** menu, click **Options**.and under **Simulation**, click **Collision**.
- 2 On the Navigation pane to the left, select **Simulation: Collision**.
- 3 On the Collision page to the right, select **Log collisions to Output window** check box.

The collision log is displayed in the output window.

- 4 On the Collision page to the right, select **Log collisions to file** check box and enter the name and path to the log file in the box.

A separate file for logging collisions is created below the check box.

## 4 Simulating programs

---

### 4.3 Creating an event

### 4.3 Creating an event

---

#### Overview

Events enhance your simulations by defining actions that are carried out when specific trigger conditions are fulfilled. You can use events to:

- Attach one object to another, for example, a work piece to a gripper when simulating material handling, see [Attaching and detaching objects on page 71](#).
- Set signals, for example, when simulating signals set by equipment other than the controller, see [Simulating I/O signals on page 141](#).
- Start or stop the process timer, see [Measuring simulation time on page 143](#).

Used for creating new events, the **Create New Event Wizard** is launched from the Event manager, see [Event Manager on page 332](#).

---

#### Prerequisites

Before creating the event, make sure that the station contains all signals and objects that are planned to be used as triggers or affected by the action.

#### 4.4 Simulating I/O signals

---

##### Procedures

When simulating I/O signals you can either create events that set signal values when specified trigger conditions are fulfilled, or you can set signal values manually.

For procedures using the event manager, see [Event Manager on page 332](#).

For procedures using the I/O simulator, see [I/O Simulator on page 341](#).

---

##### Related information

For information about controlling I/O signals from the RAPID program, see [Creating RAPID instructions for setting I/O signals on page 116](#).

## 4 Simulating programs

---

### 4.5 Enabling simulation monitoring

### 4.5 Enabling simulation monitoring

---

#### Overview

The simulation monitor commands are used to visually detect critical robot movements during simulation by drawing a colored line that follows the TCP.

---

#### To enable TCP tracing

To enable TCP tracing, follow these steps:

- 1 On the **Simulation** tab, click **Monitor** to bring up a dialog box.
- 2 In the left pane, select the appropriate robot.
- 3 On the **TCP Trace** tab, select the **Enable TCP Trace** check box. This activates TCP tracing for the selected robot.
- 4 Optionally, change the length and color of the trace. For detailed information, see [Monitor on page 343](#).

---

#### To enable simulation alerts

To enable simulation alerts, follow these steps:

- 1 On the **Simulation** menu, click **Monitor** to bring up a dialog box.
- 2 In the left pane, select the appropriate robot.
- 3 On the **Alerts** tab, select the **Enable Simulation Alerts** check box. This activates simulation alerts for the selected robot.
- 4 In the threshold value boxes, specify the threshold for the alerts. Setting the threshold to 0 is equivalent to disabling the alert. For detailed information, see [Monitor on page 343](#).

## 4.6 Measuring simulation time

---

### Stopwatch feature for measuring simulation time

The Simulation tab's **Stopwatch** feature is used for measuring the time taken between two trigger points in a simulation, and also for the simulation as a whole. The two trigger points are called the Start Trigger and the End Trigger.

When a stopwatch is setup, the timer starts when the Start Trigger occurs, and stops when the End Trigger occurs. The kinds of triggers you can specify are:

- Simulation Start
- Simulation Stop
- Target Changed

Additionally, specify the mechanical Unit and the target.

- I/O Value

Additionally, specify the source mechanical unit from where the signal comes, the type of I/O signal and the value of the signal.

You can several stopwatches setup for a simulation. You can also specify a different name for each stopwatch.

For information on how to use the Stopwatch feature, see [Stopwatch on page 344](#)



#### Tip

While you are on the *Simulation* tab, you can check the RobotStudio status bar for the Simulation Time, that is, from simulation start to simulation stop.

**This page is intentionally left blank**

## 5 Deploying and distributing

### 5.1 Copying programs

---

#### Overview

RAPID programs are normally stored in the systems that run on the virtual controllers of your station. To copy programs to systems on other controllers, save the programs to file on the PC and then load these files to the destination controllers. You can save either entire programs or specific modules.

---

#### Copying a program

To copy a program from one controller to another, follow these steps:

- 1 In the **Controller** browser, select the controller that contains the program to copy.
- 2 Save the program to file on the disc. For details, see [Saving a program on page 427](#).
- 3 If necessary, copy the files to a location that is accessible to the other controller.
- 4 For instructions on loading the program to a system on a virtual controller, a FlexController or a non-running system, see the table below.

System location	Do this
Virtual controller, running in RobotStudio	See <a href="#">Loading a RAPID program on page 427</a> .
FlexController	Connect to the FlexController and load the program.
A non-running system stored on the PC	Start the system in a virtual controller, then load the program, see <a href="#">Adding a system on page 85</a> and <a href="#">Loading a RAPID program on page 427</a> , respectively.

## 5 Deploying and distributing

---

### 5.2 Pack & Go / Unpack & Work

### 5.2 Pack & Go / Unpack & Work

---

#### Overview

The Pack & Go / Unpack & Work feature makes it possible to create a package (zip file) of an active station that can be unpacked on another computer. The package contains all necessary files, except media pools, but additional, option-based media pools are included.

For procedures, see [Pack and Go on page 191](#) and [Unpack and Work on page 192](#).

## 5.3 Screen Capture

### Overview

Screen capture entails two functions useful for demonstrations and training purposes:

- The Screenshot function which allows you to capture an image of the application.
- The Record Movie function which allows you to make a recording of your work in RobotStudio, either of the entire GUI or just the graphics window.

### Taking screenshots

The Screenshot function allows you to capture an image of the entire application or an active document window such as the graphics window.



#### Note

The Screenshot feature is available only for RobotStudio Premium users.

Configure the options for the screenshot function as per your requirement under Options:General:Screenshot in the File tab. For more information, see [Options:General:Screenshot on page 196](#).

You can take screenshots using the keyboard shortcut CTRL + B. Alternatively, you can use the **Screenshot** button on the Quick Access Toolbar, but you must enable it first.

To enable the screenshot command button:

- 1 Click the Quick Access Toolbar's down arrow. The Customize Quick Access Toolbar menu is displayed.
- 2 Click **Screenshot** to add the check mark to the command. This adds the screenshot button to the Quick Access Toolbar.

### Recording movies

You can record your activities in the RobotStudio application as a video. You can also record videos of simulations. For information on how to record videos in RobotStudio, see [Record Movie on page 353](#).

This page is intentionally left blank

## 6 Working online

### 6.1 Connecting a PC to the controller

---

#### General

In general there are two ways of physically connecting a PC to the controller, to the service port or to the factory network port.

#### The service port

The service port is intended for service engineers and programmers connecting directly to the controller with a PC.

The service port is configured with a fixed IP-address, which is the same for all controllers and cannot be changed, and has a DHCP server that automatically assigns an IP-address to the connected PC.

#### The factory network port

The factory network port is intended for connecting the controller to a network.

The network settings can be configured with any IP-address, typically provided by the network administrator.

---

#### Limitations



#### Note

The maximum number of connected network clients using Robot Communication Runtime is:

- LAN port: 3
- Service port: 1
- FlexPendant: 1

The maximum number of applications using Robot Communication Runtime running on the same PC connected to one controller has no built-in maximum. However, UAS limits the number of logged-on users to 50.

The maximum number of concurrently connected FTP clients is 4.

*Continues on next page*

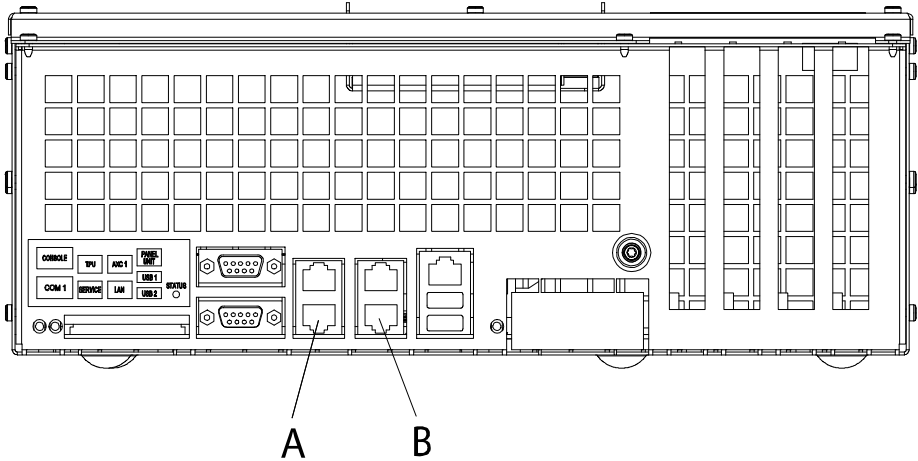
## 6 Working online

### 6.1 Connecting a PC to the controller

*Continued*

#### Ports on the computer unit DSQC 639

The illustration below shows the two main ports on the computer unit DSQC 639, the service port and the LAN port.



connecti

A	Service port on the computer unit (connected to the service port on the controller front through a cable).
B	LAN port on the computer unit (connects to factory network).



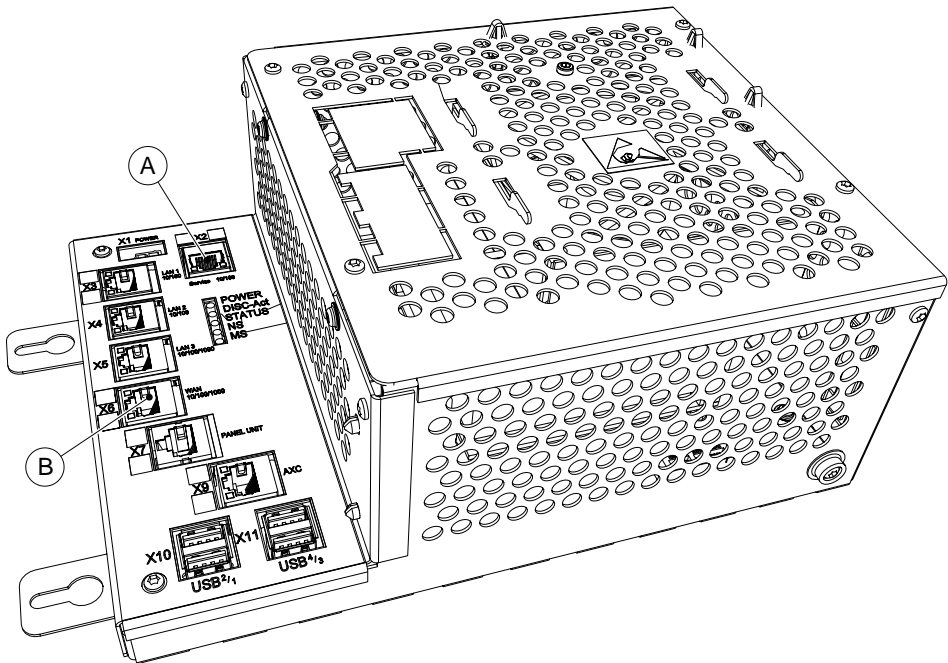
#### Note

The LAN port is the only public network interface to the controller, typically connected to the factory network with a public IP-address provided by the network administrator.

*Continues on next page*

Ports on the computer unit DSQC1000

The illustration below shows the two main ports on the computer unit DSQC1000, the service port and the WAN port.



xx1300000609

A	Service port on the computer unit (connected to the service port on the controller front through a cable).
B	WAN port on the computer unit (connects to factory network).



Note

The WAN port is the only public network interface to the controller, typically connected to the factory network with a public IP-address provided by the network administrator.

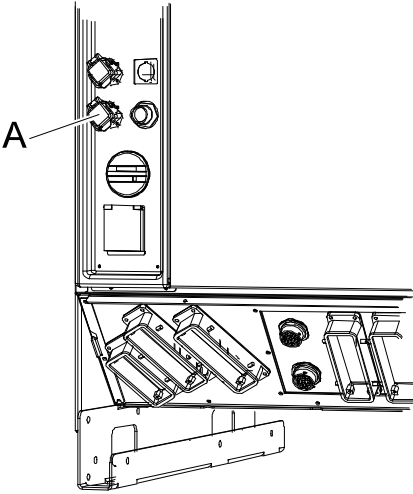
LAN1, LAN2, and LAN3 can only be configured as private networks to the IRC5 controller.

## 6 Working online

### 6.1 Connecting a PC to the controller

*Continued*

#### Connecting a PC to the controller

	Action	Note
1	<p>Make sure that the network setting on the PC to be connected is correct.</p> <p><b>When connecting to the service port:</b></p> <ul style="list-style-type: none"><li>The PC must be set to “Obtain an IP address automatically” or set as described in <b>Service PC Information</b> in the <b>Boot Application</b> on the FlexPendant.</li></ul> <p><b>When connecting to the factory network port:</b></p> <ul style="list-style-type: none"><li>The network settings for the PC depend on the network configuration setup by the network administrator.</li></ul>	<p>Refer to the system documentation for your PC, depending on the operative system you are running.</p> <p>For more information, see <a href="#">Network settings on page 153</a>.</p>
2	Connect a network cable to the network port of your PC.	
3	<p><b>When connecting to the service port:</b></p> <ul style="list-style-type: none"><li>Connect the network cable to the service port on the controller, or to the service port on the computer unit.</li></ul> <p><b>When connecting to the factory network port:</b></p> <ul style="list-style-type: none"><li>Connect the network cable to the factory network port on the computer unit.</li></ul>	 <p>connectb</p> <p>A Service port on the controller</p>

## 6.2 Network settings

### Overview

This topic describes the network settings for a PC connected to a controller, which is a prerequisite for working online.

You can connect the PC to the controller through an Ethernet network in the following ways:

- Local network connection
- Service port connection
- Remote network connection

### Local network connection

You can connect your PC to the same Ethernet network that the controller is connected to. When the PC and the controller are connected correctly and to the same subnet, the controller will be automatically detected by RobotStudio.

The network settings for the PC depend on the network configuration. For setting up the PC, contact the network administrator.

### Service port connection

When connecting to the controller's service port, you can either obtain an IP address for the PC automatically, or you can specify a fixed IP address.

If you are not sure how to set up the service port connection, contact the network administrator.

#### Automatic IP address

The controller's service port has a DHCP server that will automatically give your PC an IP address if it is configured for this. For detailed information see Windows help on configuring TCP/IP.

#### Fixed IP address

Instead of obtaining an IP address automatically, you can also specify a fixed IP address on the PC you connect to the controller.

Use the following settings for a fixed IP address:

Property	Value
IP address	192.168.125.2
Subnet mask	255.255.255.0

For detailed information about how to set up the PC network connection, see Windows help on configuring TCP/IP.

*Continues on next page*

## 6 Working online

### 6.2 Network settings

*Continued*



#### Note

Obtaining an IP address automatically might fail if the PC already has an IP address from another controller or Ethernet device.

To ensure that you get a correct IP address if the PC was previously connected to an Ethernet device, do one of the following:

- Restart the PC before connecting to the controller.
- Run the command `ipconfig /renew` from the command prompt after connecting the PC to the controller.

#### Remote network connection

To enable a connection to a controller on a remote subnet or over the local network, the relevant network traffic must be allowed through any firewalls between the PC and the controller.

The firewalls must be configured to accept the following TCP/IP traffic from the PC to the controller:

- UDP port 5514 (unicast)
- TCP port 5515
- Passive FTP

All TCP and UDP connections to remote controllers are initiated by the PC, that is the controller only responds on the given source port and address.

#### Firewall settings

The firewall settings are applicable irrespective of whether you are connected to a real controller or a virtual controller.

The following table describes the necessary firewall configurations:

Status	Name	Action	Direction	Protocol	Remote Address	Local Service	Remote Service	Application
	RobNetscan-Host	Allow	Out	UDP/IP	Any	Any	5512,5514	robnetscan-host.exe
	IRS5Controller	Allow	In	UDP/IP	Any	5513	Any	robnetscan-host.exe
	RobComCtrlServer	Allow	Out	TCP/IP	Any	Any	5515	robcomctrlserver.exe
	RobotFTP	Allow	Out	TCP/IP	Any	Any	FTP(21)	Any

In addition, the following table describes the necessary firewall configurations for the RobotWare option, Integrated Vision:

Status	Name	Action	Direction	Protocol	Remote Address	Local Service	Remote Service	Application
	Telnet	Allow	Out	TCP/IP	Any	Any	23	RobotStudio.exe
	In-Sight Protocol	Allow	Out	TCP/IP	Any	Any	1069	RobotStudio.exe

*Continues on next page*

Status	Name	Action	Direction	Protocol	Remote Address	Local Service	Remote Service	Application
	In-Sight Discovery	Allow	In/Out	UDP/IP	Any	1069	1069	RobotStudio.exe
	Upgrade port (PC only)	Allow	Out	TCP/IP	Any	Any	1212	RobotStudio.exe
	DataChannel	Allow	Out	TCP/IP	Any	Any	50000	RobotStudio.exe

**Note**

RobotStudio uses the current Internet Options, HTTP, and proxy settings to get the latest RobotStudio news. To view the latest RobotStudio news, go to the **File** tab and then the **Help** sub-tab.

**Connecting to the controller**

- 1 Make sure the PC is connected to the controller's service port and that the controller is running.
- 2 On the **File** menu, click **Online** and then select **One Click Connect**.  
This takes you to the **Controller** tab.
- 3 Click **Add Controller**
- 4 Click **Request Write access**.

If the controller is in mode	Then
Auto	You will now get Write Access if it is available.
Manual	A message box on the FlexPendant will allow you to grant remote Write Access to RobotStudio.

### 6.3 User Authorization

---

#### Overview

This section describes the controller's User Authorization System (UAS), which restricts what different users are allowed to do with the robot. This is for protecting data and functionality from unauthorized use.

The user authorization is managed by the controller, which means that the UAS settings remain for the controller regardless of which system it is running. It also means that the UAS settings apply to all tools for communicating with the controller, like RobotStudio or the FlexPendant. The UAS settings defines the users and groups that can access the controller, and what actions they are granted access to.

For procedures, see [User Accounts on page 395](#).

#### Users

UAS users are accounts with which persons log on to the controller. Furthermore, the users are added to groups to which access grants are given.

The users are defined in the controller by a user name and a password. For logging on to a controller, the user must type in a defined user name with a correct password.

A user can either have the state activated or deactivated in the UAS. When a user is deactivated it is not possible to log on to the controller using that account. It is the UAS administrator that activates and deactivates the users.

#### The Default user

All controllers have a default user named *Default User* with a publicly known password *robotics*. The *Default User* cannot be removed and the password cannot be changed. However, the user having the grant *Manage UAS settings* can modify the controller grants and application grants of the *Default User*.

#### Groups

In UAS, groups are defined sets of grants for accessing the controller. To the groups you then add the users who shall have the grants defined by the group.

A good practice is to create groups that resembles the professions that work with the robots in your organization. For example, you can create groups for administrators, programmers and operators.

#### The Default group

All controllers have a default group named *Default Group*, to which all grants are given and to which the default user belongs. This group cannot be removed, but it can be changed by the user having the grant *Manage UAS settings*.

---

*Continues on next page*



#### Note

There is a risk changing the group membership of the default user. If you by mistake clear the *Default User* check box or any *Default Group* grant, you will get a warning. Make sure that there is at least one user defined that has the grant *Manage UAS settings*. If the *Default group* and no other group have the grant *Manage UAS settings*, you may lose your ability to manage users and groups.

---

## Grants

Grants are permissions to perform actions or access data on the controller. You use the grants by giving them to groups, to which you then add the users who shall have the grants.

Grants may be either *controller grants* or *application grants*. Depending on the actions that you will perform, you may need several grants. For procedures, see [UAS Grant Viewer on page 400](#).

### Controller grants

Controller grants are validated by the robot controller and apply to all tools and devices accessing the controller.

### Application grants

Application grants are used by a specific application, for example the FlexPendant, and will only be valid using that application. Application grants can be added by additional options and used in customer applications.

## 6.4 The System Builder

### 6.4.1 System Builder Overview

---

#### Overview

This section describes how you create, build, modify and copy systems to run on virtual and real controllers. These systems may even be converted to boot media and downloaded to a real controller.

The system points out the robot models and options to use; it also stores configurations and programs for the robots. Therefore, it is good practice to use a unique system for each station even if the stations use the same basic setup. Otherwise, changes in one station may accidentally overwrite data used in another station.

---

#### About virtual and real systems

The system you run on virtual controllers can either be a real system built on real RobotWare keys or a virtual system built on virtual keys.

When using real systems, the RobotWare keys define which options and robot models shall be used, thus helping you to configure the system correctly. Real systems can be run both on virtual controllers and real IRC5 controllers.

When using virtual keys, all options and robot models are available, which is useful for evaluation purposes, but requires more configuration when creating the system. Systems built on virtual keys can only be run on virtual controllers.

---

#### Prerequisites

Creating a system entails applying a predefined template to a station, reusing an existing system or letting RobotStudio propose a system based on a layout.

To create a system, the following conditions must be met:

- The RobotWare media pool must be installed on your PC.
- You must have a RobotWare key for the system, if creating a system to run on a real controller. The RobotWare key is a license key that determines which robot models to use and which RobotWare options to run on the controller. The license key is delivered with the controller.
- If you want to create a system for virtual use only, you can use a virtual key instead. Virtual keys are generated by the wizard. When using virtual keys, you select the robot models and options to use in the *Modify Options* section of the wizard.
- Downloading to the real controller requires a direct connection from your computer to the service or Ethernet port of the controller.

---

#### Administering systems

Systems can be administered from the **System Builder** dialog box in the following ways:

- View system properties, see [Viewing system properties on page 160](#).
- Build a system, see [Building a new system on page 161](#).

*Continues on next page*

- Modify or delete a system, see [Modifying a system on page 165](#).
- Copy a system, see [Copying a system on page 169](#).
- Create a system from backup, see [Creating a system from backup on page 170](#).
- Download a system to a controller, see [Downloading a system to a controller on page 171](#).
- Create boot media, see [Creating boot media on page 172](#).

### 6.4.2 Viewing system properties

---

#### Overview

All systems you create with the System Builder are stored locally on your computer. It is recommended that you store them in one or more dedicated system directories.

#### Viewing system properties

To view system properties and add comments, follow these steps:

- 1 In the **System Builder** dialog box, select a system from the **Systems** box.  
If necessary, in the **System directory** list, you can navigate to the folder in which your systems are stored
- 2 The system properties are then displayed in the **System Properties** box. Optionally, type a comment in the **Comments** box, and click **Save**.

### 6.4.3 Building a new system

---

#### Overview

The **New Controller System Wizard**, used for building a new system, is launched from the System Builder.

---

#### Starting the wizard

To start the wizard, follow these steps:

- 1 Click **System Builder** to bring up a dialog box.
- 2 In the **Actions** group, click **Create New**. This starts the wizard.
- 3 Read the information on the welcome page and click **Next**.

---

#### Specifying the name and location

To determine where on your computer to store the system you are creating, follow these steps:

- 1 In the **Name** box, enter a name for the system you are creating.
- 2 In the **Path** box, enter the path to the system directory in which you will store the system.  
You can also click the **Browse** button and browse to the system directory.
- 3 Click **Next**.

---

#### Entering the RobotWare keys

The RobotWare keys determine which RobotWare versions and parts to use in the system.

Creating a system to run on either IRC5 controllers or virtual controllers requires at least two keys: one for the controller module and one for each drive module in the cabinet. The keys are delivered together with the controller.

For creating a system to run on virtual controller only (for example, in Virtual IRC5), you can use virtual keys. Virtual keys give access to all options and robot models, but limits the use of the system to virtual controllers only.

To enter the key for the controller module, follow these steps:

- 1 In the **Controller Key** box, enter the controller key. You can also click **Browse** and browse to the key file. If creating a system for virtual use only, select the **Virtual Key** check box, and the controller key will be generated by the wizard.
- 2 In the **Media Pool** box, enter the path to the media pool. You can also click **Browse** and browse to the folder.system
- 3 In the **RobotWare Version** list, select which version of the RobotWare you want to use. Only RobotWare versions that are valid for the used key are available.
- 4 Click **Next**.

*Continues on next page*

---

#### Entering the drive keys

To enter the keys for the drive modules:

- 1 In the **Drive Key** box, enter the key for the drive module. You can also click the **Browse** button and browse to the key file. If you used a virtual controller key, a virtual drive key is already generated by the wizard.
- 2 Click the right arrow button next to the **Drive Key** box. The key now appears in the **Added drive keys** list.

For real systems the drive key determines the connected robot model. For virtual systems you select the robot model in the *Modify Options* page. The default model is IRB140.

- 3 If you have a MultiMove system, repeat steps 1 and 2 for each drive key to add.

If you have a MultiMove system, make sure that the keys are numbered in the same way as their corresponding drive modules are connected to the controller module. Use the up and down arrows to rearrange the drive keys, if necessary.

- 4 If you want to create the system as it is now, click **Finish**.

If you want to modify options, or add options, parameter data or additional files to the home directory, click **Next**.

---

#### Adding additional options

Here you can add options, such as external axes and dispense applications, that are not included in the basic system. Options require a license key and must be first imported to the media pool. To add additional options, follow these steps:

- 1 In the **Key** box, enter the option key. You can also click the **Browse** button and browse to the option's key file.
- 2 Click the **Arrow** button.

The option that the key unlocks is now displayed in the **Added Options** list.



#### Note

If several versions of an additional option exists, only the latest version can be selected. To use an older version, remove the other versions of the additional option from the Mediapool.

- 3 Repeat steps 1 and 2 for all options you want to include.
- 4 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

If you want to modify options, or add parameter data or additional files to the home directory, click **Next**.

**Modifying options**

Here you can set up and configure the options in your system. For virtual systems, you also select the robot models to use. To modify any options, follow these steps:

- 1 In the **Option** tree, expand the option folders to the level where you find the option you want to modify.  
Only the options unlocked by the used keys are available.
- 2 Modify the option.
- 3 Repeat steps 1 and 2 for all options you want to modify.
- 4 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

If you want to add parameter data or additional files to the home directory, click **Next**.

---

**Adding parameter data**

Parameter data is stored in the parameter data files (.cfg files). Each parameter topic has its own parameter file. You can add only one parameter file for each topic. To add parameter data, follow these steps:

- 1 In the **Parameter data** box, enter the path to the folder for the parameter data files. You can also click the **Browse** button and browse to the folder.
- 2 In the list of parameter data files, select the file you want to include and press the **Arrow** button. Repeat for all files you want to include.

The included parameter data files will now appear in the **Added parameter data files** list.

Repeat steps 1 and 2 for each parameter data file you want to add.

- 3 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

If you want to add additional files to the home directory, click **Next**.

---

**Adding files to the home directory**

You can add any type of file to the system's home directory. When the system is loaded to a controller, these files will also be loaded. To add files to the system's home directory, follow these steps:

- 1 In the **Files** box, enter the path to the folder for the files you want to include. You can also click the **Browse** button and browse to the folder.
- 2 In the list of files, select the file to add and click the **Arrow** button. Repeat for all files you want to add.

The added files will now appear in the **Added files** list.

- 3 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

If you want to read a summary before you create the system, click **Next**.

---

*Continues on next page*

## 6 Working online

---

### 6.4.3 Building a new system

*Continued*

---

#### Completing the New Controller System Wizard

To complete the wizard, follow these steps:

- 1 Read the system summary.
- 2 If the system is OK, click **Finish**.

If the system is not OK, click **Back** and make modifications or corrections.

## 6.4.4 Modifying a system

---

### Overview

The **Modify Controller System Wizard**, used to modify existing systems, is launched from the System Builder. The wizard helps you with tasks like changing robots, adding and removing external axes and other options. A system that is running must be first shut down before modification.

---

### Starting the wizard

To start the wizard when creating a new station:

- 1 If the system is currently running, on the **Controller** menu, point to **Shutdown** and then click **Shutdown**.
- 2 On the **Controller** menu, click **System Builder** to bring up a dialog box.
- 3 In the **System directory** list, enter or browse to the system directory. Select a system from the list beneath, review the system properties and add and save any comments.
- 4 In the **Actions** group, click **Modify**. This starts the wizard.
- 5 Read the information on the welcome page and click **Next**.

---

### Modifying the program revision

The RobotWare versions that are available for the system are determined by the controller key. The key is essential to the system and cannot be modified.

To use another RobotWare version than the available ones, create a new system with another key.

To optionally modify the program revision, follow the appropriate step or steps:

- 1 To keep the current RobotWare version, select **Yes** and then click **Next**.
- 2 To replace the current RobotWare version, Select **No, replace it..**
- 3 In the **Media pool** box, enter the path to the media pool. You can also click the **Browse** button and browse to the folder.
- 4 In the **New program revision** box, select which version of RobotWare you want to use. Only RobotWare versions that are valid for the RobotWare key are available.
- 5 Click **Next**.

---

### Adding or removing drive keys

The drive key corresponds to the drive modules in your controller. For MultiMove systems, you have one drive module (and one key) for each robot. The keys for your system are delivered together with the controller.

the system is created with a virtual controller key, virtual drive keys are generated by the wizard. when you have added one virtual drive key for each robot, you select which robot to use for each key on the *Modify Options* page.

To optionally add or remove the keys for the drive modules, follow these steps:

- 1 To add a key for a drive module, enter the key in the **Enter Drive Key** box. You can also click the **Browse** button and browse to the key file.

*Continues on next page*

## 6 Working online

---

### 6.4.4 Modifying a system

*Continued*

- 2 Click the right arrow button. The key now appears in the **Added drive key** list.  
If you have a MultiMove system, repeat steps 1 and 2 for each drive key to add.
- 3 To remove a drive module, select the corresponding key in the **Added drive key** list and click **Remove drive key**.  
If you have a MultiMove system, repeat step 3 for each drive key to remove.
- 4 If you have a MultiMove system, make sure that the keys are numbered in the same way as their corresponding drive modules are connected to the controller module. Use the up and down arrows to rearrange the drive keys, if necessary.
- 5 Choose whether you want to create the system as it is now, or to continue with the wizard.  
If you want to create the system as it is now, click **Finish**.  
If you want to modify options, parameter data or add files to or remove files from the home directory, click **Next**.

---

#### Adding or removing additional options

To optionally add or remove additional options:

- 1 To add an additional option, in the **Enter Key** box, enter the option key. You can also click the **Browse** button and browse to the option's key file.
- 2 Click the **Arrow** button.  
The option that the key unlocks is now displayed in the **Added Options** list.



#### Note

If several versions of an additional option exists, only the latest version can be selected. To use an older version, remove the other versions of the additional option from the Mediapool.

- 3 Repeat steps 1 and 2 for all options you want to include.
- 4 To remove an additional option, in the **Added options** list, select the option you want to remove.
- 5 Click **Remove**.
- 6 Choose whether you want to create the system as it is now, or to continue with the wizard.  
If you want to create the system as it is now, click **Finish**.  
If you want to modify parameter data or add files to or remove files from the home directory, click **Next**.

---

#### Modifying options

To optionally modify any options, follow these steps:

- 1 In the **Option** tree, expand the option folders to the level where you find the option you want to modify.  
Only the options unlocked by the used keys are available.

*Continues on next page*

- 2 Modify the option.
- 3 Repeat steps 1 and 2 for all options you want to modify.
- 4 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

If you want to modify parameter data or add files to or remove files from the home directory, click **Next**.

---

#### Adding or removing parameter data

Parameter data is stored in the parameter data files (.cfg files). Each parameter topic has its own parameter file. You can add only one parameter file for each topic. To add or remove parameter data, follow these steps:

- 1 To add parameter data, in the **Parameter data** box, enter the path to the folder for the parameter data files. You can also click the **Browse** button and browse to the folder.
- 2 In the list of parameter data files, select the file you want to include and press the **Arrow** button. Repeat for all files you want to include.

The included parameter data files will now appear in the **Added parameter data files** list.

Repeat steps 1 and 2 for each parameter data file you want to add.

- 3 To remove parameter data, in the **Added parameter data files** list, select the parameter data file to remove.
- 4 Click **Remove**.
- 5 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

If you want to add to or remove files from the home directory, click **Next**.

---

#### Add files to or remove files from the home directory

You can add any type of file to the system's home directory, or remove files from it. When the system is loaded to a controller, these files will also be loaded. To optionally add files to or remove files from the system's home directory, follow these steps:

- 1 To add files, in the **Files** box, enter the path to the folder for the files you want to include. You can also click the **Browse** button and browse to the folder.
- 2 In the list of files, select the file to add and click the **Arrow** button. Repeat for all files you want to add.

The added files will now appear in the **Added files** list.

- 3 To remove files, in the **Added files** list, select the file to remove.
- 4 Click **Remove**.
- 5 Choose whether you want to create the system as it is now, or to continue with the wizard.

If you want to create the system as it is now, click **Finish**.

*Continues on next page*

## 6 Working online

---

### 6.4.4 Modifying a system

*Continued*

If you want to read a summary before you create the system, click **Next**.

---

#### Complete the Modify Controller System wizard

To complete the wizard, follow these steps:

- 1 Read the system summary.
- 2 If the system is OK, click **Finish**.

If the system is not OK, click **Back** and make modifications or corrections.

---

#### Result

Modifications will take effect when the wizard is completed.

If the system has been downloaded to a controller, it must be downloaded again before the modifications will take effect on the controller.

If the system is used by a VC, perform an I-start for the changes to take effect.

---

#### Deleting a system

To delete a system, follow this steps:

- 1 From the **System Builder** dialog box, select the system and then click **Delete**.

#### 6.4.5 Copying a system

---

##### Copy a system

To copy a system, follow these steps:

- 1 From the **System Builder** dialog box, select the system and then click **Copy** to bring up a dialog box.
- 2 Enter a name for the new system and a path, and then click **OK**.

#### 6.4.6 Creating a system from backup

---

##### Overview

The **Create System from Backup Wizard**, which creates a new system from a controller system backup, is launched from the System Builder. In addition, you can change the program revision and options.

---

##### Starting the wizard

To start the wizard, follow these steps:

- 1 From the **System Builder** dialog box, click **Create from Backup**. This starts the wizard.
  - 2 Read the information on the welcome page and click **Next**.
- 

##### Specifying the name and location

To specify the destination folder, follow these steps:

- 1 In the **Name** box, enter a name for the system you are creating.
  - 2 In the **Path** box, enter the path to the system directory in which you will store the system.  
You can also click the **Browse** button and browse to the system directory.
  - 3 Click **Next**.
- 

##### Locating the backup

To locate a system from backup, follow these steps:

- 1 In the **Backup folder** box, enter the path to the backup folder. Alternatively, click the **Browse** button to browse to it. Click **Next**.
- 2 In the **Media Pool** box, enter the path to the media pool containing the appropriate RobotWare program. Confirm the backup information that now appears in the wizard. Click **Next**.

## 6.4.7 Downloading a system to a controller

### Overview

All systems you access from the System Builder are stored on your computer. If you wish to run a system on a robot controller, you must first load it to the controller, which then requires a restart.

### Load a system

To load a system to a controller, follow these steps:

- 1 From the System Builder dialog box, select a system and then click **Download to Controller** to bring up a dialog box.



#### Note

Systems with incompatible hardware versions will not be displayed in the **Download to Controller** dialog box.

- 2 Specify the Destination Controller for the system.

You can select by using the...	if...
Select controller from list option	the controller has been detected automatically.
Specify IP address or controller name option	your PC and the robot is connected to the same network. You can only use the controller name in DHCP networks.
Use service port option	your PC is directly connected to the controller's service port.

- 3 Optionally, click **Test Connection** to confirm that the connection between the computer and the Controller is OK.
- 4 Click **Load**.
- 5 Answer **Yes** to the question **Do you want to restart the controller now?**

Yes	The controller restarts immediately and the downloaded system starts automatically.
No	The controller does not restart immediately. To start using the downloaded system, you have to: <ol style="list-style-type: none"> <li>a perform a C-start or an X-start</li> <li>b select the system manually</li> </ol>
Cancel	The downloaded system is removed from the controller.

### 6.4.8 Creating boot media

---

#### Overview

Boot media is an entire system which the System Builder packs to a single file and commonly stores on a hard disk or USB memory. The controller then accesses the file through its Ethernet port or USB port, respectively.

---

#### Creating boot media

To create boot media, follow these steps:

- 1 From the **System Builder** dialog box, create a new system. For creating a new system, see [Building a new system on page 161](#).
  - 2 From the **System Builder** dialog box, select a new system or an existing system and then click **Boot Media**.
  - 3 In the **Path** box, enter the path to the folder where you want to store the boot media file. Alternatively, browse to the location.
  - 4 Click **OK**.
- 

#### Result

To load the boot media system to a controller, first connect it and then restart the controller with the advanced restart method X-start.

## 6.4.9 Examples using the System Builder when offline

### 6.4.9.1 A MultiMove system with two coordinated robots

---

#### Overview

In this example we will use the System Builder to create a coordinated offline system with one IRB2400 and one IRB1600 robot to use in a new RobotStudio station.

---

#### Starting the New Controller System Wizard

To create a system like the one described above, follow these steps:

- 1 Click **System Builder** to bring up the dialog box.
- 2 In the dialog box, click **Create New** to bring up the **New Controller System Wizard**.
- 3 Read the welcome text, and click **Next** to continue to the next page.

---

#### Entering the name and path

- 1 In the **Name** box, enter the name of the system. The name must not contain blank spaces or non-ASCII characters.  
In this example, name the system *MyMultiMove*.
- 2 In the **Path** box, enter the path for the folder to save the system in, or click the **Browse** button to browse to the folder or create a new one.  
In this example, save the system in *C:\Program Files\ABB Industrial IT\Robotics IT\RobotStudio\ABB Library\Training Systems*.
- 3 Click **Next** to continue to the next page.

---

#### Entering the controller key

- 1 Select the **Virtual key** check box. A virtual controller key now appears in the Controller Key box. In this example we will use the default media pool and RobotWare version.
- 2 Click **Next** to continue to the next page.

---

#### Entering drive keys

- 1 Click the **Right Arrow** button next to the **Enter Drive key** box twice to create one drive key for each robot.
- 2 Click **Next** to continue to the next page.

---

#### Adding options

- 1 This system does not require any additional option keys. Click **Next** and continue to the next page of the wizard.

---

#### Modifying options

When creating robot systems from real robot keys, the key sets the options. But since we are using a virtual key, we have to set the options manually.

*Continues on next page*

To set the options necessary for a MultiMove, follow these steps:

- 1 Scroll down to the **RobotWare / Motion Coordination 1** group and select the **MultiMove Coordinated** check box.
- 2 Scroll down to the **RobotWare / I/O control** group and select the **Multitasking** and the **Advanced RAPID** check boxes.
- 3 Scroll down to the **DriveModule1 / Drive module application** group and expand the **ABB Standard manipulator** option. Select the **IRB 2400 Type A** option, manipulator variant **IRB 2400L Type A**.
- 4 Scroll down to the **DriveModule2 / Drive module application** group and expand the **ABB Standard manipulator** option. Select the **IRB 1600** option, manipulator variant **IRB 1600-5/1.2**.
- 5 Click **Finish** and the system will be created.

### 6.4.9.2 A system with support for one robot and one positioner external axis

#### Overview

In this example we will use the System Builder to create an offline system to use in a new RobotStudio station with one IRB1600 robot and one IRBP 250D positioner external axis.

#### Prerequisites

When creating systems for positioner external axes, you need the media pool and the license key file for that specific positioner. In this example we will use a media pool and license key file for a demo positioner.

Paths to files and folders assume that RobotStudio and the RobotWare media pool have been installed at their default locations on Windows XP. If not, adjust the paths accordingly.

#### Starting the New Controller System Wizard

To create a system like the one described above, follow these steps:

- 1 Click **System Builder** to bring up a dialog box.
- 2 In the dialog box, click **Create New** to bring up the **New Controller System Wizard**.
- 3 Read the welcome text, and click **Next** to continue to the next page.

#### Entering the controller key

- 1 Select the **Virtual key** check box. A virtual controller key now appears in the Controller Key box. In this example we will use the default media pool and RobotWare version.
- 2 Click **Next** to continue to the next page.

#### Entering drive keys

- 1 Click the **Right Arrow** button next to the **Enter Drive key** box to create one drive key for the robot.
- 2 Click **Next** to continue to the next page.

#### Adding options

This is where we point out the key file for the positioner.

- 1 Next to the **Enter key** box, click the browse button and select the key file.

In this example, browse to and select the file *extkey.kxt* in the folder  
*C:\Program Files\ABB Industrial IT\Robotics  
 IT\MediaPool\3HEA-000-00022.01.*



#### Tip

In the *MediaPool* folder media pools for several standard positioners are installed. They are named by the positioner's article number, with a suffix that indicates if it is configured for single-robot or MultiMove systems.

*Continues on next page*

- 2 Click the *Right Arrow* button next to the **Enter key** box to add the key for the positioner.
- 3 Click **Next** and continue to the next page of the wizard.

---

#### Modifying options

When creating robot systems from real robot keys, the key sets the options. But since we are using a virtual key, we have to set the options manually. To set the options necessary for a positioner, follow these steps:

- 1 Scroll down to the **RobotWare / Hardware** group and select the **709-x DeviceNet** check box.  
This option is for the communication between the controller and the track external axis.
- 2 Scroll down to the **DriveModule1 / Drive module application** group and expand the **ABB Standard manipulator** option. Select the **IRB 1600** option.  
This option sets the robot to an IRB 1600-5/1.2.
- 3 Scroll down to the **DriveModule1 > Drive module configuration** group; select the **Drive System 04 1600/2400/260** option; expand the **Additional axes drive module** group and select the **R2C2 Add drive** option.
  - a Expand the **Drive type in position Z4** group and select the **753-1 Drive C in pos Z4** option
  - b Expand the **Drive type in position Y4** group and select the **754-1 Drive C in pos Y4** option
  - c Expand the **Drive type in position X4** group and select the **755-1 Drive C in pos X4** option

This option adds drive modules for the positioner axes.



#### Note

When using the latest drive system, do the following:

Scroll down to the **DriveModule1 > Drive module configuration** group; select the **Drive System 09 120/140/1400/1600 Compact** option; expand the **Power supply configuration** group and select **1-Phase Power supply** or **3-Phase Power supply** (as applicable) > **Additional axes drive module > Additional drive**

- a Expand the **Drive type in position X3** group and select the **Drive ADU-790A in position X3** option
- b Expand the **Drive type in position Y3** group and select the **Drive ADU-790A in position Y3** option
- c Expand the **Drive type in position Z3** group and select the **Drive ADU-790A in position Z3** option

- 4 Click **Finish** and the system will be created. When starting the system in a RobotStudio station, you have to set up the system to load a model for the positioner and to get the motions to work properly. See [Placing external axes on page 98](#) for more information.

### 6.4.9.3 Options settings for systems with positioners

---

#### Overview

This is an overview of the RobotWare options to set when creating a system for positioner external axes. Note that besides setting the RobotWare options, you must add an additional option key for the positioner.

---

#### Media pools and option keys for the positioners

If you have the media pool and option key for your positioner, you can use these files.

If not, media pools for standard positioners are installed with RobotStudio. The path to these media pools in a default installation is: *C:\program files\ABB Industrial IT\Robotics IT\MediaPool*. In this folder a media pool for each positioner is located. These are named by the article number of the positioner, with a suffix that indicates if it is configured for a single-robot or a MultiMove system.

In the **Add additional options** page of the **System Builder**, you should add the option for the positioner by opening the mediapool folder for the positioner to add and selecting the *extkey.kxt* file.

---

#### Options for positioners in single-robot systems

When adding a positioner to a single-robot system, the positioner will be added to the same task as the robot. Below, the options to set on the **Modify Options** page of the **System Builder** for such a system are listed:

- **RobotWare > Hardware > 709-x DeviceNet > 709-1 Master/Slave Single**
- Optionally, for using the system with ArcWare also add **RobotWare > Application arc > 633-1 Arc**
- **DriveModule 1 > Drive module configuration > Drive System 04 1600/2400/260 > RC2C Add drive > 753-1 Drive C in pos Z4 > 754-2 Drive T in pos Y4 > 755-3 Drive U in pos X4**

---

#### Options for positioners in MultiMove robot systems

When adding a positioner to a MultiMove robot system, the positioner shall be added to a task of its own (thus you also have to add a drive key for the positioner). Below, the options to set on the **Modify Options** page of the **System Builder** for such a system are listed:

- **RobotWare > Hardware > 709-x DeviceNet > 709-1 Master/Slave Single**
- **RobotWare > Motion coordinated part 1 > 604-1 MultiMove Coordinated**  
Optionally, expand the MultiMove Coordinated option and select process options for the robots.
- Optionally, for using the system with ArcWare, add **RobotWare > Application Arc > 633-1 Arc**
- **DriveModule 1 > Drive module configuration > Drive System 04 1600/2400/260 > RC2C Add drive > 753-1 Drive C in pos Z4 > 754-2 Drive T in pos Y4 > 755-3 Drive U in pos X4**. For the other drive modules, no additional axes should be configured.

## 6 Working online

---

### 6.5 Handle I/O

### 6.5 Handle I/O

---

#### Overview

The I/O system handles input and output signals to and from the controller. Below are the parts of the system described, as well as common types of signals.

The I/O system window is used to view and set previously configured signals, and to activate and deactivate I/O units.

---

#### The I/O system

The I/O system of a controller consists of I/O buses, I/O units and I/O signals. The I/O buses are the controller's connections for I/O units (for instance I/O boards) and the I/O units contain channels for the actual signals.

The I/O buses and units are displayed in the robot view, as child nodes under each controller and the I/O signals are displayed in the I/O window.

---

#### I/O signals

I/O signals are used to communicate between the controller and external equipment, or to change variables within a robot program.

---

#### Input signals

Input signals notify something for the controller, for instance a feeder belt can set an input signal when it has positioned a work piece. The input signal can then be programmed to start a specific part of the robot program.

---

#### Output signals

The controller uses output signals to notify that a specified condition has been fulfilled. For instance, after the robot has finished its sequence, an output signal can be set. This signal can then be programmed to start a feeder belt, update a counter or trigger any other action.

---

#### Simulated signals

A simulated signal is a signal that is manually given a specific value that overrides the actual signal. Thus simulated signals might be useful for testing robot programs without activating or running equipment.

---

#### Virtual signals

Virtual signals are signals that are not configured to belong to a physical I/O unit. Instead, they reside inside the controller's memory. A common use for virtual signals is to set variables and store changes in a robot program.

---

#### Procedures

For using the I/O system window, see [Inputs / Outputs on page 361](#).

For adding a signal, see [Add Signals on page 372](#).

## 6.6 Configure systems

### Configuring system parameters

System parameters may be configured as follows:

- To view topics, types, instances and parameters
- To edit the parameters of an instance
- To copy and paste instances
- To add and delete instances
- To load and save complete configuration files to and from controllers

When working with configurations, the following tools, see [Configuration editor on page 372](#), are useful:

Tool	Use
The Configuration Editor	With the Configuration Editor you work with the types and instances of a specific topic.
The Instance Editor	With the Instance Editor, you specify the values of the parameters in the instances of system parameter types.



#### Note

To edit system parameters, you must have write access to the controller.

### Terms

<i>System parameters</i>	The sum of all parameters that configure the system, these are divided into topics and types.
<i>Topic</i>	A collection of parameters relating to a specific area, and the highest level in the system parameter structure. Examples are Controller, Communication and Motion.
<i>Type</i>	A set of parameters for a specific configuration task. A type can be seen as a pattern describing the construction and properties for the parameters included in the task. For instance, the type <i>Motion System</i> defines which parameters shall be used for configuring a motion system.
<i>Instance</i>	An actualization of a type, an instance is a specific set of parameters with unique values created from a type pattern. In the Configuration Editor, each row in the Instance list is an instance of the type selected in the Type list.
<i>Parameter</i>	A property to set when configuring the robot system.
<i>Configuration file</i>	Contains all public parameters of a specific topic.

### Viewing configurations

- 1 To view the topics of a controller, from the **Controller** tab, expand the **Configuration** node for the controller.  
All topics in are now displayed as child nodes to the Configuration node.
- 2 To view the types and instances of a topic, double-click the topic node for the topic to view.

*Continues on next page*

The Configuration Editor is now opened, listing all types of the topic in the **Type name** list. In the **Instance** list, each instance of the type selected in the **Type name** list is displayed as row. The parameter values of the instances are displayed in the columns of the instance list.

- 3 To view detailed parameter information for an instance, double-click the instance.

The instance editor now displays the current value, restrictions and limits of each parameter in the instance.

---

#### Editing parameters

You can either edit the parameters of one single instance, or you can edit several instances at one time. Editing several instances at one time is useful when you want to change the same parameter in several instances, like when moving signals from one I/O unit to another.

- 1 In the **Controller** tab, expand the **Controller** and the **Configuration** node and double-click the topic that contains the parameters to edit.

This opens the Configuration Editor.

- 2 In the **Type name** list of the Configuration Editor, select the type that the parameter to edit belongs to.

The instances of the type is now displayed in the Instance list of the Configuration Editor.

- 3 In the **Instance** list, select the instances to edit and press the Enter Key. To select several instances at once, hold down the SHIFT or CTRL key while selecting.

Alternatively, right-click an instance and then click **Edit**.

The Instance Editor is now displayed.

- 4 In the Parameter list of the Instance Editor, select the parameter to edit and change the value of the parameter in the **Value** box.

When editing several instances at one time, the parameter values you specify will be applied to all instances. For parameters that you do not specify any new value, each instance will keep its existing value for that parameter.

- 5 Click **OK** to apply the changes to the configuration database of the controller.

For many parameters, the changes will not take affect until the controller is restarted. If your changes require a restart, you will be notified of this.

You have now updated the controller's system parameters. If you are going to make several changes, you can wait with the restart until all changes are done.

---

**Adding instances**

With the Configuration Editor, you can select a type and create a new instance of it. For example, adding a new instance of the type Signal creates a new signal in the system.

- 1 In the **Controller** tab, expand the **Controller** and the **Configuration** node and double-click the topic that contains the type of which you want to add an instance.

This opens the Configuration Editor.

- 2 In the **Type name** list of the Configuration Editor, select the type of which you want to add an instance.
- 3 On the **Controller** menu, point to **Configuration** and click **Add type** (the word type is replaced by the type you selected previously).

You can also right-click anywhere in the configuration editor and then select **Add type** from the shortcut menu.

A new instance with default values is added and displayed in the **Instance Editor** window.

- 4 If required, edit the values.
- 5 Click **OK** to save the new instance.

The values in the new instance are now validated. If the values are valid, the instance is saved. Otherwise, you will be notified of which parameter values to correct.

For many instances, the changes will not take affect until the controller is restarted. If your changes require a restart you will be notified of this.

You have now updated the controller's system parameters. If the changes require a restart of the controller, the changes will not take affect until you do this. If you are going to make several changes, you can wait with the restart until all changes are done.

---

**Copying an instance**

- 1 In the **Controller** tab, expand the **Controller** and the **Configuration** node and double-click the topic that contains the instance to copy.

This opens the Configuration Editor.

- 2 In the **Type name** list of the Configuration Editor, select the type of which you want to copy an instance.
- 3 In the **Instance** list, select one or several instances to copy.

If you select several instances and they don't have the same value for all parameters, these parameters will have no default values in the new instances.

- 4 On the **Controller** menu, point to **Configuration** and click **Copy Type** (the word type is replaced by the type you selected previously).

You can also right-click the instance to copy and then select **Copy Type** from the shortcut menu.

A new instance with the same values as the one you copied is added and displayed in the **Instance Editor** window.

*Continues on next page*

## 6 Working online

---

### 6.6 Configure systems

*Continued*

- 5 Change the name of the instance. If required, also edit the other values.
- 6 Click **OK** to save the new instance.

The values in the new instance are now validated. If the values are valid, the instance is saved. Otherwise, you will be notified of which parameter values to correct.

For many instances, the changes will not take affect until the controller is restarted. If your changes require a restart you will be notified of this.

You have now updated the controller's system parameters. If the changes require a restart of the controller, the changes will not take affect until you do this. If you are going to make several changes, you can wait with the restart until all changes are done.

---

#### Deleting an instance

- 1 In the **Controller** tab, expand the **Controller** and the **Configuration** node and double-click the topic that contains the type of which you want to delete an instance.

This opens the Configuration Editor.

- 2 In the **Type name** list of the Configuration Editor, select the type of which you want to delete an instance.

- 3 In the **Instance** list, select the instance to delete.

- 4 On the **Controller** menu, point to **Configuration** and then click **Delete type** (the word type is replaced by the type you selected previously).

You can also right-click the instance to delete and then select **Delete type** from the shortcut menu.

- 5 A message box is displayed, asking if you want to delete or keep the instance. Click **Yes** to confirm that you want to delete it.

For many instances, the changes will not take affect until the controller is restarted. If your changes require a restart you will be notified of this.

You have now updated the controller's system parameters. If the changes require a restart of the controller, the changes will not take affect until you do this. If you are going to make several changes, you can wait with the restart until all changes are done.

---

#### Save one configuration file

The system parameters of a configuration topic can be saved to a configuration file, stored on the PC or any of its network drives.

The configuration files can then be loaded into a controller. They are thereby useful as backups, or for transferring configurations from one controller to another.

- 1 In the **Controller** tab, expand the **Configuration** node and select the topic to save to a file.
- 2 On the **Controller** menu, point to **Configuration** and select **Save System Parameters**.

You can also right-click the topic and then select **Save System Parameters** from the shortcut menu.

*Continues on next page*

- 3 In the **Save As** dialog box, browse for the folder to save the file in.
- 4 Click **Save**.

#### Saving several configuration files

- 1 In the **Controller** tab, select the **Configuration** node.
- 2 On the **Controller** menu, point to **Configuration** and click **Save System Parameters**.  
You can also right-click the configuration node and then click **Save System Parameters**.
- 3 In the **Save System Parameters** dialog box, select the topics to save to files. Then click **Save**.
- 4 In the **Browse for Folder** dialog box, browse for the folder to save the files in, and then click **OK**.  
The selected topics will now be saved as configuration files with default names in the specified folder.

#### Loading a configuration file

A configuration file contains the system parameters of a configuration topic. They are thereby useful as backups, or for transferring configurations from one controller to another.

When loading a configuration file to a controller, it must be of the same major version as the controller. For instance, you cannot load configuration files from an S4 system to an IRC 5 controller.

- 1 In the **Controller** tab, select the **Configuration** node.
- 2 On the **Controller** menu, point to **Configuration** and select **Load Parameters**.  
You can also right-click the configuration node and then select **Load Parameters** from the context menu.  
This opens the Select mode dialog box.
- 3 In the Select mode dialog box, select how you want to combine the parameters in the configuration file to load with the existing parameters:

If you want to	then
replace the entire configuration of the topic with the one in the configuration file.	select <b>Delete existing parameters before loading</b>
add new parameters from the configuration file to the topic, without modifying the existing ones.	click <b>Load parameters if no duplicates</b>
add new parameters from the configuration file to the topic and update the existing ones with values from the configuration file. Parameters that only exist in the controller and not in the configuration file will not be changed at all.	click <b>Load parameters and replace duplicates</b>

- 4 Click **Open** and browse to the configuration file to load. Then click **Open** again.
- 5 In the information box, click **OK** to confirm that you want to load the parameters from the configuration file.

*Continues on next page*

- 6 When the loading of the configuration file is finished, close the Select mode dialog box.

If a restart of the controller is necessary for the new parameters to take affect, you will be notified of this.

## 6.7 Handle events

### Overview

An event is a message that notifies you that something has happened to the robot system, be it merely a change in operation mode or a severe error that calls for your immediate attention. If the event requires any action from you, this is stated in the event.

Events are displayed in the event logs of the FlexPendant and RobotStudio.

The event log keeps you informed of system status, allowing you to:

- view controller events.
- filter events.
- sort events.
- get detailed information about an event.
- save event log files on your PC.
- clear event records.

### Event Log list

The event log list consists of all events matching your filter settings, with the following information for each event:

Type	The event type is an indication of the severity of the event.
Code	The event code is a number that identifies the event message.
Title	The event title is a short event description.
Category	The event category is an indication of the source of the event.
Seq. Number	The sequential number indicates the chronological order of the event.
Date and Time	Date and time is when the event occurred.

When you select an event in the list, detailed information will appear to the right.

### Event type

The event type is an indication of the severity of the event.

There are three types of events:

Event type	Description
Information	A normal system event, such as starting and stopping programs, change of operating mode, motors on/off and so on. Information messages never require any action from you, but can be useful for error tracking, statistics collecting or monitoring user triggered event routines.
Warning	An event that you need to be aware of, but not so severe that the process or RAPID program needs to be stopped. Warnings, however, often indicate underlying problems that sooner or later must be solved. Warnings must sometimes be acknowledged.
Error	An event that prevents the robot system from proceeding. The running process or RAPID program cannot continue, but is stopped. All errors must be acknowledged. Most errors also require some immediate action from you in order to solve the problem.

*Continues on next page*

## 6 Working online

---

### 6.7 Handle events

*Continued*



#### Note

This information is also indicated by color: blue for information, yellow for warning and red for an error which needs to be corrected in order to proceed

---

#### Event code

The event code is a number that identifies the event message. Together with the event date and time each event has a unique identity.

---

#### Event title

The event title is a short description of the event.

---

#### Event category

The category is an indication of the source of the event.

Category	Display
Common	All recent events.
Operational	Events related to changes in operation or operating mode.
System	Events related to the current system.
Hardware	Events related to controller hardware.
Program	Events related to the running process applications and RAPID programs.
Motion	Events related to the movement of robots or other mechanical units.
I/O & Communication	Events related to input and output signals, serial or network communication and process buses.
User	Custom messages that have been programmed into RAPID programs.
Internal	Internal low-level controller errors for ABB service personnel.
Process	Events related to Industrial Processes options., such as Spot, Arc and Dispense.
Configuration	Errors in a configuration file.
RAPID	Events related to Rapid instruction.

Depending on how the system is configured, additional categories may exist.

---

#### Seq number

The sequential number indicates the chronological order of the event; the higher the number the more recently the event occurred.

---

#### Date and Time

Date and time indicate exactly when the event occurred. Along with the event code, this timestamp guarantees that each event has a unique identity.

*Continues on next page*

**Event description**

When you select an event in the list, detailed description about the selected event is displayed on the right-hand side of the pane which contains a description section, consequences, causes and suggested actions to solve the problem.

**Overview**

The Event Log automatically logs all controller events once it is started. By default, events are displayed in the chronological order specified by **Seq Numbers**.

**Note**

Any modifications to the list you see will never affect the event log of the controller. What you see is just a copy.

**Managing events**

- 1 In the Robot View Explorer, select a system.
- 2 Double-click the **Event Log** node.

To sort events	Click the header for the column you want to sort by. To switch between ascending and descending sorting, click the header once again.
To filter events	In the <b>Category</b> list select the event category you want to be displayed.
To clear the event log	Click <b>Refresh</b> . This will not affect the event log of the robot controller. It might still be impossible, however, to retrieve all events from a cleared record once again, as the oldest ones may have been erased from the controller hard disk due to lack of space. It is therefore recommended to save the record to a log file before clearing.
To save all events to a single log file on the computer	Select the <b>Log to File</b> check box. If it remains checked, the log file will be updated with new events as they occur.
To save events of one or several categories to files on the computer	Click <b>Save</b> and then make your category choice. Specify the location for the log file(s) in the <b>Save As</b> dialog and then click <b>OK</b> . If you select <b>All</b> when selecting categories, a log file for each event category will be created.

**Retrieving controller events**

To clear the list and retrieve all existing events from the robot controller:

- 1 Optionally, save the existing Event Log record.
- 2 Select whether you want the list to be updated when new events occur , or if you are only interested in viewing events that have already occurred.

To ...	...then...
get automatic updates when new events occur	check the <b>Auto Update</b> check box. (Selected by default.)
say no to automatic updates when new events occur	clear the <b>Auto Update</b> check box.

*Continues on next page*

## 6 Working online

---

### 6.7 Handle events

*Continued*

- 3 To clear the current list, fetch and display all events that are currently stored in the controller log files.

## 7 File tab

### 7.1 Overview

#### Overview

The File tab contains the options to create new station, create new robot system, connect to a controller, save station as viewer, and RobotStudio options.

The following table lists the various options, presented in different tabs, available in the File tab:

Tabs	Description
Save / Save As	Saves a station.
Open	Opens a saved station.
Close	Closes a station.
Info	Once a station is open in RobotStudio, this tab shows the properties of the station, and also the robot systems and library files that are part of the open station.
Recent	Displays recently accessed stations.
New	Creates a new station. See <a href="#">New on page 190</a> .
Print	Prints the contents of the active window.
Share	Shares data with others. <ul style="list-style-type: none"> <li>• <a href="#">Pack and Go on page 191</a></li> <li>• <a href="#">Unpack and Work on page 192</a></li> <li>• <a href="#">Station Viewer on page 193</a></li> </ul>
Online	Connects to a controller. <ul style="list-style-type: none"> <li>• <a href="#">Add Controller on page 358</a></li> </ul> Imports and Exports controller. Creates and works with robot system. <ul style="list-style-type: none"> <li>• <a href="#">Building a new system on page 161</a></li> <li>• <a href="#">Import Options on page 388</a></li> </ul>
Help	Displays information on installing and licensing RobotStudio. See <a href="#">Installing and licensing RobotStudio on page 40</a> .
Options	Displays information on RobotStudio Options. See <a href="#">Options on page 195</a> .
Exit	Closes RobotStudio.

### 7.2 New

---

#### Creating an empty station

- 1 On the **File** tab, click **New**.
- 2 Click **Empty Station**, and then click **Create**.  
A new empty station is created.

---

#### Creating a station with Robot Controller

- 1 On the **File** tab, click **New**.
- 2 Click **Station with Robot Controller**.  
The available small, medium, large, paint and special purpose robots are listed.
- 3 From the list, click to select an appropriate robot. Alternatively, click **Browse** to browse for and select a system.  
RobotStudio automatically creates a matching virtual controller for your system.
- 4 Click **Create**.

---

#### Creating a station with existing Robot Controller

- 1 On the **File** tab, click **New**.
- 2 Select **Station with existing Robot Controller**.
- 3 In the **System Pool** drop-down list, select the folder which contains the system you need.  
The default System Pool path is  
C:\User\ABB\Documents\RobotStudio\System.  
You can also add a folder to the System Pool list. To add a folder, click **Add...**, browse to the required folder, and then click **Select Folder**. To remove a folder from the System Pool list, select it and then click **Remove**.
- 4 In the **Systems Found** list, select a system, and then click **Create**.

---

#### Creating a new RAPID module file

- 1 On the **File** tab, click **New**.
- 2 Click **RAPID Module File**.
- 3 Choose from the following options:
  - Click **Module (Program Module)** to create a blank RAPID module file.
  - Click **Main Module (Program Module)** to create a module with a main routine.
  - Click **Module (System Module)** to create a module with attributes for read-only, view-only and not step-in.

According to the selection you make, the created RAPID module file opens in the RAPID editor.

For more information on managing file based RAPID modules, see [Managing file based RAPID modules on page 423](#).

## 7.3 Share

### 7.3.1 Pack and Go

---

#### Packing a station

- 1 On the **File** menu, click **Share** and select **Pack & Go** to open the **Pack & Go Wizard**.
- 2 On the **Welcome to the Pack & Go Wizard** page, click **Next**.
- 3 On the **Destination** page, click **Browse** and specify the destination directory of the package. Click **Next**.

To password protect your Pack & Go file, select the **Password protect the package** check box. Then, specify the password. To view the password you have typed in, select the **Show password** check box. When opening the Pack & Go file, this password needs to be provided for the station to load.
- 4 On the **Libraries** page, select one of the three options. Click **Next**.
- 5 On the **Systems** page, select **Include backups of all robot systems** checkbox. Optionally, select **Include a media pool for additional options** checkbox. Click **Next**.
- 6 On the **Ready to pack** page, review the information and then click **Finish**.
- 7 On the **Pack & Go finished** page, review the results and then click **Close**.

### 7.3.2 Unpack and Work

---

#### Unpacking a station

- 1 On the **File** menu, click **Unpack & Work** to open the **Unpack & Work Wizard**.
- 2 On the **Welcome to the Unpack & Work Wizard** page, click **Next**.
- 3 On the **Select package** page, click **Browse** and **Select the Pack & Go file to unpack** and **Select the directory where the files will be unpacked**. Click **Next**.
- 4 On the **Controller Systems** page, select the **RobotWare version** and click **Browse** to select the path to the **Media Pool**. Optionally, select the check box to automatically restore backup. Click **Next**.
- 5 On the **Ready to unpack** page, review the information and then click **Finish**.
- 6 On the **Unpack & Work finished** page, review the results and then click **Close**.



#### Note

If the Pack & Go file was set as password-protected during creation, then that password needs to be provided for the station to load.

### 7.3.3 Station Viewer

#### Overview

The Station Viewer can playback a station in 3D on computers that do not have RobotStudio installed. It packages the station file together with files needed to view the station in 3D. It can also play recorded simulations.

#### Prerequisites

The .NET Framework 4.5 must be installed on the playback computer.



#### Note

RobotStudio 64-bit edition can create 64-bit Station Viewers. However, a 64-bit station viewer can run only on the Windows 64-bit operating system.

#### Creating and loading a Station Viewer

- 1 To create a Station Viewer, on the **File** menu, click **Share** and select **Save Station as Viewer**
- 2 Specify a file name and save as **.exe** file.
  - Select the option **Show comments on startup** and add text in the box to view the comment when the Station Viewer is started.
  - To save the simulation as Station Viewer, go to the **Simulation Control** group, click **Play**, and then select **Record to Viewer**. For more information, see [Running a simulation on page 340](#).



#### Note

Record to viewer is disabled when Free Run mode is enabled.

- 3 To load a Station Viewer, double-click the package (**.exe**) file on the target computer.

The results are displayed in the Output window and the embedded station file is automatically loaded and presented in a 3D view.

#### Configuring user settings of a Station Viewer

To configure the user settings of a Station Viewer, on the **File** menu, click **Options**.

#### Command Buttons

<b>Apply</b>	Click this button to save all options in the current page.
<b>Reset</b>	Click this button to reset to the settings you had before this session all values that you have changed on the current page.
<b>Default</b>	Click this button to reset to their default values all settings on the current page.

#### Options:General:Appearance

<b>Select application language</b>	Select the language to be used. The default language is the same as that of the target user's operating system if available, otherwise it is English.
------------------------------------	--

*Continues on next page*

## 7 File tab

---

### 7.3.3 Station Viewer

*Continued*

<b>Select color theme</b>	Select the color to be used.
---------------------------	------------------------------

#### **Options:General:Graphics**

<b>Background color</b>	Select the color from the color theme, or from the color stored in the stations.
-------------------------	--

---

### **Simulation**

When you run a simulation, the movements and visibility of objects are recorded. This recording is optionally included in the Station Viewer.

Simulation control buttons are enabled when the Station Viewer contains a recorded simulation.

Following are the Simulation control buttons:

<b>Play</b>	Starts or resumes simulation playback
<b>Stop</b>	Stops simulation playback
<b>Reset</b>	Resets all objects to their initial state and process time display to zero
<b>Run mode</b>	Select to run the simulation once or continuously
<b>Process time</b>	Displays the current simulation time

## 7.4 Options

### Common buttons

<b>Apply</b>	Click this button to save all options in the current page.
<b>Reset</b>	Click this button to reset to the settings you had before this session all values that you have changed on the current page.
<b>Default</b>	Click this button to reset to their default values all settings on the current page.

### Options:General:Appearance

<b>Select application language</b>	Select the language to be used.
<b>Select color theme</b>	Select the color to be used.
<b>Default scale for zoomable windows</b>	Defines the default scale to use for windows that are zoomable, for example, RAPID Editor, RAPID Data Editor and Configuration Editor.
<b>Show ScreenTips</b>	Select this check box to view ScreenTips.
<b>Display Position Edit boxes with Red/Green/Blue background</b>	Select the check box if you want to display the position boxes in the modify dialog boxes with colored background. Default value: selected.
<b>Group related document windows under one tab</b>	Select this check box to group related document window under one tab. Modifying this option requires a restart for the changes to take effect.
<b>Restore hidden dialogs and messages</b>	Select this check box to restore dialogs or messages which you may have hidden while using RobotStudio.

### Options:General:Licensing

<b>View installed licenses</b>	Click to view the licenses listed by feature, version, type, expiration date and status.
<b>Activation Wizard</b>	Click to activate RobotStudio license.
<b>RobotStudio user experience program</b> <ul style="list-style-type: none"> <li>I want to help improve RobotStudio</li> <li>I do not want to participate right now</li> </ul>	For RobotStudio Basic users, it is mandatory to participate in the user experience report. For RobotStudio Premium users, you can choose whether or not to participate in the user experience report .

### Options:General:Units

<b>Quantity</b>	Select the quantity for which you want to change the units.
<b>Unit</b>	Select the unit for the quantity.
<b>Display decimals</b>	Enter the number of decimals that you want to be displayed.
<b>Edit decimals</b>	Enter the number of decimals that you want when modifying.

### Options:General:Advanced

<b>Number of undo/redo steps</b>	The number of operations that can be undone or redone. Lowering this value can decrease memory usage.
----------------------------------	---

*Continues on next page*

## 7 File tab

### 7.4 Options

*Continued*

<b>Warn about running Virtual Controller processes on startup</b>	Warns of orphaned VC processes.
<b>Show acknowledge dialog box when deleting objects</b>	Warns when deleting objects.
<b>Show acknowledge dialog box when deleting targets and corresponding move instructions</b>	Warns when deleting targets and move instructions.
<b>Bring the output window to front if an error message is displayed</b>	Select this check box to bring the output window to front if an error message is displayed

#### Options:General:Files & Folders

<b>User Project Folder</b>	Enter the path to your project folder. This will be the folder displayed in the open and save dialog boxes in RobotStudio.
...	To browse for your project folder, click the browse button.
<b>Automatically create document subfolders</b>	Select this check box to enable the creation of individual subfolders for document types.
<b>Enable Autosave</b>	Select the check box to automatically save the station with defined intervals. Default value: cleared
<b>Interval</b>	Specify the interval between the savings when using Autosave in this box.
<b>Clear Recent Stations and Controllers</b>	Clears the list of recently accessed stations and controllers
<b>Document Locations</b>	Launches the Document Locations dialog box. For more information, see <a href="#">The Documents window on page 62</a> .

#### Options:General:Screenshot

<b>Entire application window</b>	Select this option to capture the entire application.
<b>Active document window</b>	Select this option to capture the active document window, typically the graphics window.
<b>Copy to clipboard</b>	Select this check box to save the captured image to the system clipboard.
<b>Save to file</b>	Select this check box to save the captured image to file.
<b>Location</b>	Specify the location of the image file. The default location is the "My Pictures" system folder.
...	Browse for the location.
<b>File name</b>	Specify the name of the image file. The default name is "RobotStudio" to which is added a date.
<b>The file suffix list</b>	Select the desired file format. The default format is JPG.

#### Options:General:Screen Recorder

<b>Framerate</b>	Specify the framerate in frames per second.
<b>Start recording after</b>	Select this option to start recording after the specified time.
<b>Stop recording after</b>	Select this option to stop recording after the specified time.

*Continues on next page*

<b>Resolution - Same as window</b>	Select this option to use the same resolution as in the graphics window.
<b>Resolution - Limit resolution</b>	Select this option to scale down the resolution as per the <b>Maximum Width</b> and <b>Maximum Height</b> you specify.
<b>Maximum width</b>	Specify the maximum width in pixels.
<b>Maximum height</b>	Specify the maximum height in pixels.
<b>Video compression</b>	Select the video compression format. Note that DivX format is not supported.
<b>File name and Output file format</b>	Enter a file name and a file format. The default format is WMV. You can also save the output format as MP4. The recommended format is MP4.

**Options:Robotics:RAPID Editor**

<b>Show line numbers</b>	Select this check box to view line numbers in the RAPID editor
<b>Show ruler</b>	Select this check box to show the ruler in the RAPID editor
<b>Show whitespace</b>	Select this check box to show whitespace characters in the RAPID editor
<b>Wrap long lines</b>	Select this check box if you want to wrap long lines.
<b>Convert tabs to spaces</b>	Select this check box to convert tabs to spaces in the RAPID editor
<b>Tab size</b>	Specify the number of spaces for a Tab press.
<b>Text styles</b>	Specify the appearance of the various text classes.
<b>Text color</b>	Specifies the text color of the RAPID editor.
<b>Background color</b>	Specifies the background color of the RAPID editor.
<b>Bold</b>	Select this check box for bold-face fonts in the RAPID editor.
<b>Italic</b>	Select this check box for italicized fonts in the RAPID editor.

**Options:Robotics:RAPID Profiler**

<b>Default RAPID log file</b>	Specify the name of the default RAPID log file.
<b>Always ask for filename</b>	Select this check box to manually specify the file name of the log file always.
<b>Open analysis when logging is stopped</b>	Select this check box to open the analysis after the log is made.

**Options:Robotics:Graphical Programming**

<b>Show dialog when warning for globally defined workobjects</b>	Select this check box if you want RobotStudio to display a warning when there are workobjects with the same name that have been declared as in other tasks. Default value: selected.
<b>Show synchronize dialog box after loading program/module</b>	Select this check box if you want the synchronize dialog box to be displayed when you have loaded a program or a module. Default value: selected.
<b>Show notification that default data is used</b>	Select this check box if you want to be notified that <i>wobj0</i> and/or <i>tool0</i> is active and will be used in the current action. Default value: selected.

*Continues on next page*

## 7 File tab

### 7.4 Options

*Continued*

<b>Set as active when creating tooldata</b>	Select this check box if you want newly created tooldata to be set as active. Default value: selected.
<b>Set as active when creating workobjects</b>	Select this check box if you want newly created workobjects to be set as active. Default value: selected.
<b>AutoPath</b>	Specify the maximum gap allowed when creating an AutoPath.

#### Options:Robotics:Synchronization

<b>Use default synchronization locations</b>	Converting data, such as target to Workobject, shall use the default behavior for synchronization locations. Default value: selected.
<b>Show default synchronization locations notification</b>	Notifies of the behavior above. Default value: selected.
<b>Declaration default locations</b>	Specify the locations for corresponding objects when synchronizing to the VC.

#### Options:Robotics:Mechanism

<b>Approach Vector</b>	Select the approach vector. Default value: Z.
<b>Travel Vector</b>	Select the travel vector. Default value: X.
<b>Enable configuration check for jump to target/move instruction</b>	Select this check box if you want to enable the configuration check configurations when jumping to target or move instructions. When selected and a target does not have a validated configuration assigned, you will be asked to set one. When cleared, the configuration closest to the current one is used. Default value: selected.

#### Options:Robotics:Virtual Controller

<b>Always on top</b>	Select this check box if you want to have the virtual FlexPendant always on top. Default value: selected.
<b>Enable transparency</b>	Select this check box if you want parts of the virtual FlexPendant to be transparent. Default value: selected.
<b>Logging</b>	When the controller is warm started, <ul style="list-style-type: none"><li>• Select this check box to log the console output to "console.log" in the controller directory</li><li>• Select this check box to log the console output to a console window</li></ul>
<b>Automatically open virtual Operator Window</b>	Select this check box to automatically open the virtual Operator Window. Default value: Enabled.

#### Options:Online:Authentication

<b>Recent Users</b>	Lists the recent users.
<b>Remove/Remove All</b>	Click these buttons to remove one or all recent users, respectively.
<b>Enable Automatic Logoff</b>	Select the check box if you want to log off automatically.
<b>Timeout</b>	Determines the length of the session before being automatically logged off.

#### Options:Online:Online Monitor

<b>Update Rate (s)</b>	Specifies the update interval.
------------------------	--------------------------------

*Continues on next page*

<b>Revolute Joint Limits</b>	Sets the revolution limit for joints.
<b>Linear Joint Limits</b>	Sets the linear limit for joints.
<b>Singularities</b>	Sets the singularities.

**Options:Graphics:Appearance**

<b>Anti-aliasing</b>	Move the slider to control the multisampling level used to smooth jagged edges. The available options are hardware dependent. RobotStudio must be re-started for this setting to take effect.
<b>Advanced lighting</b>	Select the check box to enable advanced lighting by default.
<b>Perspective</b>	Click this option to view the perspective view of the object by default.
<b>Orthographic</b>	Click this option to view the orthographic view of the object by default.
<b>Custom background color</b>	Click the colored rectangle to change default background color.
<b>Show floor</b>	Select the check box if you want the floor (at z=0) to be displayed by default. Change the floor color by clicking the colored rectangle. Default values: selected.
<b>Transparent</b>	Select the check box if you want the floor to be transparent by default. Default values: selected.
<b>Show UCS Grid</b>	Select the check box if you want the UCS grid to be displayed. Default value: selected.
<b>Grid Space</b>	Change the UCS grid space in the X and y coordinate directions by entering the requested value in the box. Default value: 1000 mm (or equivalent in other units).
<b>Show world coordinate system</b>	Select the check box if you want the coordinate systems to be displayed. Default value: selected.
<b>Show navigation and selection buttons</b>	Select this check box to have the navigation and selection buttons on the graphics window.

The settings you make take effect when creating a new station or when selecting **Default View Settings** from the **Settings** menu of the **View** tab of the **Graphics Tools** ribbon.

**Options:Graphics:Performance**

<b>Detail level</b>	Select if the detail level is to be Auto, Fine, Medium or Coarse. Default value: Auto.
<b>Render both sides</b>	Select the check box if you want to ignore the back-facing triangles. Default value: selected.  Culling back-facing triangles improves the graphics performance but may give unexpected display if surfaces in models are not faced correctly.
<b>Cull objects smaller than</b>	Select the size in pixels under which objects will be disregarded. Default value: 2 pixels.

The settings you make here are generic for all objects in RobotStudio. With the **Graphic Appearance** dialog box you can, however, override some of these settings for single objects.

*Continues on next page*

## 7 File tab

### 7.4 Options

*Continued*

#### Options:Graphics:Behavior

<b>Navigation</b>	Select a navigational activity and then specify the mouse buttons to be used for the selected navigational activity.
<b>Navigation sensitivity</b>	Select the navigation sensitivity when using the mouse movements or navigation buttons by clicking the bar and dragging it into position. Default value: 1.
<b>Selection radius (pixels)</b>	Change the selection radius (that is, how close the mouse cursor click must be to an item to be selected) by entering the requested pixel value in the box. Default value: 5.
<b>Selection highlight</b>	Set if the selected object shall be distinguished in the <b>Graphics</b> window by a color, by an outline or not at all. Default value: color.
<b>Highlight color</b>	Click the colored rectangle to change the highlight color.
<b>Activate selection preview</b>	Select the check box to enable temporarily highlighting of items that may be selected when the mouse cursor passes over them. Default value: selected.
<b>Show local coordinate system for selected objects</b>	Select the check box to show the local coordinate system for the selected objects. Default value: selected.

#### Options:Graphics:Geometry

<b>Detail Level</b>	Specify the level of detail required when importing geometries. Select <b>Fine</b> , <b>Medium</b> or <b>Coarse</b> as required.
---------------------	--

#### Options:Simulation:Collision

<b>Perform collision detection</b>	Select if collision detection is to be performed during simulation or always. Default value: always.
<b>Pause/stop simulation at collision</b>	Select this check box if you want the simulation to stop at a collision or at a near miss. Default value: cleared.
<b>Log collisions to Output window</b>	Select this check box if you want the collisions to be logged to the output window. Default value: selected.
<b>Log collisions to file:</b>	Select this check box if you want to log the collisions to a file. Browse for the file to log in by clicking the browse button. Default value: cleared.
<b>Enable fast collision detection</b>	Select this check box to enhance the performance by detecting collisions between geometrical bounding boxes instead of geometrical triangles. This might result in falsely reported collisions, since the triangles are the true geometry and the bounding boxes always are larger. All true collisions will, however, be reported. The larger the object, the greater the number of false collisions that are likely to be detected.
<b>View</b>	Click this button to open the log file specified in the file box in Notepad.
<b>Clear</b>	Click this button to delete the log file specified in the file box.
<b>...</b>	Click this button to browse for the file in which you want to log the collisions.

*Continues on next page*

---

**Options:Simulation:Virtual Time**

<b>Virtual Time mode- Free run</b>	This option makes RobotStudio always use the free run mode. Simulations created using the Smart Component are now supported with VC in this mode. As a result, FlexPendant and ScreenMaker applications can be executed on the FlexPendant together with Smart Component simulations.
<b>Virtual Time mode - Time Slice</b>	This option makes RobotStudio always use the time slice mode.
<b>Run time slice in parallel for multiple controllers</b>	When simulating a large number of controllers (such as ten controllers), this option may increase performance by utilizing multiple CPU cores. This option is hardware dependent and hence may give different results depending on the computer used.

---

**Options:Simulation:Accuracy**

<b>Simulation speed</b>	Sets the simulation speed relative to real time. You can define the simulation speed to a maximum of 200%
<b>As fast as possible</b>	Select this check box to run the simulation as fast as possible. When you select this option, the simulation speed slider is disabled.
<b>Simulation timestep</b>	Specifies the simulation timestep.

**This page is intentionally left blank**

## **8 Home tab**

### **8.1 Overview**

---

**The Home tab**

The Home tab contains the controls required for building stations, creating systems, programming paths and placing items.

## 8 Home tab

---

### 8.2 ABB Library

### 8.2 ABB Library

---

#### About this button

With this button, you can choose robots, positioners and tracks from their respective galleries.

## 8.3 Import Library

---

### About this button

With this button, you can import equipment, geometries, positioners, robots, tools and training objects to your station libraries.

---

### Importing a library

Use this procedure for importing library files to a station:

- 1 On the **Home** menu, click **Import Library** and select one of the following controls:
  - Equipment
  - User Library
  - Documents
  - Locations
  - Browse for Library



#### Note

You can also import component xml files (\*.rsxml) to your station.

- 2 Click **Equipment** to import predefined ABB mechanism libraries.
- 3 Click **User Library** to select the user defined libraries.
- 4 Click **Documents** to open the Documents window. See [The Documents window on page 62](#)
- 5 Click **Locations** to open the Document Locations window. See [Document Locations window on page 67](#).
- 6 Click **Browse for Library** to select the saved library files.

## 8.4 Robot System

### 8.4.1 Robot System

---

#### About this button

With the **Robot System** button, you can either create a system from layout or template, choose an existing system, or select a system from a robot gallery and setup a conveyor tracking mechanism.

---

#### Creating a system from layout

- 1 Click **From Layout** to bring up the first page of the wizard.
- 2 In the **Name** box, enter the name of the system.
- 3 In the **Location** box, enter the path to the folder where the system will be stored. Alternatively, click **Browse** and browse to the folder.
- 4 In the **Media Pool** box, enter the path to the media pool. Alternatively, click **Browse** and browse to the folder.
- 5 In the **RobotWare Version** list, select the version of RobotWare you want to use.
- 6 Click **Next**.
- 7 In the **Mechanisms** box, select the mechanisms that you want to include in the system.
- 8 Click **Next**.

The wizard now proposes a mapping of the mechanisms to a specific motion task, in accordance with the following rules:

- Only one TCP robot is allowed per task.
- Up to six motion tasks may be added, but only four TCP robots can be used, and they must be assigned to the first four tasks.
- The number of tasks may not exceed the number of mechanisms.
- If the system contains one TCP robot and one external axis, they will be assigned to the same task. It is, however, possible to add a new task and assign the external axis to it.
- If the system contains more than one TCP robot, any external axes will be assigned to a separate task. It is, however, possible to move them to other tasks.
- The number of external axes in a task is limited by the number of available drive modules in the cabinet (one for large robots, two for medium, three for small).

If only one mechanism was selected in the previous page, this page will not be shown.

Tasks can be added and removed using the respective buttons; mechanisms can be moved up or down using the respective arrows. To map the mechanisms to tasks, follow this step:

- 9 Optionally, make any edits in the mapping, and then click **Next**.

*Continues on next page*

The System Option page appears.

- 10 On the System Option page, you have the option to align Task Frame(s) with the corresponding Base Frame(s).
  - For single robot system, select the checkbox to align task frame with base frame:
  - For MultiMove Independent system, select the check box to align task frame with base frame for each robot.
  - For MultiMove Coordinated system, select the robot from the drop down list and select the check box to align task frame with base frame for the selected robot.
- 11 Verify the summary and then click **Finish**.

If the system contains more than one robot, the number of tasks and the baseframe positions of the mechanism should be verified in the System Configuration window.

---

#### Adding a template system

- 1 Click **From Template** to bring up a dialog box.
- 2 In the **Select Template System** list, either select an appropriate template or click **Browse** and browse to one.
- 3 In the **Libraries** group, select whether to import libraries or to use the existing station libraries.
- 4 In the **System** group, enter a name and location, and then click **OK**.

---

#### Adding an existing system

- 1 Click **Existing** to bring up a dialog box.
- 2 In the **Select System Pool** list, select a folder.
- 3 In the **Systems Found** list, select a system.
- 4 In the **Libraries** group, select whether to import libraries or to use the existing station libraries.
- 5 Click **OK**.

---

#### Selecting a system from a robot gallery

- 1 Click **Quick System** to bring up a gallery, and then click the appropriate robot.

---

#### Setting up a conveyor

- 1 Click **Setup**.
- 2 In the **Part Sequence** tab, select **Part** from **Available Parts**.

The right arrow button is enabled.
- 3 Click right arrow button to move the **Part** to **Parts moved by Conveyor** list.
- 4 Click up and down arrow buttons to move the selected part in **Parts moved by Conveyor** list.
- 5 In the **Part Tracking** tab, select **Part** from **Parts moved by Conveyor** list.
- 6 Select **CNV1** from the **Mechanical Unit** list.

*Continues on next page*

## 8 Home tab

---

### 8.4.1 Robot System

*Continued*

- 7 Select a workobject from the **Workobject** list.
- 8 Click **Add**. The workobject appears in the list.  
If the same workpiece is tracked by more than one robot, add a pair of workobject for each robot that tracks the workpiece. This procedure has to be repeated for each workpiece that should be tracked.
- 9 Click **OK**.
- 10 Activate the Conveyor Mechanical Unit (CNV1). See [Activate Mechanical Units on page 339](#)

---

#### Removing objects from conveyor

- 1 Click **Setup**.  
The Conveyor Setup dialog box appears.
- 2 In the **Part Sequence** tab, select **Part** from the **Parts moved by Conveyor** list  
The left arrow button is enabled.
- 3 Click left arrow to remove the part from the **Parts moved by Conveyor** list to **Available Parts** list.

## 8.4.2 External Axis Wizard

### Overview

The ABB IRC5 controller is capable of controlling a vast number of mechanical units other than the ABB robot manipulator. Some external equipment such as workpiece positioners and robot tracks are standard ABB equipment for which ABB supplies and maintains controller system configuration files. But in many situations there is a need for customized external equipment.

It is possible to use standard ABB motor units and gear units in customized external equipment. The configuration file of the motor unit or gear unit in isolation is supplied and maintained by ABB. The External Axis Wizard tool simplifies the controller configuration for different combinations of motor units and gear units in customized mechanical units.

Mechanism modeling functionality in RobotStudio makes it possible to define custom kinematic mechanisms. The External Axis Wizard lets you specify mechanisms to include in the system. First, connect and configure each axis in a mechanism to a corresponding motor unit or gear unit. Then, the template configuration files are used to assemble a complete system configuration according to specification.



#### Note

The External Axis Wizard can be downloaded from the RobotApps website. To access the RobotApps website, visit [www.abb.com/roboticssoftware](http://www.abb.com/roboticssoftware).

### Limitation

For IRC5P systems (used for painting), the External Axis Wizard supports only up to three external axes.

### Prerequisites

- Build the station, and import or model the geometry of the mechanism. For information about creating a new station, see [Workflow of building a station on page 75](#).
- Use the Mechanism Modeling functionality to define custom kinematic mechanisms. For more information, see [Create Mechanism on page 317](#).
- Add a virtual controller to the station, and include additional drives corresponding to the axes of the mechanism to the controller system.
- The robot will not be attached to the mechanism. You must manually attach the robot to the external axis after successful configuration by the External Axis Wizard.

### Using the External Axis Wizard

- 1 From the **Robot System** menu, click **External Axis Wizard**.

The first page of the wizard appears. It lists the the previously defined mechanisms (including robots) in the **Mechanisms** box.

*Continues on next page*

- 2 From the **Mechanisms** box, select the mechanisms to include in the system.
  - The **mechanism model** must be built in such a way that a kinematics model can be created. The joint chain must be defined such that it can be described by Denavit-Hartenberg parameters. The mechanism model must sometimes be modified to be able to keep the flange in the desired position. This can be done automatically by the External Axis Wizard, by adding a locked axis.
  - For adding an **additional locked axis**, when prompted, click **Ok**.

This is a dummy axis with frame definition. This dummy axis is added to the controller configuration and also to the RobotStudio mechanism. You cannot jog this additional axis.
- 3 Click **Next**.

The mechanisms along with their joints are listed.
- 4 Configure the **Mechanical Unit** based on the following information. The name of the mechanism in RobotStudio corresponds to the mechanical unit of the motion configuration.
  - Select the drive module (DM1 - DM4) to use for the mechanical unit.

If the mechanism has more than three joints, or, if there are several external mechanisms which are to be part of the configuration, then you need to use additional drive modules. In such cases, before using the External Axis Wizard, configure the controller system with the appropriate number of drive modules.
  - Optionally, you can use an **Activation Relay** by selecting the corresponding check box.

For more information, see *Technical reference manual - System parameters (3HAC17076-1)*.
  - For two axis positioners with rotating axes, where the two axes are in series, you can optionally select the **Error Model** check box.

Mechanisms defined with the error model can be calibrated with the FlexPendant using the standard four point method for each axis. This will compensate for deviations present in the real mechanisms.
  - You can configure two mechanical units to share a drive module.

To share the drive module with a mechanical unit, select that mechanical unit in the **Common Drive** list. The list shows all mechanisms (except TCP robot) with the same number of joints as the selected one. The joints of the mechanical units use the same logical axis and drive module. Note that the common drive option is not available unless an activation relay is chosen for the mechanical unit. This ensures that two mechanical units sharing a drive cannot be activated at the same time.
- 5 Configure the **joint** based on the following information.
  - For each joint select the **Motor Unit**. You can select a motor unit (MU), or a gear unit (MTD), or an interchange unit (MID). The list is populated with the standard motor and gear units provided by ABB.

Your choice will affect the capacity and cycle time of the external axis.

- **Electronically linked motors** are two motor units that drive the same axis. To link a motor unit to another one electronically, select the respective joint in the **Follow** list.
- The **Drive Unit** list is populated with the available drive units of the system. Each joint will be represented by its selected drive unit.
- You can configure **Logical Axis, Transmission, Link, Board** and **Node** according to your needs.

For more information, see *Technical reference manual - System parameters (3HAC17076-1)*.

**Note**

Default values are given for all attributes, except Motor Unit. However, you must review and change to the correct parameters to give a valid configuration.

**6 Click Next.**

The Finish page comes up.

**7 To save the configuration to a file, click Save.**

The configured kinematics of external axis devices are saved to a configuration file.

**8 To load the saved configuration to the system on exiting from the wizard, select the Load Configuration to System check box.****9 Click Finish to exit the wizard.**

Use the saved configuration files to assemble a complete controller system configuration according to specification. When a system is configured, the MOC.cfg file with a subset for the external equipment is saved and a virtual controller is started for verification.

**Note**

All mechanisms which have been used in this configuration will be disconnected from the library. To maintain these changes in a library file (.rslib), you need to manually save it. This is because the External Axis Wizard may have automatically adjusted the mechanisms to enable them to be configured.

## 8.5 Import Geometry

---

### Importing geometry

- 1 On the **Home** menu, click **Import Geometry** and select one of the following controls:
  - **User Geometry**
  - **Browse for Geometry**

- 2 Click **User Geometry** to select the user defined geometry.

- 3 Click **Browse for Geometry** to browse to the folder where the geometry is located.

For predefined geometries, click the **Geometry** icon to the left in the dialog box.

- 4 Select the required geometry and click **Open**.

If you want the geometry to move with another object, attach it to the requested object, see [Attach to on page 450](#).

To modify the detail level for import of geometries, see [Options on page 195](#).



#### Note

The **Modeling** tab also contains the **Import Geometry** option.

## 8.6 Frame

### 8.6.1 Frame

#### Creating a frame

- 1 Click **Frame**.
- 2 In the dialog box, specify the positions for the frame.

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Frame Position</b>	Click in one of these boxes, and then click the frame position in the graphics window to transfer the values to the <b>Frame Position</b> boxes.
<b>Frame Orientation</b>	Specify the coordinates for the frame orientation.
<b>Set as UCS</b>	Select this check box to set the created frame as the user coordinate system.



#### Note

The **Modeling** tab also contains the **Frame** option.

### 8.6.2 Frame from Three Points

#### Creating a frame from three points

- 1 Click **Frame from Three points** to bring up a dialog box.
- 2 Decide how you want to specify the frame:

To specify the frame using	Select
X, Y and Z coordinates, a point on the X axis and a point in the X-Y plane	<b>Position</b>
two points on the X axis and one point on the Y axis	<b>Three Point</b>

- 3 If you select **Position**:
  - Enter the **Position** for the object.
  - Enter the **Point on X axis** for the object.
  - Enter the **Point on X-Y plane** for the object.
  - Click **Create**.
- 4 If you select **Three Point**:
  - Enter the **First Point on X axis** for the object. This is the point closest to the frame's origin.
  - Enter the **Second Point on X axis** for the object. This is the point further away in the positive X direction.
  - Enter the **Point on Y axis** for the object.
  - Click **Create**.

#### The Create Frame From Three Points dialog box

<b>Position</b>	Select this option if you want to create the frame by using a position and two points.
<b>Frame Position</b>	Click in one of these boxes, and then click the frame position in the graphics window to transfer the values to the <b>Frame Position</b> boxes.
<b>Point on X axis</b>	Click in one of these boxes, and then click the point position in the graphics window to transfer the values to the <b>Point on X axis</b> boxes.
<b>Point on X-Y plane</b>	Click in one of these boxes, and then click the point position in the graphics window to transfer the values to the <b>Point on X-Y plane</b> boxes.
<b>Three Point</b>	Select this option if you want to create the frame by using three points.
<b>First Point on X axis</b>	Click in one of these boxes, and then click the point position in the graphics window to transfer the values to the <b>First Point on X axis</b> boxes.
<b>Second Point on X axis</b>	Click in one of these boxes, and then click the point position in the graphics window to transfer the values to the <b>Second Point on X axis</b> boxes.
<b>Point on Y axis</b>	Click in one of these boxes, and then click the point position in the graphics window to transfer the values to the <b>Point on Y axis</b> boxes.

*Continues on next page*

<b>Set as UCS</b>	Select this check box to set the created frame as the user co-ordinate system.
-------------------	--

## 8.7 Workobject

### Creating a workobject

- 1 On the **Home** tab, in the **Path Programming** group, click **Other** and select **Create Workobject**.  
The **Create Workobject** dialog box appears.
- 2 In the **Misc Data** group, enter the values for the new workobject.
- 3 In the **User Frame** group, do one of the following:
  - Set the position of the user frame by entering values for the **Position x,y,z** and the **Rotation rx, ry, rz** for the workobject by clicking in the **Values** box.
  - Select the user frame by using the **Frame by points** dialog box.
- 4 In the **Object Frame** group you can reposition the object frame relative to the user frame by doing any of the following:
  - Set the position of the object frame by selecting values for **Position x, y, z** by clicking in the **Values** box.
  - For the **Rotation rx, ry, rz**, select **RPY (Euler XYZ)** or **Quaternion**, and enter the rotation values in the **Values** dialog box.
  - Select the object frame by using the **Frame by points** dialog box.
- 5 In the **Sync Properties** group, enter the values for the new workobject.
- 6 Click **Create**. The workobject will be created and displayed under the **Targets** node under the robot node in the **Paths&Targets** browser.

### The Create Workobject dialog box

<b>Name</b>	Specify the name of the workobject.
<b>Robot holds workobject</b>	Select whether the workobject is to be held by the robot. If you select <b>True</b> , the robot will hold the workobject. The tool can then either be stationary or held by another robot.
<b>Moved by mechanical unit</b>	Select the mechanical unit that moves the workobject. This option is applicable only if <b>Programmed</b> is set to <b>False</b> .
<b>Programmed</b>	Select <b>True</b> if the workobject is to use a fixed coordinate system, and <b>False</b> if a movable (that is, external axes) will be used.
<b>Position x, y, z</b>	Click in one of these boxes, and then click the position in the graphics window to transfer the values to the <b>Position</b> boxes.
<b>Rotation rx, ry, rz</b>	Specify the rotation of the workobject in the UCS.
<b>Frame by points</b>	Specify the frame position of the user frame.
<b>Position x, y, z</b>	Click in one of these boxes, and then click the position in the graphics window to transfer the values to the <b>Position</b> boxes.
<b>Rotation rx, ry, rz</b>	Specify the rotation of the workobject.
<b>Frame by points</b>	Specify the frame position of the object frame.
<b>Storage type</b>	Select <b>PERS</b> or <b>TASK PERS</b> . Select the <b>Storage Type</b> <b>TASK PERS</b> if you intend to use the workobject in multimove mode.
<b>Module</b>	Select the module in which to declare the workobject.

## 8.8 Tooldata

---

### Creating tooldata

- 1 In the **Layout** browser, make sure the robot in which to create the tooldata is set as the active task.
- 2 On the **Home** tab, in the **Path Programming** group, click **Other**, and then click **Tooldata**.  
This opens the *Create Tooldata* dialog box.
- 3 In the **Misc Data** group:
  - Enter the **Name** of the tool.
  - Select whether the tool is to be held by the robot in the **Robot holds tool** list.
- 4 In the **Tool Frame** group:
  - Define the **Position x, y, z** of the tool.
  - Enter the **Rotation rx, ry, rz** of the tool.
- 5 In the **Load Data** group:
  - Enter the **Weight** of the tool.
  - Enter the **Center of gravity** of the tool.
  - Enter the **Inertia** of the tool.
- 6 In the **Sync Properties** group:
  - In the **Storage type** list, select **PERS** or **TASK PERS**. Select **TASK PERS** if you intend to use the tooldata in MultiMove mode.
  - In the **Module** list, select the module in which to declare the tooldata.
- 7 Click **Create**. The tooldata appears as a coordinate system in the graphics window.

## 8.9 Target

### 8.9.1 Teach Target

---

#### Teaching a target

To teach a target, follow these steps:

- 1 In the **Layout** browser, select the workobject and tool for which you want to teach the target.
- 2 Jog the robot to the preferred position. To jog a robot linearly, its VC must be running.
- 3 Click **Teach Target**.
- 4 A new target will be created in the browser, under the active workobject node. In the graphics window a coordinate system will be created at the TCP position. The configuration of the robot at the target will be saved.

## 8.9.2 Create Target

### Creating a target

- 1 In the **Layout** browser, select the workobject in which you want to create the target.
- 2 Click **Create Target** to bring up a dialog box.
- 3 Select the **Reference** coordinate system you want to use to position the target:

If you want to position the target	Select
absolute in the world coordinate system of the station	<b>World</b>
relative to the position of the active workobject	<b>Work Object</b>
in a user-defined coordinate system	<b>UCS</b>

- 4 In the **Points** box, click **Add New** and then click the desired position in the graphics window to set the position of the target. You can also enter the values in the **Coordinates** boxes and click **Add**.
- 5 Enter the **Orientation** for the target. A preliminary cross will be shown in the graphics window at the selected position. Adjust the position, if necessary. To create the target, click **Create**.
- 6 If you want to change the workobject for which the target is to be created, expand the **Create Target** dialog box by clicking the **More** button. In the **WorkObject** list, select the workobject in which you want to create the target.
- 7 If you want to change the target name from the default name, expand the **Create Target** dialog box by clicking the **More** button and entering the new name in the **Target name** box.
- 8 Click **Create**. The target will appear in the browser and in the graphics window.



#### Note

The created target will not get any configuration for the robot axes. To add the configuration values to the target, use either **ModPos** or the **Configurations** dialog box.

If using external axes, the position of all activated external axes will be stored in the target.

### The Create Target dialog box

<b>Reference</b>	Select the reference coordinate system to which all positions or points will be related.
<b>Position</b>	Click in one of these boxes, and then click the position in the graphics window to transfer the values to the <b>Position</b> boxes.
<b>Orientation</b>	Specify the orientation of the target.

*Continues on next page*

## 8 Home tab

---

### 8.9.2 Create Target

*Continued*

<b>Add</b>	Click this button to add a point and its coordinates to the <b>Points</b> list.
<b>Modify</b>	Click this button to modify an already defined point, after you have selected it in the <b>Points</b> list and entered new values.
<b>Points</b>	The target points. To add more points, click <b>Add New</b> , click the desired point in the graphics window, and then click <b>Add</b> .
<b>More/Less</b>	Click this button to expand or collapse parts of the create target dialog box.
<b>Target name</b>	Here you can change the name of the target you are creating. It is visible only when the create target dialog box is expanded.
<b>Workobject</b>	Here you can change the workobject in which the target is to be created. It is visible only when the create target dialog box is expanded.

### 8.9.3 Create Jointtarget

---

#### Creating a jointtarget

- 1 Click **Create Jointtarget** to bring up a dialog box.
- 2 If you want to change the default name of the jointtarget, enter the new name in the **Name** box.
- 3 In the **Axes Values** group, do as follows:
  - For the **Robot axes**, click the **Values** box and then click the down arrow. The **Joint Values** dialog box will be displayed. Enter the joint values in the boxes and click **Accept**.
  - For the **Joint axes**, click the **Values** box and then click the down arrow. The **Joint Values** dialog box will be displayed. Enter the joint values in the boxes and click **Accept**.
- 4 Click **Create**. The jointtarget will appear in the browser and in the graphics window.

---

#### The Create Jointtarget dialog box

<b>Name</b>	Specify the name of the jointtarget.
<b>Robot axes</b>	Click the <b>Values</b> list, enter the values in the <b>Joint values</b> dialog box and click <b>Accept</b> .
<b>External axes</b>	Click the <b>Values</b> list, enter the values in the <b>Joint values</b> dialog box and click <b>Accept</b> .
<b>Storage Type</b>	Select the <b>Storage Type</b> <b>TASK PERS</b> if you intend to use the jointtarget in multimove mode.
<b>Module</b>	Select the module in which you want to declare the jointtarget.

#### 8.9.4 Create Targets on Edge

##### Overview

Targets on Edge creates targets and move instructions along the edges of the geometric surface by selecting target points in the graphics window. Each point on a geometric edge has certain properties that can be used to position robot targets relative to the edge.

##### Creating targets on edge

- 1 On the **Home** tab, click **Target** and select **Create Targets on Edge**.

The **Targets on Edge** dialog box appears.



##### Note

The selection mode in graphics window is automatically set to **Surface**, and the snap mode is set to **Edge**.

- 2 Click on the surface of the body or part to create target points.

The closest point on the adjacent edge is calculated and added to the list box on as target points Point 1, Point 2 ....



##### Note

When an edge is shared between two surfaces, the normal and tangent directions depend on the surface selected.

- 3 Use the following variables to specify how a target is related to a point on the edge.

Select...	to...
Vertical offset	specify the distance from the edge to the target in the surface normal direction.
Lateral offset	specify the distance from the edge to the target perpendicular to the edge tangent.
Approach angle	specify the angle between the (inverse) surface normal and the approach vector of the target.
Reverse travel direction	specify if the travel vector of the target is parallel or inversely parallel to the edge tangent.



##### Note

For each target point, a preview of the approach and travel vectors are displayed as arrows and as a sphere representing the point on the edge in the graphics window. The preview of the arrows are updated dynamically once the variables are modified.

- 4 Click **Remove** to remove the target points from the list box.

*Continues on next page*

- 5 Click **More** to expand the **Create Targets on Edge** dialog box and to choose the following advanced options:

Use...	to..
<b>Target name</b>	change the target name from the default name to a new user defined name
<b>Task</b>	select the task for which to add targets. By default, active task in the station is selected.
<b>Workobject</b>	select the workobject for which you want to create the targets on edge
<b>Insert Move Instructions in</b>	create Move instructions in addition to targets, which will be added to the selected path procedure. The active process definition and process template will be used.

- 6 Click **Create**.

The target points and Move instructions (if any) are created and are displayed in the Output window and graphics window.

### 8.10 Empty Path

---

#### Creating an empty path

- 1 In the **Paths&Targets** browser, select the folder in which you want to create the path.
- 2 Click **Empty Path**.
- 3 To set the correct motion properties for the targets, select the active process in the **Change Active Process** box in the **Elements** toolbar.
- 4 If the active template is set to **MoveAbsJoint**, then:
  - A target that is dragged into a path will be converted into a **jointtarget** (recognized by a different icon on in the browser).
  - Jointtargets and their instructions can only use *wobj0* and *tool0*.
  - One target can not be used as different types, for example, **MoveJoint**, but must be deleted and re-created.
  - When the target has been synchronized with the virtual controller, the **jointtarget** values will be calculated and inserted in the RAPID program.

## 8.11 AutoPath

### Overview

AutoPath helps in generating accurate paths (linear and circular) based on CAD geometry.

### Prerequisites

You need to have a geometric object with edges, curves, or both.

### Creating a path automatically

AutoPath feature can create paths from curves or along the edges of a surface. To create a path along a surface use selection level Surface, and to create a path along a curve, use selection level Curve. When using Selection level Surface, the closest edge of the selection will be picked for inclusion in the path. An edge can only be selected if connected to the last selected edge.

When using Selection level Curve, the selected edge will be added to the list. If the curve does not have any branches, all edges of the entire curve will be added to the list if holding the SHIFT button when selecting an edge. The Approach and Travel directions as defined in the RobotStudio options are used to define the orientation of the created targets, see [Options:Robotics:Mechanism on page 198](#).

Use this procedure to automatically generate a path.

- 1 In the **Home** tab, click **Path** and select **AutoPath**.

The AutoPath tool appears.

- 2 Select the edge or curve of the geometric object for which you want to create a path.

The selection is listed as edges in the tool window.



#### Note

- If in a geometric object, you select curve (instead of an edge), then all the points that result in the selected curve are added as edges to the list in the graphic window.
- Ensure you always select continuous edges.

- 3 Click **Remove** to delete the recently added edge from the graphic window.



#### Note

To change the order of the selected edges, select **Reverse** check box.

- 4 You can set the following **Approximation Parameters**:

Select or enter values in	to
MinDist	Set the minimum distance between the generated points. That is, points closer than the minimum distance are filtered.
Tolerance	Set the maximum deviation from the geometric description allowed for the generated points.

*Continues on next page*

## 8 Home tab

### 8.11 AutoPath

*Continued*

Select or enter values in	to
MaxRadius	Determines how large a circle radius has to be before considering the circumference as a line. That is, a line can be considered as a circle with infinite radius.
Linear	Generate a linear move instruction for each target.
Circular	Generate circular move instructions where the selected edges describe circular segments.
Constant	Generate points with a Constant distance
End Offset	Set the specified offset away from the last target.
Start Offset	Sets the specified offset away from the first target.

The **Reference Surface** box shows the side of the object that is taken as normal for creating the path.

Click **More** to set the following parameters:

Select or enter values in	to
Approach	Generate a new target at a specified distance from the first target.
Depart	Generates a new target at a specified distance from the last target.

- 5 Click **Create** to automatically generate a new path.

A new path is created and move instructions are inserted for the generated targets as set in the Approximation parameters.



#### Note

The targets are created in the active workobject.

- 6 Click **Close**.

## 8.12 MultiMove

### Overview

For browsing between the pages of the MultiMove window, click the tabs in the navigation pane. By default the tabs are arranged in an order that corresponds to the typical workflow:

### Setup tab

<i>System Config</i>	<b>Select System</b>	Here you select the system that contains the robots to program.
	<b>System</b>	Each robot in the system is presented in its own row in this grid. In the columns you make the settings as described below.
	<b>Enable</b>	Select this check box to use the robot in the MultiMove program.
	<b>Type</b>	Specify if the robot holds the tool or the work piece.
	<b>Robot</b>	Displays the name of the robot.
<i>Path Config</i>	<b>Update</b>	Click this button to update the paths in the grid if any of the paths have been changed. The button turns red if a change has been detected and an update is necessary.
	<b>Paths</b>	Each path in the station is presented in its own row in this grid. In the columns you make the settings as described below.
	<b>Enable</b>	Select this check box for the paths to use for the program.
	<b>Order</b>	Displays the order in which the paths will be executed. To change the order, use the lists in the path column for rearranging the rows in which the paths appear.
	<b>Path</b>	Sets the path to be executed here.
<i>Start Position</i>	<b>Select Robot that other shall jump to</b>	When creating a new start position, select a robot that the other will try to reach here.
	<b>Apply</b>	Jumps the other robots to the new start position.

### Motion Behavior tab

This is used for specifying constraints and rules for how the robots shall move relative to each other. The default setting is no particular constraints, which results in the fewest joint movements. However, changing the motion behavior might be useful for:

- Locking the orientation or position of the tool.
- Optimizing cycle time or reachability by allowing tolerances.

*Continues on next page*

- Avoiding collisions or singularity by restricting joint motions.

Both Joint Influence and TCP Constraints restrict the robot's motions. Changes in these settings might result in lower performance or situations where it is impossible to find proper solutions. The weight values for Joint Weights and TCP Constraints set how much the setting for each joint or TCP direction shall affect the robots relative to each other. It is the difference between the weight values that matters, not the absolute values. If contradicting behaviors have been set, the one with the lowest weight value will win.

Tool Tolerance, instead of restricting, enables more motions. Therefore, tolerances may improve cycle and process times and enhance the reachability of the robots. Tolerances, too, have weight value; here is set how much the robots shall use the tolerance. A low value indicates that the tolerance will be used a lot, while a high value means that the robots will try to avoid using the tolerance.

The joint influence controls the balance of how much the robots will use their joints. Decreasing the weight value for one axis will restrict the motion for this axis, while increasing it will promote motion on this axis relative to alternative axes.

The TCP constraints control the position and orientation of the tool. Enabling a TCP constraint will decrease the motion of the tool and increase the motion of the work piece.

The tool tolerances control the allowed deviation between the tool and the work piece. By default, tolerances are not enabled, which means that no deviation is allowed. Enabling a tolerance, if applicable, might improve motion performance. For example, if the tool is symmetric around its Z axis, you can enable the Rz tolerance without affecting the accuracy of the generated paths.

The tool offset sets a fixed distance between the tool and the paths.

<b>Joint Influence</b>	<b>Select Robot</b>	Select the robot's joints to constrain in this box.
	<b>Joints for Robot</b>	Displays the robot's joints and their constraint weights. Each joint is presented in its own row.
	<b>Axis</b>	Displays which axis the constraint affects.
	<b>Influence</b>	Specify how much the motion for the axis is constrained. 0 means a locked axis, while 100 means no constraint relative to default constraint values.
<b>TCP Constraints</b>	<b>Active TCP</b>	This grid displays the position and rotations of the TCP together with their constraint weights.
	<b>Enable</b>	Select this check box to activate the constraint for this TCP pose.
	<b>Pose</b>	Displays the TCP pose that is affected by the constraint.
	<b>Value</b>	Specify the pose value to constrain at. Either type the value, or click the Pick from TCP button to use the values of the current TCP position.

*Continues on next page*

	<b>Influence</b>	Specify how much the motion for the TCP value is constrained. 0 means a locked TCP at this pose, while 100 means no constraint relative to default constraint values.
<i>Tool Tolerance</i>	<b>Enable</b>	Select this check box to activate the tolerance for this tool pose.
	<b>Pose</b>	Displays the tool pose that is affected by the constraint.
	<b>Value</b>	Specify the pose value to apply the tolerance around.
	<b>Influence</b>	Specify the size of the tolerance. 0 means no deviation is allowed, while 100 means all deviations are allowed.
<i>Tool Offset</i>	<b>Enable</b>	Select this check box to activate the offset for this tool pose.
	<b>Pose</b>	Displays the tool pose that is affected by the offset setting.
	<b>Offset</b>	Specify the value of the offset here.

### Create Paths tab

This tab is used for creating RobotStudio paths for the MultiMove robots. The created paths will accord with the motions displayed during the most recently played test simulation.

With the settings group you set up the MultiMove properties that connect the tasks for the tool robot and work piece robot to each other.

With the WP robot settings group you set up the properties for the task that will be generated for the work piece robot.

The generate path group contains the button that creates the paths.

<i>Settings</i>	<b>Start ID</b>	Specify the first ID number for the synchronization of the instructions for the robots.
	<b>ID step index</b>	Specify the gap between the succeeding ID numbers.
	<b>Sync ident prefix</b>	Specify a prefix for the syncident variable, which connects the sync instructions in the tasks for the tool robot and the work piece robot with each other.
	<b>Task list prefix</b>	Specify a prefix for the tasklist variable, which identifies the tasks for the tool robot and work piece robot to synchronize.
	<b>Path Prefix</b>	Specify a prefix for the generated paths.
	<b>Target Prefix</b>	Specify a prefix for the generated targets.
<i>WP robot settings</i>	<b>WP Workobject</b>	Specify the work object to which the targets generated for the work piece robot shall belong.

*Continues on next page*

## 8 Home tab

### 8.12 MultiMove

*Continued*

	<b>WP TCP</b>	Specify which tool data the work piece robot shall use when reaching its targets.
<i>Generate Paths</i>	<b>Create paths</b>	Clicking this button generates paths in RobotStudio for the latest tested motions according to the settings specified.

### MultiTeach tab

With this tab you teach complete synchronized move instructions for the robots in the MultiMove program.

<i>Settings</i>	<b>Path Prefix</b>	Specify a prefix for the paths to create.
	<b>Target Prefix</b>	Specify a prefix for the generated targets here.
	<b>Start ID</b>	Specify the first ID number for the synchronization of the instructions for the robots.
	<b>ID step index</b>	Specify the gap between the succeeding ID numbers.
	<b>Sync ident prefix</b>	Specify a prefix for the syncident variable, which connects the sync instructions in the tasks for the tool robot and the work piece robot with each other.
	<b>Task list prefix</b>	Specify a prefix for the tasklist variable, which identifies the tasks for the tool robot and work piece robot to synchronize.
	<b>Select type of Sync instruction</b>	Select the type of synchronization to use. <b>Coordinated</b> implies that all move instructions are synchronized for the robots. <b>Semicoordinated</b> implies that the robots work independently at some times and wait for each other at other (like when repositioning the work piece). For detailed information about the coordination types, see the <i>Application manual - Multimove</i> .
	<b>Setup</b>	Select the robots for teaching targets. This grid also displays the workobjects and tools that will be used for the targets.
<i>Teach</i>	<b>MultiTeach Information</b>	Displays a hierarchal tree which contains the created move instructions. The tree is organized the same way as the tree in the Layout browser.
	<b>MultiTeach</b>	Creates move instructions for the robots selected in the settings to their current positions. The created move instructions are immediately inserted at their correct places in the MultiTeach Information tree.

*Continues on next page*

	<b>Done</b>	Confirms the creation of the instructions.
--	-------------	--

**Test tab**

RobotStudio's MultiMove window has a page with commands for testing multimove programs. Its default placement is at the bottom of the MultiMove window.

The status group displays the status of the simulation, that is, whether the current settings have been tested or if errors have occurred.

In addition to the status group, information from the virtual controller is also displayed in RobotStudio's Output window during simulation.

<i>Play</i>	<< < > >>	Jumps the robots, respectively, to the previous and next targets in paths. The double arrow buttons jump several targets at once, while the single arrow buttons jump one target for each click.
	<b>Play</b>	Click this button to move the robots along the paths. Play also has a list box in which you can activate the following commands: <ul style="list-style-type: none"> <li>• <b>Save current position:</b> Saves the current start position. Since the calculated motions are based on the robot start position saving the start position is useful when testing alternative solutions.</li> <li>• <b>Restore saved position:</b> Moves the robots back to the saved start positions.</li> <li>• <b>Restore last closed loop position:</b> Moves the robots back to the list used start position.</li> <li>• <b>Close loop:</b> Finds a suitable start position based on the robots' current positions and prepares the calculation of movements.</li> <li>• <b>Calculate:</b> Calculates and executes the movements.</li> </ul>
	<b>Simulation speed</b>	Sets the speed at which the simulation is performed.
<b>Settings</b>	<b>Stop at end</b>	Select this check box to make the simulation stop after running the paths one time. If cleared, the simulation will continue playing over and over until it is stopped manually.
	<b>Simulate Online</b>	Select this check box to run the simulation as the movements are calculated.  This is useful for troubleshooting purposes, since it displays and reports targets the robot cannot reach.

*Continues on next page*

## 8 Home tab

### 8.12 MultiMove

*Continued*

	<b>Cancel on error</b>	Select this check box to stop the simulation if an error occurs. Cancel on errors is recommended to use when simulate online is used to minimize the number of error messages once the first error is identified.
	<b>Watch Close Loop</b>	Select this check box to display the search for a suitable start position in the graphics window. Clear the check box to jump the robots to the start position when it is found.

#### The MultiMove configure system wizard

The MultiMove configure system wizard guides you through configuring robots and workobjects for MultiMove system. If the workobjects are not configured correctly when the MultiMove functions are started, you will be asked whether to run the wizard. You can also start it manually from the Tools page of MultiMove.

The wizard contains four pages, the information pane at the bottom indicates the current page.

Workpiece robot	The workpiece robot page contains a list in which you select the robot that holds the workpiece. Only one robot can be set up as workpiece robot. If your station has several robots that hold the workpiece, set up one of them as workpiece robot and the other as tools robots, and create paths for these robots in which they only hold the workpiece.
Tool robots	The tool robots page contains a list in which you select the robots that operate on the work piece. All robots selected as tools robots will be coordinated to the workpiece robot. Any robot of the system that is selected neither as workpiece robot or tool robot will not be coordinated.
Workobjects	The Workobjects page contains a box for each tool robot in which you specify the workobject in which the targets for the MultiMove paths shall be created. The wizard will attach this workobject to the workpiece robot, for enabling MutliMove. Either type in a name for a new Workobject to create in the box, or select the <b>Use existing WorkObject</b> check box and then select one from the list.
Result	The Result page displays a summary of the station configuration. Either click <b>Done</b> to finish or <b>Previous</b> to go back and change the setup.



#### Note




The wizard will not help you configure the RobotWare system correctly. If the correct options are not used you will not be able to synchronize generated MultiMove paths to the virtual controller, even if you can use the MultiMove functions in RobotStudio.

*Continues on next page*

**The Analyze path tool**

This tool checks whether existing paths are coordinated correctly for MultiMove.

The analyzer opens in a window of its own and contains three pages. The information pane at the bottom indicates the current page.

<i>Select Paths</i>	<b>Enable</b>	Select to include the task in the analysis.
	<b>Task</b>	Displays the name of the task.
	<b>Path</b>	Select the path to analyze for the current task.
<i>Analyze</i>	<b>Analyze</b>	Click this button to start the analysis.
<i>Report</i>	 reportok	OK. The paths are compatible in the specified aspect.
	 reportin	Information. The paths are not fully compatible in the specified aspect, but the robot program may still be executable.
	 reporter	Error. The paths are not compatible in the specified aspect, and the robot program is not executable.

**The Recalculate ID tool**

This is one of the tools for working manually with MultiMove programs. It sets new sync ID arguments to the move instructions in a MultiMove path. By using the tool with the same start ID and ID step index on all paths that shall be synchronized, you can be sure that the IDs match if all paths contain the same number of move instructions.

<b>Title</b>	Displays the name of the path to recalculate IDs for.
<b>Start ID</b>	Sets the number of the first ID in the path.
<b>ID Step index</b>	Sets the size of the step to increase the ID number for each move instruction.
<b>Only update instructions that has ID defined</b>	Select this check box to recalculate only those IDs for instructions that have existing IDs. Clear this check box to also create IDs for instructions that have no IDs (for example, if you have added new move instructions that shall be coordinated).
<b>The Only update instructions between SyncMove On/Off check box.</b>	Select this check box to affect only those move instructions that belong to already synchronized parts of the path. Clear this check box to update IDs for instructions in all parts of the path.

*Continues on next page*

## 8 Home tab

---

### 8.12 MultiMove

*Continued*

---

#### The Convert path to MultiMove path tool

This is one of the tools for working manually with MultiMove programs. It adds sync ID arguments to all move instructions in the path and, optionally, `SyncMoveOn/Off` instructions, thus preparing an ordinary path for MultiMove use.

You use the tool on one path a time, so for creating a MultiMove program, you convert one path for each robot and then create a tasklist and syncidents which you add to the Sync instructions.

<b>Title</b>	Displays the name of the path to recalculate IDs for.
<b>Start ID</b>	Sets the number of the first ID in the path.
<b>ID Step index</b>	Sets the size of the step to increase the ID number for each move instruction.
<b>Add SyncMove On/Off before and after</b>	Select this check box to add instructions that start and stop the synchronization.

---

#### The Create Tasklist tool

This is one of the tools for working manually with MultiMove programs. It creates a variable of the RAPID data type `tasks`, which identifies the tasks that will be synchronized. In each `SyncMoveOn` or `WaitSyncTask` instruction, you then specify which tasklist to use.

<b>Tasklist name</b>	Specifies the name of the tasklist.
<b>Tasks that will be included</b>	Select the check box for each task to include in the list.

---

#### The Create Syncident tool

This is one of the tools for working manually with MultiMove programs. It creates a variable of the RAPID data type `SyncIdent`, which identifies the sync instructions that shall be synchronized.

<b>Syncident name</b>	Specifies the name of the <code>SyncIdent</code> variable to create.
<b>Tasks that Syncident will be created in</b>	Select the check box for each task to use this Syncident in.

## 8.13 Teach Instruction

---

### Teaching a move instruction

- 1 In the **Layout** browser, make sure the settings for active robot, workobject, tool, motion type and path are appropriate for the move instruction to create.
- 2 Jog the robot to the desired location. If jogging the robot using the freehand mode, you can also use snap modes for snapping its TCP to objects in the station.
- 3 Click **Teach Instruction**. A move instruction is now created last in the path.

## 8.14 Move Instruction

---

### Creating a move instruction and a corresponding target

To create a move instruction, follow these steps:

- 1 Click **Move Instruction** to bring up a dialog box.
- 2 Select the **Reference coordinate system** for the move instruction.
- 3 Enter the **Position** to reach for the move instruction by clicking **Add New** in the **Coordinates** box and then click the required to-points in the graphics window. You can also enter the values in the **Coordinates boxes** and click **Add**.
- 4 Enter the **Orientation** for the move instruction.
- 5 By clicking the **More/Less** button, you can expand or collapse the **Create Move Instruction** dialog box. When the dialog box has been expanded, you can change the **Target name** and the **Work object** to which the target (with the move instruction) will belong.
- 6 Click **Create** to create the move instruction. The move instruction will appear under the path node as a reference to the target.

### The Create Move Instruction dialog box for jointtarget movements

<b>Name</b>	Here you can change the name of the target you create when creating the move instruction.
<b>Robot axes</b>	Specify the joint values for the robot. Select the box and click the list to set the values.
<b>External axes box</b>	Specify the joint values for external axes, if any exist in the station. Select the box and click the list to set the values.
<b>Storage Type</b>	Click this button to expand or collapse parts of the create move instruction dialog box.
<b>Module</b>	Specify the module in which the jointtarget shall be declared.

## 8.15 Action Instruction

### Creating an action instruction

- 1 In the **Paths&Targets** browser, select where to insert the action instruction.

To insert the action instruction	Select
at the beginning of a path	the path
after another instruction	the proceeding instruction

- 2 Right click **Path** and select **Insert Action Instruction**.  
The **Create Action Instruction** dialog box appears.
- 3 From the **Instruction Templates** list, select the action instruction to create.
- 4 Optionally, modify instruction arguments in the **Instruction Arguments** grid.  
For detailed information about the arguments for each instruction, see the [Action Instruction on page 237](#).
- 5 Click **Create**.

## 8 Home tab

### 8.16 Instruction Template Manager

## 8.16 Instruction Template Manager

### Overview

The Instruction Template Manager is used to add support for instructions other than the default set that comes with the RobotStudio.

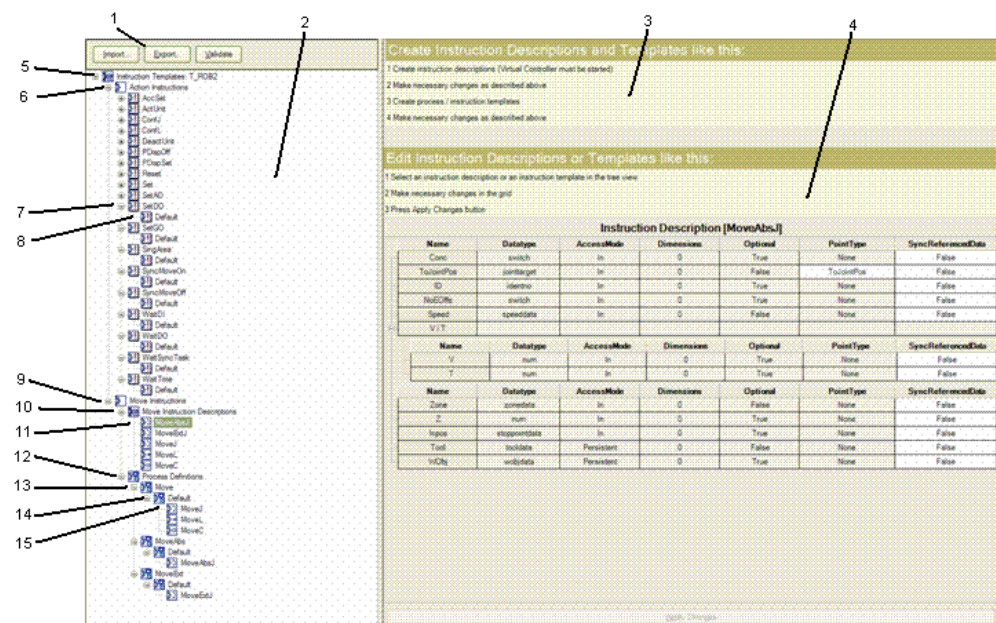
For example, a robot controller system with the RobotWare Dispense option has specialized move instructions related to glueing like DispL and DispC. You can manually define the instruction templates for these using the Instruction Template Manager. The instruction templates are exported to XML format and reused later.

RobotStudio has pre-defined XML files that are imported and used for robot controller systems with the appropriate RobotWare options. These XML files have both the Move and Action instructions.

It is recommended to use RobotStudio ArcWelding PowerPac while using RobotWare Arc.

The instruction template supports the following Robotware options:

- Cap (Continuous Application Process)
- Disp (Dispense)
- Trigg (Fixed Position Events)
- Spot Pneumatic
- Spot Servo
- Spot Servo Equalizing
- Paint



xx0600003320

Item	Description
1	Buttons for importing, exporting and validating.

Continues on next page

Item	Description
2	The instruction template tree. This hierarchal tree set organizes the templates. Templates are always the lowest level nodes. For details about specific nodes in the tree, see item 5 and below.
3	Brief description for editing and creating instruction templates.
4	The <b>Instruction grid</b> . All arguments and settings for the object selected in the tree are displayed here. Only white boxes are editable. Red values indicate that the values are invalid.
5	The <b>Instruction templates</b> top node. Here you can see to which task the templates belong.
6	The <b>Action instructions</b> node contains everything related to action instruction templates.
7	An <b>Action instruction description</b> node, here represented by the <i>Set DO</i> instruction, defines the arguments that can be set for the action instruction templates of that kind. You can create action instruction descriptions for all action instructions known by the system running on the virtual controller.
8	An <b>Action instruction template</b> node, here represented by <i>Default</i> , contains instances of the action instruction descriptions, with defined values for the arguments.
9	The <b>Move instructions</b> node contains everything related to move instruction templates.
10	The <b>Move instruction descriptions</b> node contains all move instructions descriptions for the task. If the description for an instruction is not present in the list, right-click this node to add it. You can create move instruction descriptions for all move instructions known by the system running on the virtual controller.
11	A <b>Move instruction description</b> node, here represented by the <i>MoveAbsJ</i> node, defines the arguments that can be set for the move instruction templates of that kind. Unlike action instructions, instruction templates related to a certain move instruction descriptions are not stored in child nodes under the description, due to a more complex hierarchy.
12	The <b>Process definitions</b> node, which gathers all process definitions, contains sets of process templates which in turn contain instruction templates optimized for specific processes.
13	A <b>Process definition</b> node, here represented by the generic <i>Move</i> process, contains sets of process templates which in turn contain instruction templates optimized for specific processes.
14	A <b>Process template</b> node, here represented by the generic <i>Default</i> process, contains sets of move instruction templates with argument values optimized for specific processes. A process template can hold one move instruction template for each move instruction type defined by a move instruction description.
15	A <b>Move instruction template</b> node, here represented by <i>MoveJ</i> , contains instances of move instruction descriptions with argument values customized for specific processes.

### Importing a template

- 1 Click **Import** to bring up the **Open File** dialog box.
- 2 Select the file to import, and click **OK**.

*Continues on next page*

## 8 Home tab

---

### 8.16 Instruction Template Manager

*Continued*

---

#### Exporting a template

- 1 Select an exportable node in the tree view and click **Export** to bring up the **Save File** dialog box.
- 2 Click **OK**.

---

#### Validating the templates

- 1 Select a node in the tree view and click **Validate**.  
Any invalidity will be reflected by the icons and ToolTips of the respective node and reported in the Output window.

## **8.17 Settings**

### **8.17.1 Task**

---

#### **Selecting a Task**

Select a task from the **Task** drop-down list. The selected task indicates active task, to which new workobjects, tooldata, target, empty path or path from curve will be added.

#### 8.17.2 Workobject

---

##### Selecting a Workobject

Select a workobject from the **Workobject** drop-down list. The selected workobject indicates active workobject, to which new targets and move instructions are added.

### **8.17.3 Tool**

---

#### **Selecting a Tool**

Select a tool from the **Tool** drop-down list. The selected tool indicates active tool, to which move instructions are added.

## 8 Home tab

---

### 8.18 The Freehand Group

### 8.18 The Freehand Group

---

#### Move

- 1 In the **Layout** browser, select the item you want to move.
- 2
- 3 Click **Move**.
- 4 In the graphics window, click one of the axes and drag the item into position.

*Continues on next page*

## **8.18.1 Rotate**

---

### **Rotating an item**

- 1 In the **Layout** browser, select the item you want to rotate.
- 2 Click **Rotate**.
- 3 In the graphics window, click one of the rotational rings and drag the item into position.

If you press the **ALT** key while rotating, the item will snap 10 degrees at a time.

#### 8.18.2 Jog Joint

---

##### Jogging the joints of a robot

- 1 In the **Layout** browser, select the robot you want to move.
- 2 Click **Jog Joint**.
- 3 Click the joint you want to move and drag it to the preferred position.

If you press the **ALT** key when jogging the joints of the robot, the robot will move 10 degrees at a time. If you press the **f** key, the robot will move 0.1 degree at a time.

### **8.18.3 Jog Linear**

---

#### **Jogging the TCP of a robot**

- 1 In the **Layout** browser, select the robot you want to move.
- 2 In the **Freehand** group, click **Jog Linear**. A coordinate system will be displayed at the TCP of the robot.
- 3 Click the axis you want to move and drag the TCP to the preferred position.  
If you press the **f** key while jogging the robot linearly, the robot will move with a smaller step size.

#### 8.18.4 Jog Reorient

---

##### Reorienting the TCP rotation

- 1 In the **Layout** browser, select the robot you want to reorient.
- 2 In the **Freehand** group, click **Jog Reorient**.  
An orientation ring appears around the TCP.
- 3 Click the orientation ring and drag the robot to rotate the TCP to the preferred position.  
The X, Y, and Z orientation appears with units.



##### Note

If you press the **ALT** key while reorienting, the robot moves by 10 units and if you press the **F** key, it moves by 0.1 unit.



##### Note

The behavior of orientation differs with the different reference coordinate system (World, Local, UCS, Active Workobject, Active Tool).

## **8.18.5 MultiRobot Jog**

---

### **Jogging robots in multirobot mode**

- 1 In the **Freehand** group, click **MultiRobot Jog**. Select the robots to be jogged from the list of available robots.
- 2 Select the jogging mode, jog one of the robots and the other ones will follow the movement.

## 8 Home tab

---

### 8.19 Graphics Tools

### 8.19 Graphics Tools

---

#### Overview

The **Graphics Tools** helps you to control the graphics view and to modify the appearance of objects. All available options are grouped under the following tabs.

- View Tab
- Edit Tab

*Continues on next page*

## 8.19.1 View Tab

### Introduction

Use the View tab to choose view settings, control graphics view and create new views, and to show/hide the selected targets, frames, paths, parts, and mechanisms. The View tab options are grouped under the following groups.

- View
- Navigate
- Markups
- Lights
- Clip Planes
- Freehand
- Close

### View Group

#### Introduction

The **View** group helps you to choose view settings, control graphics view and create new views, and to show/hide the selected targets, frames, paths, parts, and mechanisms. The following options are available:

- New View
- View Settings : Projection, Representation, Frame Size
- Settings
- Advanced Lighting
- Show/Hide

Select **New View** to create a new view.

#### View Settings

You can select the following different view settings.

Setting	Description
Projection <ul style="list-style-type: none"> <li>• Orthographic</li> <li>• Perspective</li> </ul>	To view the orthographic and perspective view of the object.
Representation <ul style="list-style-type: none"> <li>• Surface</li> <li>• Wireframe</li> <li>• Both</li> <li>• Hidden-line removal</li> </ul>	To view the objects as a surface, wireframe, both surface and wireframe, by removing hidden lines.
Frame Size <ul style="list-style-type: none"> <li>• Large</li> <li>• Medium</li> <li>• Small</li> </ul>	To view the frame in large, medium and small sizes.

*Continues on next page*

## 8 Home tab

---

### 8.19.1 View Tab

*Continued*

#### Advanced Lighting

Advances Lighting activates the advanced lighting model that enables features such as multiple lights, shadows and environment mapping. This feature requires graphics hardware that supports Direct3D feature level 10\_1 or higher.

The standard lighting model is supported on all systems and is optimized for performance and visibility.

#### Settings

The **Settings** button in the **View** tab provides various display options for graphics. The **Settings** commands work on the active graphics view and do not affect persisted settings or other views. The available options are:

Settings	Description
Show Floor	Shows or hides the floor the graphics view.
Show UCS Grid	Shows or hides the UCS grid in the graphics view.
Show World Coordinates	Shows or hides the world coordinates in the graphics view.
Show Buttons	Shows or hides buttons in the graphics view.
Reset Floor Size	Adjusts the floor and UCS grid to cover the area used by objects in the station.
Background Color	Allows the user to set a custom background for the view.
Default View Settings	Opens the <b>Options</b> dialog box that shows the defaults settings.
Reset to Defaults	Resets the active view to the default settings.

#### Show/Hide

You can either show or hide the following options:

- Target Names
- Frame Names
- Path Names
- All Targets/Frames
- All Paths
- All Parts
- All Mechanisms

*Continues on next page*

## Navigate Group

### Introduction

**Navigate** group contains buttons for creating and managing viewpoint and for controlling the orientation of graphics.

Setting	Description
View Orientation	To view the objects in the following different orientations. <ul style="list-style-type: none"> <li>• View All</li> <li>• View Center</li> <li>• Top</li> <li>• Bottom</li> <li>• Front</li> <li>• Back</li> <li>• Left</li> <li>• Right</li> </ul>
Create Viewpoint	Stores the location and direction of a virtual camera in the 3D environment.

### Viewpoint

A Viewpoint stores the location and direction of a virtual camera in the 3D environment. It stores points of interest in a station to create camera movements during simulation.

### Creating Viewpoint

You can create a viewpoint in a station in two ways:

- 1 In the **View** tab, click **Create Viewpoint**.
- 2 In the **Layout** browser, right-click the station and select **Create Viewpoint**.

A viewpoint is created and displayed (as an eye icon) in the layout browser to the left.

The position and direction of the Viewpoint can also be visualized as an arrow in the 3D graphics. By default, the newly created viewpoints are not visible and cannot be selected by clicking on the graphics.

### Viewpoint functions

In the Layout browser, right-click Viewpoint to perform the following functions:

Function	Description
Move to Viewpoint	Moves the active 3D view to the location stored in the viewpoint.
Set to current	Modifies the viewpoint to the current location and direction of the active 3D view. This action cannot be undone.
Visible	Toggles the visibility of the viewpoint 3D representation. This action cannot be undone.
Delete	Deletes the viewpoint. This action cannot be undone.
Rename	Renames the viewpoint. This action cannot be undone.

*Continues on next page*

## 8 Home tab

---

### 8.19.1 View Tab

*Continued*

#### Move to Viewpoint

You can also move active 3D view to the location stored in the Viewpoint using Event manager.

- 1 Create a viewpoint.
- 2 Add an event.  
The Create new event dialog box appears.
- 3 Select **Simulation** under **Activation** and **Simulation time** under **Event trig type**. Click **Next**.
- 4 Set the activation time. Click **Next**.
- 5 Select **Move to Viewpoint** from **Set Action type**. Click **Next**.
- 6 Select the viewpoint from **select Viewpoint** and set the transition time.
- 7 Click **Finish**.

Move to viewpoint function is also executed when replaying the simulation in a Station Viewer. In addition, you can switch between view points using the MoveToViewpoint Smart Components, see [Running a simulation on page 340](#).

---

## Markups Group

### Overview

A markup is a text box displayed in the 3D graphics. It is similar to the temporary text shown when performing measurement or freehand movement, but it is part of the station.

The markup is displayed as a node in the layout browser and remains so when the station is saved. It appears as text bubble pointing to a position in the graphics window.

### Creating Markup

Use this procedure to create markup to an object

- 1 In the **Home** tab, click **View** and select **Create Markup**.

The **Create Markup** dialog box appears.



#### Note

Alternatively, in the **Layout** browser, right-click the station and select **Create Markup** for the dialog box to appear.

- 2 In the **Markup Text** field, enter a name for the markup text.
- 3 In the **Pointer position** field, set the position of the pointer.
- 4 Select **Always on top** if you want to display the text on top.
- 5 Click **Create**.

### Markup functions

In the Layout browser, right-click **Markup** to perform the following functions:

Function	Description
Visible	Shows or hides the markup in the 3D graphics.

*Continues on next page*

Function	Description
Modify Markup	Modifies the Markup properties.
Attach to	Attaches the markup to another graphical object.
Detach	Detaches the attached markup.
Delete	Deletes the markup.
Rename	Changes the name of the markup object.

## Modify Markup

Use this procedure to modify the markup properties:

- 1 In the **Layout** browser, right-click the markup and select **Modify Markup**.  
The **Modify Markup** dialog box appears.
- 2 Modify the markup text, pointer position or text position.
- 3 Click **Apply** for the changes to take effect.
- 4 Click **Close**.

## Lights Group

### Overview

The Lights group helps you to control the number and nature of light sources in the station when advanced lighting is enabled. There are four different kinds of light sources:

- Ambient Light – Controls the ambient (background) light level in the station.
- Infinite Light – A directional light source, similar to the sun, this light source can cast shadows.
- Spotlight – A light source with a cone of influence, this light source can cast shadows.
- Point light – Casts light radially from a specified position, this light source can not cast shadows.

By default, a new station contains two infinite light sources in addition to the ambient light source. The ambient light source cannot be removed or created, but it can be disabled. The **Create Light** menu contains commands for creating a new light source and adding it to the station.

### Light Properties

In the browser, right-click the light source and then select **Light Properties** from the context menu. You can view the **Light Properties** window. Use this window to modify the light source. The **Light Properties** window contains different controls depending on the type of light.

Settings	Description
Enabled (all kinds)	Enables or disables the light source
Casts shadows (infinite and spot lights)	Causes objects influenced by the light to cast shadows
Color (all kinds)	Controls the color of the light
Ambient intensity (ambient light)	Controls the ambient (background) intensity of the light

*Continues on next page*

## 8 Home tab

### 8.19.1 View Tab

*Continued*

Settings	Description
Diffuse intensity (infinite, spot and point lights)	Controls the intensity of diffuse highlights
Highlight (infinite, spot and point lights)	Controls the intensity of specular highlight
Position (spot and point lights)	Controls the position of the light source
Direction (infinite and spot lights)	Controls the direction of the light
Limit range (spot and point lights)	Optionally limits the range of influence of the light
Spotlight angle (spot lights)	Controls the angle of the light cone

#### Light presets

Light sources are saved in the station. User can use light presets to save a set of lights for reuse. The **Presets** menu contains a list of user defined light presets. When a preset is selected, the light sources in the station are replaced by the preset. The menu also contains the following commands:

Command	Description
Reset lights to default	Resets light to the default preset
Save lights as preset	Saves the lights in the station as a preset
Edit presets	Allows the user to remove previously created presets

## Clip Planes

### Overview

A clip plane is an infinite plane that cuts through geometric objects in the station. Objects on one side of the plane are visible while objects on the other side are invisible. A station can contain multiple clip planes, but each graphics view can only have one active clip plane.



#### Note

Non-geometric objects such as graphic representations of paths and targets are not affected.

### Creating and Editing a Clip Plane

Use the following procedure to create and edit a clip plane.

- 1 In the **View** tab, click **Create Clip Plane**.
- 2 In the browser, right-click the clip plane. The context menu appears.
- 3 In the context menu, click **Edit**. The **Clip Plane** property window appears.

When a clip plane is selected in the browser, a graphical representation of the plane is displayed in the **Graphics** view. This allows the user to move and rotate the clip plane by using freehand controls.

### Clip Plane Functions

Function	Description
Position	Controls the position of the clip plane, you can select or type in the x, y and z co-ordinates of the clip plane.

*Continues on next page*

---

Function	Description
Normal	Controls the orientation of the clip plane, you can select or type in the x, y and z co-ordinates of the clip plane.
Flip	Allows the user to reverse the orientation of the clip plane.
Active	Enables or disables the clip plane in the active graphics view.

---

## Freehand

Freehand options are used to move and rotate the selected object. User can select the coordinate system.

### Moving an item

- 1 In the **Layout** browser, select the item you want to move.
- 2 Click **Move**.
- 3 In the graphics window, click one of the axes and drag the item into position.

### Rotating an item

- 1 In the **Layout** browser, select the item you want to rotate.
- 2 Click **Rotate**.
- 3 In the graphics window, click one of the rotational rings and drag the item into position.

If you press the **ALT** key while rotating, the item will snap 10 degrees at a time.

## World

Use this option to move or rotate an object in the station in relation with the specified coordinate system. The following options are available:

- World
- Local
- UCS
- Active Workobject
- Active Tool

---

## Close Graphics

Select this option to close the **Graphics Tools** tabs.

## 8 Home tab

---

### 8.19.2 Edit Tab

### 8.19.2 Edit Tab

---

#### Overview

The **Edit** tab contains commands for working with materials and their application on geometric objects. The available options are:

- Materials
- Pick Material
- Apply Material
- Edit Materials
- Adjust Textures
- Close Graphics

#### Appearance Group

The **Appearance** group contains the following options:

Settings	Description
Materials	Contains a gallery of predefined and user defined materials
Pick Material	Allows the user to select a material
Apply Material	Activates or deactivates apply material mode
Edit Materials	Allows editing of user defined materials
Adjust Textures	Allows the user to adjust texture coordinates on surfaces directly in the main graphics view

#### Materials

The **Material** menu contains controls for setting material parameters or for selecting a material from the list of user defined and predefined materials. It is also possible to save the current material to the user defined materials list.

#### Apply Material

When **Apply Material** is activated, the active material is automatically applied to the geometric surfaces or objects (depending on selection mode) selected in the Graphics view. The active material is selected by using the Materials menu or the Pick Material command. When a material is selected, Apply Material is automatically activated.

#### Edit Materials

The **Edit Materials** menu allows editing of user defined materials. Click this button to see the list of defined materials. This list can be edited by adding, deleting and copying materials. The name and description of a material can be changed by right-clicking in the list. A preview of the selected material is displayed in the Graphics view.

#### Adjust Textures

The **Adjust Textures** menu allows the user to adjust texture coordinates on surfaces directly in the main Graphics view. When the command is activated and a textured surface is selected, the user can use the keyboard to move, rotate, mirror and scale the texture on the surface.

*Continues on next page*

**Graphics browser**

A separate browser window is displayed when one of the graphics menu is selected. It contains objects in the station that are related to the graphics system. The following options are available:

- Markups
- Viewpoints
- Lights
- Clip planes

---

**Close Graphics**

Select this option to close the **Graphics Tools** tabs .

**This page is intentionally left blank**

## 9 Modeling tab

### 9.1 Overview

---

**The Modeling tab**

The Modeling tab contains the controls for creating and grouping components, creating bodies, measurements and CAD operations.

## 9.2 Component Group

---

### Creating a component group

- 1 Click **Component Group**. The **Group** node will be displayed in the **Layout** browser.
- 2 Click the objects to add to the group. Drag them to the **Group** node.

## **9.3 Empty Part**

---

### **Creating an empty part**

- 1 Click **Empty Part**. The **Part** node will be displayed in the **Layout** browser.

## 9 Modeling tab

---

### 9.4.1 Smart Component

## 9.4 Smart Component

### 9.4.1 Smart Component

---

#### Overview

Smart Component is a RobotStudio object (with or without a 3D graphical representation) that has the behavior which can be implemented by code-behind and/or aggregation of other Smart Components.

---

#### Terminology

The following table describes the different terminologies that you come across when working with Smart Component.

Term	Definition
<b>Code behind</b>	A .NET class associated with a Smart Component that can implement custom behavior by reacting to certain events, for example simulation time steps and changes in property values.
<b>[Dynamic] property</b>	An object attached to a Smart Component that has value, type and certain other characteristics. The property value is used by code behind to control the behavior of the Smart Component.
<b>[Property] binding</b>	Connects the value of one property to the value of another property.
<b>[Property] attributes</b>	Key-value pairs that contain additional information about a dynamic property, for example value constraints.
<b>[I/O] signal</b>	An object attached to a Smart Component that has a value and a direction (input/output), analogous to I/O signals on a robot controller. The signal value is used by code behind to control the behavior of the Smart Component.
<b>[I/O] connection</b>	Connects the value of one signal to the value of a different signal.
<b>Aggregation</b>	The process of connecting several Smart Components using bindings and/or connections in order to implement a more complex behavior.
<b>Asset</b>	Data object contained in a Smart Component. Uses include code behind assembly and localized resources.

---

## 9.4.2 Smart Component Editor

---

### Overview

The Smart Component Editor allows you to create, edit, and aggregate Smart Components using a graphical user interface. It is an alternative to using the xml based library compiler.

---

### Layout of a Smart Component Editor

It consists of an icon, the name, the description for the component wherein the description can be modified by typing in the text box, and a combo box.

The combo box specifies the language for editing localized strings (captions and descriptions) in the component. The default language is always English, even if the application language is different. For more information, see [Assets on page 267](#).

The Smart Component Editor consists of the following tabs:

- [The Compose tab on page 266](#)
- [The Properties and Bindings tab on page 269](#)
- [The Signals and Connections tab on page 272](#)
- [The Design tab on page 275](#)

---

### Opening a Smart Component Editor

Click **Smart Component** or select **Edit Component** from the context menu.

The **Smart Component Editor** window appears.

---

### Protecting a Smart Component from edits

You can protect a Smart Component from being edited. To protect the smart component, right-click the smart component, and then click **Protected**. You can also optionally specify a password that will be required to unlock the component for edits.

Protecting a Smart Component hides its internal structure and protects it from edits. You can use this feature to hide the complexity in the smart component, and to secure the functionality. Child components in a protected smart component are hidden in all RobotStudio browsers and also in the Signal Analyzer browser.



#### Note

Protecting a Smart Component in this manner is a way of hiding complexity, and is not for providing security or securely protecting it.

## 9 Modeling tab

### 9.4.3 The Compose tab

### 9.4.3 The Compose tab

#### Overview

The Compose tab consists of the following:

- [Child components on page 266](#)
- [Saved States on page 267](#)
- [Assets on page 267](#)

#### Child components

It is a list box that displays all the objects contained by the component. Objects connected to a library have an overlay that indicates that the objects are locked. Smart Components are displayed first followed by other type of objects.

If an object is selected from the list, the following commands are displayed in the right panel:

Command	Description
<b>Add component</b>	Adds a child object to the component from the list. You can select a built-in base Smart Component, a new empty Smart Component, a library from file or a geometric part from file. Base components are organized as sub-menus based on the usage. For example, <i>Signals and Properties</i> , <i>Sensors</i> , <i>Actions</i> and so on. Recently used base components are listed at the top. For more information about base Smart Component, see <a href="#">Basic Smart Components on page 276</a> .
<b>Edit parent</b>	Sets the context of the Editor to the parent of the component that is currently being edited. If the parent is the station, see <a href="#">Station Logic on page 338</a> .
<b>Disconnect from library</b>	Disconnects the selected object from library, allowing it to be edited.
<b>Export to XML</b>	Opens a dialog box where you can export and save the component definition along with its properties as an *.rsxml file

Right-click on the selected object to display the following context menu items

Item	Description
<b>Edit</b>	Sets the context of the Editor to the selected child object.
<b>Delete</b>	Deletes the child object.
<b>Show in Browser</b>	Indicates if the object should be displayed in the Layout browser.
<b>Set as Role</b>	Sets the object as the <b>Role</b> of the component. The Smart Component will inherit certain characteristics of the Role object. For example, attaching a component with a tool as Role to a robot will cause a ToolData to be created.
<b>Properties</b>	Opens the Property editor dialog box for the object. See <a href="#">Property Editor on page 293</a> .

*Continues on next page*

## Saved States

The state of the component can be saved to be restored later. The state contains selected modifiable aspects of the component and its child components at the time when the state was saved. The following commands are available:

Command	Description
Save Current State	Opens the <b>Save Current State</b> dialog box. See <a href="#">Save Current State on page 267</a> .
Restore Selected State	Restores the component to the selected state.
Details	Opens a window that displays detailed information about the selected state.
Delete	Deletes the selected state.

## Save Current State

- 1 In the **Name** text box, enter a name for the state. If a state with the same name already exists, you will be asked to overwrite the existing state.
- 2 In the **Description** text box, enter the description for the state.
- 3 In the **Values to save**, select the value to be saved.
- 4 Select the check box to save the state of all child components.



### Note

When working on a station level,

- In the **Values to save**, you can also select certain Virtual Controller values in the saved states.
- You need not select the option **Recursive** as the state of the station is always saved.

For more information, see [Station Logic on page 338](#).

## Assets

The assets contained in the component are displayed as grid.

The following commands are available:

Command	Description
Add Asset	Opens a dialog box and allows you to browse and select any file as an asset.
Set Icon	Opens a dialog box and allows you to browse and select an image that represents a Smart Component.
Update All Assets	Replaces the data of all the assets with the data of the corresponding file on the disk. If the file is not available, a warning message is displayed in the output window.
View	Opens the selected asset in the associated program
Save	Opens a dialog box and allows you to save the selected asset.
Delete	Deletes the selected asset.

Continues on next page

## 9 Modeling tab

---

### 9.4.3 The Compose tab

*Continued*



#### Note

The text resources (descriptions) for properties and signals are stored in an asset called *Resources.<language-id>.xml*. If this is deleted, the texts for that language will be empty and the default (English) will be used. The default language when authoring a component is always English, regardless of the application language.

## 9.4.4 The Properties and Bindings tab

### Overview

The Properties and Bindings tab consists of the following:

- [Dynamic Properties on page 269](#)
- [Property Bindings on page 270](#)

### Dynamic Properties

The dynamic properties contained in the component are displayed in a grid.

The following commands are available

Command	Description
Add Dynamic Property	Opens the Add Dynamic Property dialog box. See <a href="#">Add or Edit Dynamic Property on page 269</a> .
Expose Child Property	Opens the Expose Child Property dialog box. See <a href="#">Expose Child Property on page 270</a> .
Edit	Opens the Edit Dynamic Property dialog box for the selected property.
Delete	Deletes the selected property.

### Add or Edit Dynamic Property

The Add Dynamic Property dialog box allows you to create a new dynamic property or edit an existing property.

The following controls are available:

Control	Description
Property Identifier	Specifies an identifier for the property. The identifier must be alphanumeric, start with a letter and must be unique.
Description	Detailed description of the property.
Read-only	Indicates if the property value should be possible to modify in GUI such as the Property Editor.
Property Type	Specifies the type of the property from a list of allowed types.
Property Value	Specifies the value of the property. The control is updated when you change the property type and/or attributes.
Attributes	Allows you to add, remove, and modify property attributes. The following lists the available attributes: <ul style="list-style-type: none"> <li>• MinValue</li> <li>• MaxValue</li> <li>• Quantity</li> <li>• Slider</li> <li>• AutoApply</li> </ul> Numeric attributes are specified in SI units.



#### Note

When editing an existing property, the Identifier and Type controls are locked and cannot be modified. If the input is valid the OK button is enabled, allowing the you to add or update the property. If not, an error icon is displayed.

*Continues on next page*

## 9 Modeling tab

---

### 9.4.4 The Properties and Bindings tab

*Continued*

#### Expose Child Property

The **Expose Child Property** dialog box allows you to add a new property and bind to an existing property in a child object. The new property will have the same type and attributes as the child property.

The following controls are available:

Control	Description
Property Identifier	Identifier of the new property. By default, this is the same as the identifier of the selected child property.
Binding Direction	Specifies the direction of the property binding
Source or Target Object	Specifies the child object for which to expose a property.
Source or Target Property	Specifies the child property.

---

#### Property Bindings

The property bindings contained in the component are displayed in a grid.

The following commands are available:

Command	Description
Add Binding	Opens the <b>Add Binding</b> dialog box.
Add Expression Binding	Opens the <b>Add Expression Binding</b> dialog box.
Edit	Opens the <b>Edit Binding</b> or <b>Edit Expression Binding</b> dialog box, depending on the type of binding selected.
Delete	Deletes the selected binding.

#### Add or Edit Binding

The **Add Binding** dialog box allows you to create or edit a property binding.

The following options are available:

Control	Description
Source Object	Specifies the owner of the source property.
Source Property	Specifies the source of the binding.
Target Object	Specifies the owner of the target property.
Target Property	Specifies the target of the binding. Only properties of the same type as the source property type are listed.
Allow cyclic binding	Allows the target property to be set two times in the same context, which otherwise generates an error. The target list box, besides dynamic properties also displays some common properties such as object transform that can only be used as target and not as source.

*Continues on next page*

**Add or Edit Expression Binding**

The **Add Expression Binding** dialog box allows you to specify a mathematical expression as the source of a property binding.

The following controls are available:

Control	Description
<b>Expression</b>	<p>Specifies the mathematical expressions.</p> <p>The following lists the allowable mathematical expressions:</p> <ul style="list-style-type: none"><li>• <b>Allowed operators:</b> +, - (unary and binary) *, /, ^ (power), Sin(), Cos(), Sqrt(), Atan() and Abs().</li><li>• <b>Allowed operands:</b> Numeric constants, PI, and Numeric dynamic properties on the current smart component and any child smart components.</li></ul> <p>The text box has the IntelliSense-like functionality which allows you to select from the available properties. If the expression entered in the text box is invalid, an error icon is displayed.</p>
<b>Target Object</b>	Specifies the owner of the target property.
<b>Target Property</b>	<p>Specifies the target of the binding.</p> <p>Only numeric properties are listed.</p>

## 9 Modeling tab

### 9.4.5 The Signals and Connections tab

### 9.4.5 The Signals and Connections tab

#### Overview

The Signals and Connections tab consists of the following:

- [I/O Signals on page 272](#)
- [I/O Connections on page 273](#)

#### I/O Signals

The I/O Signals contained in the component are displayed in a grid.

The following commands are available:

Command	Description
Add I/O Signals	Opens the <b>Add I/O Signals</b> dialog box.
Expose Child Signal	Opens the <b>Expose Child Signal</b> dialog box.
Edit	Opens the <b>Edit Signal</b> dialog box.
Delete	Deletes the selected signal.

#### Add or Edit I/O signals

The **Add I/O Signals** dialog box allows you to edit an I/O signal, or add one or more I/O signals to the component.

The following controls are available:

Control	Description
Type of Signal	Specifies the type and direction of the signal. The following are the available types of signals: <ul style="list-style-type: none"><li>• Digital</li><li>• Analog</li><li>• Group</li></ul>
Signal Base Name	Specifies the name of the signal. The name must contain an alphanumeric character and start with a letter (a-z or A-Z). If more than one signal is created, numeric suffixes specified by Start Index and Step are added to the names.
Signal Value	Specifies the initial value of the signal.
Description	Text that describes the signal. When creating multiple signals, all will have the same description.
Auto-reset	Specifies that a digital signal should have transient behavior. This applies to digital signals only. Indicates that the signal value is automatically reset to 0.
Number of Signals	Specifies the number of signals to create.
Start Index	Specifies the first suffix when creating multiple signals.
Step	Specifies the suffix interval when creating multiple signals.
Minimum	Specifies the minimum value for an analog signal. This applies to Analog signal only.
Maximum	Specifies the maximum value for an analog signal. This applies to Analog signal only.

*Continues on next page*

Control	Description
<b>Hidden</b>	Indicates if the property should not be visible in GUI such as the Property Editor and I/O Simulator.
<b>Read only</b>	Indicates if the property value should be possible to modify in GUI such as the Property Editor and I/O Simulator.

**Note**

When editing an existing signal, only the **Signal Value** and **Description** can be modified, while all other controls are locked.

If the input is valid, **OK** is enabled allowing you to create or update the signal. If not, an error icon is displayed.

**Expose Child signal**

The **Expose Child Signal** dialog box allows you to add a new I/O signal that is connected to a signal in a child object.

The following controls are available:

Control	Description
<b>Signal Name</b>	Specifies the name of the signal to be created. By default, it is the same as the name of the selected child signal.
<b>Child Object</b>	Specifies the object for which to expose a signal.
<b>Child Signal</b>	Specifies the child signal.

**I/O Connections**

The **I/O Connections** contained in the component are displayed in a grid.

The following controls are available:

Control	Description
<b>Add I/O Connection</b>	Opens the <b>Add I/O Connection</b> dialog box.
<b>Edit</b>	Opens the <b>Edit I/O Connection</b> dialog box.
<b>Manage I/O Connections</b>	Opens the <b>Manage I/O Connections</b> dialog box.
<b>Delete</b>	Deletes the selected connection.
<b>Move Up or Move Down</b>	Sorts by moving the selected connections up and down the list.

**Add or Edit I/O Connection**

The **Add I/O Connection** dialog box allows you to create an I/O connection or edit an existing connection.

The following controls are available:

Control	Description
<b>Source Object</b>	Specifies the owner of the source signal.
<b>Source Signal</b>	Specifies the source of the connection. The source must either be an output from a child component, or an input to the current component.
<b>Target Object</b>	Specifies the owner of the target signal.

*Continues on next page*

## 9 Modeling tab

### 9.4.5 The Signals and Connections tab

*Continued*

Control	Description
<b>Target Signal</b>	Specifies the target of the connection. The target must be of the same type as the source, and either an input to a child component or an output from the current component.
<b>Allow cyclic connection</b>	Allows the target signal to be set two times in the same context, which would otherwise generate an error.

#### Manage I/O Connections

The Manage I/O Connections dialog box displays a graphical view of the I/O connections of the component.

It allows you to add, remove, and modify connections. Only digital signals are displayed.

The following controls are available:

Control	Description
<b>Source / Target Signals</b>	Lists the signals used in the connections, source signals to the left and target signals to the right. Each signal is specified by the owner object and the signal name.
<b>Connections</b>	Displays connections as an arrow from source to target
<b>Logic Gates</b>	Specifies a logic operator and a delay time. It performs digital logic on input signals.
<b>Add</b>	<ul style="list-style-type: none"><li>• <b>Add Source</b> - Adds a source signal to the left.</li><li>• <b>Add Target</b> - Adds a target signal to the right.</li><li>• <b>Add Logic Gate</b> - Adds a logic gate at the center</li></ul>
<b>Remove</b>	Removes the selected signal, connection or logic gate.

#### Managing I/O connections

Use this procedure to add, remove, and create new I/O connections:

- 1 Click **Add** and select **Add Source** or **Add Target** or **Add Logic Gate** to add a source signal or target signal or logic gate respectively.
- 2 Move the cursor towards the **Source Signal** until a cross hair appears.
- 3 Click and drag the left mouse button towards the logic gate to create a new I/O connection.
- 4 Select the signal, connection, or logic gate and click **Remove** to delete.

## 9.4.6 The Design tab

### Overview

The Design tab displays a graphical view of the structure of the component. It includes the child components, internal connections, properties and bindings. The Smart Component can be organized on the viewing screen and their viewing position will be stored with the station.

### Using the Design tab

You can do the following in the Design tab:

Action	Description
Move child components and their positions.	<ul style="list-style-type: none"> <li>Click <b>Auto Arrange</b> to organize the components coherently.</li> <li>Use the Zoom slider to zoom the view.</li> </ul>
Select a component from the graphical view.	<p>Connections and bindings are color coded and highlighted to avoid confusion.</p> <p>By default, <b>Show Bindings</b>, <b>Show Connections</b>, and <b>Show unused</b> check-box are selected.</p> <ul style="list-style-type: none"> <li>De-select <b>Show Bindings</b> check-box to hide the bindings.</li> <li>De-select <b>Show Connections</b> check-box to hide the connections.</li> <li>De-select <b>Show unused</b> check-box to hide the unused components.</li> </ul>
Create connections and bindings	<ol style="list-style-type: none"> <li>Select the source signal or property. The cursor is displayed as a pen.</li> <li>Drag and drop the cursor on the target signal or property.</li> </ol> <p>If the target is valid, a connection and binding is created. If the target is invalid, the cursor changes to "not allowed" symbol.</p>

### 9.4.7 Basic Smart Components

#### Overview

The base components represent a complete set of basic building block components. They can be used to build user defined Smart Components with more complex behavior.

This lists the basic Smart Components available and are described in the following sections:

- [Signals and Properties on page 276](#)
- [Parametric Primitives on page 280](#)
- [Sensors on page 282](#)
- [Actions on page 285](#)
- [Manipulators on page 287](#)
- [Other on page 289](#)

#### Signals and Properties

##### LogicGate

The signal Output is set by the logical operation specified in Operator of the two signals InputA and InputB, with the delay specified in Delay.

Properties	Description
Operator	The logical operator to use. The following lists the various operators: <ul style="list-style-type: none"><li>• AND</li><li>• OR</li><li>• XOR</li><li>• NOT</li><li>• NOP</li></ul>
Delay	Time to delay the output signal.

Signals	Description
InputA	The first input.
InputB	The second input.
Output	The result of the logic operation.

##### LogicExpression

Evaluates a logic expression.

Properties	Description
String	The expression to evaluate.
Operator	The following lists the various operators: <ul style="list-style-type: none"><li>• AND</li><li>• OR</li><li>• NOT</li><li>• XOR</li></ul>

*Continues on next page*

Signals	Description
Result	Contains the result of the evaluation.

**LogicMux**

Output is set according to:  $\text{Output} = (\text{Input A} * \text{NOT Selector}) + (\text{Input B} * \text{Selector})$

Signals	Description
Selector	When low, the first input is selected. When high, the second input is selected.
InputA	Specifies the first input.
InputB	Specifies the second input.
Output	Specifies the result of the operation.

**LogicSplit**

The LogicSplit takes Input and sets OutputHigh to the same as Input, and OutputLow as the inverse of Input.

PulseHigh sends a pulse when Input is set to high, and PulseLow sends a pulse when Input is set to low.

Signals	Description
Input	Specifies the input signal.
OutputHigh	Goes high (1) when input is 1.
OutputLow	Goes high (1) when input is 0.
PulseHigh	Sends pulse when input is set to high.
PulseLow	Sends pulse when input is set to low.

**LogicSRLatch**

TheLogicSRLatch has one stable state.

- When Set=1, Output=1 and InvOutput=0
- When Reset=1, Output=0 and InvOutput=1

Signals	Description
Set	Sets the output signal.
Reset	Resets the output signal.
Output	Specifies output signal.
InvOutput	Specifies Inverse output signal.

**Converter**

Converts between property values and signal values.

Properties	Description
AnalogProperty	Converts to AnalogOutput.
DigitalProperty	Converts to DigitalOutput.
GroupProperty	Converts to GroupOutput.
BooleanProperty	Converts from DigitalInput and to DigitalOutput.

*Continues on next page*

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

Signals	Description
DigitalInput	Converts to DigitalProperty.
DigitalOutput	Converted from DigitalProperty.
AnalogInput	Converts to AnalogProperty.
AnalogOutput	Converted from AnalogProperty.
GroupInput	Converts to GroupProperty.
GroupOutput	Converted from GroupProperty.

#### VectorConverter

Converts between Vector3 and X, Y, and Z values.

Properties	Description
X	Specifies the X-value of Vector.
Y	Specifies the Y-value of Vector.
Z	Specifies the Z-value of Vector
Vector	Specifies the vector value..

#### Expression

The Expression consists of numeric literals (including PI), parentheses, mathematical operators +, -, \*, /, ^ (power) and mathematical functions sin, cos, sqrt, atan, abs. Any other strings are interpreted as variables, which are added as additional properties. The result is displayed in Result.

Signals	Description
Expression	Specifies the expression to evaluate.
Result	Specifies the result of evaluation.
NNN	Specifies automatically generated variables.

#### Comparer

The Comparer compares First value with Second value, using Operator. Output is set to 1 if the condition is met.

Properties	Description
ValueA	Specifies the first value.
ValueB	Specifies the second value.
Operator	Specifies the comparison operator. The following lists the various operators: <ul style="list-style-type: none"><li>• ==</li><li>• !=</li><li>• &gt;</li><li>• &gt;=</li><li>• &lt;</li><li>• &lt;=</li></ul>

Signals	Description
Output	True if the comparison evaluates to true, otherwise false.

*Continues on next page*

**Counter**

Count is increased when the input signal Increase is set, and decreased when the input signal Decrease is set. Count is reset when the input signal Reset is set.

Properties	Description
Count	Specifies the current count.
Signals	Description
Increase	Adds one to the Count when set to True.
Decrease	Subtracts one from the Count when set to True.
Reset	Resets the Count to zero when set to high.

**Repeater**

Pulses Output signal Count number of times.

Properties	Description
Count	Number of times to pulse Output.
Signals	Description
Execute	Set to high (1) to pulse Output Count times.
Output	Output pulse.

**Timer**

The Timer pulses the Output signal based on the given interval.

If Repeat is unchecked, one pulse will be triggered after the time specified in Interval. Otherwise, the pulse will be repeated at the interval given by Interval.

Properties	Description
StartTime	Specifies the time to pass before the first pulse.
Interval	Specifies the simulation time between the pulses.
Repeat	Specifies if the signal should be pulsed repeatedly or only once.
Current time	Specifies the current simulation time.
Signals	Description
Active	Set to True to activate the timer, and False to deactivate it.
Output	Sends pulses at the specified time intervals.

**StopWatch**

The StopWatch measures time during simulation (TotalTime). A new lap can be started by triggering the Lap input signal. LapTime shows the current lap time. The time is only measured when Active is set to 1. The times are reset when the input signal Reset is set.

Properties	Description
TotalTime	Specifies the accumulated time.
LapTime	Specifies the current lap time.
AutoReset	If true, TotalTime and LapTime will be set to 0 when the simulation starts.

*Continues on next page*

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

Signals	Description
Active	Set to True to activate the stop watch, and False to deactivate it.
Reset	Resets Total time and Lap time when set to high.
Lap	Starts a new lap.

#### Parametric Primitives

##### ParametricBox

The ParametricBox generates a box with dimensions specified by length, width, and height.

Properties	Description
SizeX	Specifies the length of the box in the X-axis direction.
SizeY	Specifies the length of the box in the Y-axis direction.
SizeZ	Specifies the length of the box in the Z-axis direction
GeneratedPart	Specifies the generated part.
KeepGeometry	False to remove the geometry data from the generated part. This can make other components such as Source execute faster.

Signals	Description
Update	Set to high (1) to update the generated part.

##### ParametricCircle

The ParametricCircle generates a circle with a given radius.

Properties	Description
Radius	Specifies the radius of the circle.
GeneratedPart	Specifies the generated part.
GeneratedWire	Specifies the generated wire object.
KeepGeometry	False to remove the geometry data from the generated part. This can make other components such as Source execute faster

Signals	Description
Update	Set to high (1) to update the generated part.

##### ParametricCylinder

The ParametricCylinder generates a cylinder with the dimensions given by Radius and Height.

Properties	Description
Radius	Specifies the radius of the cylinder.
Height	Specifies the height of the cylinder.
GeneratedPart	Specifies the generated part.
KeepGeometry	False to remove the geometry data from the generated part. This can make other components such as Source execute faster.

*Continues on next page*

Signals	Description
Update	Set to high (1) to update the generated part.

**ParametricLine**

The ParametricLine generates a line with a given end point or a given length. If either of them is changed, the other one will be updated accordingly.

Properties	Description
EndPoint	Specifies the end point for the line.
Length	Specifies the length of the line.
GeneratedPart	Specifies the generated part.
GeneratedWire	Specifies the generated wire object.
KeepGeometry	False to remove the geometry data from the generated part. This can make other components such as Source execute faster.

Signals	Description
Update	Set to high (1) to update the generated part.

**LinearExtrusion**

The LinearExtrusion extrudes SourceFace or SourceWire along the vector given by Projection.

Properties	Description
SourceFace	Specifies the face to extrude.
SourceWire	Specifies the wire to extrude.
Projection	Specifies the vector to extrude along.
GeneratedPart	Specifies the generated part.
KeepGeometry	False to remove the geometry data from the generated part. This can make other components such as Source execute faster.

**CircularRepeater**

The CircularRepeater creates a number of given copies of Source around the center of the SmartComponent with a given DeltaAngle.

Properties	Description
Source	Specifies the object to copy.
Count	Specifies the number of copies to create.
Radius	Specifies the radius of the circle.
DeltaAngle	Specifies the angle between the copies.

**LinearRepeater**

The LinearRepeater creates a number of copies of Source, with the spacing and direction given by Offset.

Properties	Description
Source	Specifies the object to copy.

*Continues on next page*

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

Properties	Description
Offset	Specifies the distance between copies.
Count	Specifies the number of copies to create.

#### MatrixRepeater

The MatrixRepeater creates a specified number of copies in three dimensions, with the specified spacing of the object in Source.

Properties	Description
Source	Specifies the object to copy.
CountX	Specifies the number of copies in the X-axis direction.
CountY	Specifies the number of copies in the Y-axis direction.
CountZ	Specifies the number of copies in the Z-axis direction.
OffsetX	Specifies the offset between the copies in the X-axis direction.
OffsetY	Specifies the offset between the copies in the Y-axis direction.
OffsetZ	Specifies the offset between the copies in the Z-axis direction.

#### Sensors

##### CollisionSensor

The CollisionSensor detects collisions and near miss events between the First object and the Second object. If one of the objects is not specified, the other will be checked against the entire station. When the Active signal is high and a collision or a near miss event occurs and the component is active, the SensorOut signal is set and the parts that participate in the collision or near miss event are reported in the first colliding part and second colliding part of the Property editor.

Properties	Description
Object1	The first object to check for collisions.
Object2	The second object to check for collisions.
NearMiss	Specifies the near miss distance.
Part1	The part of First object that has a collision.
Part2	The part of Second object that has a collision.
CollisionType	<ul style="list-style-type: none"><li>• None</li><li>• Near miss</li><li>• Collision</li></ul>

Signals	Description
Active	Specifies if the CollisionSensor is active or not.
SensorOut	True if there is a NearMiss or Collision.

*Continues on next page*

## LineSensor

The LineSensor defines a line by the Start, End, and Radius. When an Active signal is high, the sensor detects objects that intersect the line. Intersecting objects are displayed in the ClosestPart property and the point on the intersecting part that is closest to the line sensors start point is displayed in the ClosestPoint property. When intersection occurs the SensorOut output signal is set.

Properties	Description
Start	Specifies the start point.
End	Specifies the end point.
Radius	Specifies the radius.
SensedPart	Specifies the part that intersects the line sensor. If several parts intersect, then the one closest to the Start point is listed.
SensedPoint	Specifies the point on the intersecting part, closest to the Start point.

Signals	Description
Active	Specifies if the LineSensor is active or not.
SensorOut	True if the sensor intersects with an object.

## PlaneSensor

The PlaneSensor defines a plane by Origin, Axis1, and Axis2. When the Active input signal is set the sensor detects objects that intersect this plane. Intersecting objects are displayed in the SensedPart property and when intersection occurs the SensorOut output signal is set.

Properties	Description
Origin	Specifies the origin of the plane.
Axis1	Specifies the first axis of the plane.
Axis2	Specifies the second axis of the plane.
SensedPart	Specifies the part that intersects the PlaneSensor. If several parts intersect, then the one listed first in the Layout browser is selected.

Signals	Description
Active	Specifies if the PlaneSensor is active or not.
SensorOut	True if the sensor intersects with an object.

## VolumeSensor

The VolumeSensor detects objects that are inside or partly inside a box-shaped volume. The volume is defined by a Corner Point, the Length, Height, and Width of the sides and the orientation angles.

Properties	Description
CornerPoint	Specifies the local origin of the box.
Orientation	Specifies the orientation (Euler ZYX) relative to Reference.
Length	Specifies the length of the box.

*Continues on next page*

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

Properties	Description
Width	Specifies the width of the box.
Height	Specifies the height of the box.
Percentage	The percentage of the volume to react on. Set to 0 to react on all objects.
PartialHit	Allow an object to be sensed if only a part of it is inside the volume sensor.
SensedPart	The last object that either entered or left the volume.
SensedParts	The objects sensed in the volume
VolumeSensed	The total volume sensed

Signals	Description
Active	Set to high (1) to activate the sensor.
ObjectDetectedOut	Goes high (1) when an object is detected within the volume. Is reset immediately after an object has been detected.
ObjectDeletedOut	Goes high (1) when an object is detected to leave the volume. Is reset immediately after an object has left the volume.
SensorOut	Goes high (1) when the volume is full.

#### PositionSensor

The PositionSensor monitors the position and orientation of an object.

The position and orientation of an object is updated only during the simulation.

Properties	Description
Object	Specifies the object to monitor.
Reference	Specifies the reference coordinate system (Parent or Global).
ReferenceObject	Specifies the reference object, if Reference is set to Object.
Position	Specifies the position of the object relative to Reference.
Orientation	Specifies the orientation (Euler ZYX) relative to Reference.

#### ClosestObject

The ClosestObject defines a Reference object or a Reference point. When the Execute signal is set, the component finds the ClosestObject, ClosestPart, and the Distance to the reference object, or to the reference point if the reference object is undefined. If RootObject is defined, the search is limited to that object and its descendants. When finished and the corresponding properties are updated the Executed signal is set.

Properties	Description
ReferenceObject	Specifies the object to get the closest object to.
ReferencePoint	Specifies the point to get the closest object to.
RootObject	Specifies the object whose children to search. Empty means entire station.
ClosestObject	Specifies the object closest to Reference object or Reference point.

*Continues on next page*

Properties	Description
ClosestPart	Specifies the part closest to Reference object or Reference point.
Distance	Specifies the distance between the Reference object and the Closest object.
Signals	Description
Execute	Set to True to find the Closest part.
Executed	Sends a pulse when completed.

**Actions****Attacher**

The Attacher will attach Child to Parent when the Execute signal is set. If the Parent is a mechanism, the Flange to attach to must also be specified. When the input Execute is set, the child object is attached to the parent object. If Mount is checked, the child will also be mounted on the parent, with the Offset and Orientation specified. The output Executed will be set when finished.

Properties	Description
Parent	Specifies the object to attach to.
Flange	Specifies the Index of mechanism flange to attach to.
Child	Specifies the object to attach.
Mount	If true, the object to attach mounts on the attachment parent.
Offset	Specifies the position relative to the attachment parent when using Mount.
Orientation	Specifies the orientation relative to the attachment parent when using Mount.
Signals	Description
Execute	Set to True to attach.
Executed	Sends a pulse when completed.

**Detacher**

The Detacher will detach the Child from the object it is attached to when the Execute signal is set. If Keep position is checked, the position will be kept. Otherwise the child is positioned relative to its parent. When finished, the Executed signal will be set.

Properties	Description
Child	Specifies the object to detach.
KeepPosition	If false, the attached object is returned to its original position.
Signals	Description
Execute	Set to True to remove the attachment.
Executed	Sends a pulse when completed.

Continues on next page

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

#### Source

The Source property of the source component indicates the object that should be cloned when the Execute input signal is received. The parent of the cloned objects is specified by the Parent property and a reference to the cloned object is specified by the Copy property. The output signal Executed signifies that the clone is complete.

Properties	Description
Source	Specifies the object to copy.
Copy	Specifies the copied object.
Parent	Specifies the parent to the copy. If not specified, the copy gets the same parent as the source.
Position	Specifies the position of the copy relative its parent.
Orientation	Specifies the orientation of the copy relative its parent.
Transient	Marks the copy as transient if created during simulation. Such copies are not added to the undo queue and are automatically deleted when the simulation is stopped. This is used to avoid increased memory consumption during simulation.

Signals	Description
Execute	Set to True to create a copy of the object.
Executed	Sends a pulse when completed.

#### Sink

The Sink deletes the object referenced by the Object property. Deletion happens when the Execute input signal is received. The Executed output signal is set when the deletion is finished.

Properties	Description
Object	Specifies the object to remove.

Signals	Description
Execute	Set to True to remove the object.
Executed	Sends a pulse when completed.

#### Show

When the Execute signal is set, the object referenced in Object appears. When finished, Executed signal will be set.

Properties	Description
Object	Specifies the object to show.

Signals	Description
Execute	Set to True to show the object.
Executed	Sends a pulse when completed.

*Continues on next page*

**Hide**

When the Execute signal is set, the object referenced in Object will be hidden.  
When finished, Executed signal will be set.

Properties	Description
Object	Specifies the object to hide.
Signals	Description
Execute	Set to True to hide the object.
Executed	Sends a pulse when completed.

**Manipulators****LinearMover**

The LinearMover moves the object referenced in the Object property with a speed given by the Speed property in the direction given by the Direction property. The motion starts when the Execute input signal is set and stops when Execute is reset.

Properties	Description
Object	Specifies the object to move.
Direction	Specifies the direction to move the object.
Speed	Specifies the speed of movement.
Reference	Specifies the coordinate system in which values are specified. It can be Global, Local, or Object.
ReferenceObject	Specifies the reference object, if Reference is set to Object.
Signals	Description
Execute	Set to True to start move the object, and False to stop.

**Rotator**

The Rotator rotates the object referenced in the Object property with an angular speed given by the Speed property. The axis of rotation is given by CenterPoint and Axis. The motion starts when the Execute input signal is set and stops when the Execute is reset.

Properties	Description
Object	Specifies the object to rotate.
CenterPoint	Specifies the point to rotate around.
Axis	Specifies the axis of the rotation.
Speed	Specifies the speed of the rotation.
Reference	Specifies the coordinate system in which values are specified. It can be Global, Local, or Object.
ReferenceObject	Specifies the object which are relative to CenterPoint and Axis, if Reference is set to Object.
Signals	Description
Execute	Set to True to start rotating the object, and False to stop.

Continues on next page

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

#### Positioner

The Positioner takes an Object, Position, and Orientation as properties. When the Execute signal is set the object is repositioned in the given position relative to the Reference. When finished the Executed output is set.

Properties	Description
Object	Specifies the object to position.
Position	Specifies the new position of the object.
Orientation	Specifies the new orientation of the object.
Reference	Specifies the coordinate system in which values are specified. It can be Global, Local, or Object.
ReferenceObject	Specifies the object which are relative to Position and Orientation, if Reference is set to Object.

Signals	Description
Execute	Set to True to start move the object, and False to stop.
Executed	Set to 1 when operation is completed.

#### PoseMover

The PoseMover has a Mechanism, a Pose, and Duration as properties. When the Execute input signal is set the mechanism joint values are moved to the given pose. When the pose is reached the Executed output signal is set.

Properties	Description
Mechanism	Specifies the mechanism to move to a pose.
Pose	Specifies the Index of the pose to move to.
Duration	Specifies the time for the mechanism to move to the pose.

Signals	Description
Execute	Set to True, to start or resume moving the mechanism.
Pause	Pauses the movement.
Cancel	Cancels the movement.
Executed	Pulses high when the mechanism has reached the pose.
Executing	Goes high during the movement.
Paused	Goes high when paused.

#### JointMover

The JointMover has a Mechanism, a set of Joint Values and a Duration as properties. When the Execute input signal is set the mechanism joint values are moved to the given pose. When the pose is reached the Executed output signal is set. The GetCurrent signal retrieves the current joint values of the mechanism.

Properties	Description
Mechanism	Specifies the mechanism to move to a pose.
Relative	Specifies if J1-Jx are relative to the start values, rather than absolute joint values.
Duration	Specifies the time for the mechanism to move to the pose.

*Continues on next page*

Properties	Description
J1 - Jx	Joint values.
Signals	Description
GetCurrent	Retrieve current joint values.
Execute	Set to True to start moving the mechanism.
Pause	Pauses the movement
Cancel	Cancels the movement
Executed	Pulses high when the mechanism has reached the pose.
Executing	Goes high during the movement.
Paused	Goes high when paused.

**Other****GetParent**

The GetParent returns the parent object of the input object. The executed signal is triggered if a parent is found.

Properties	Description
Child	Specifies the object to whose parent is to be found.
Parent	Specifies the parent of the child
Signals	Description
Output	Goes high (1) if the parent exists.

**Note**

The **Child** list for **Properties:GetParent** does not show every part or object in the station. However, if you do not find the required part or object in the list, then add it from the browser or graphic window by clicking it.

**GraphicSwitch**

Switches between two parts, either by clicking on the visible part in the graphics or by setting and resetting the input signal.

Properties	Description
PartHigh	Displayed when the signal is high.
PartLow	Displayed when the signal is low.
Signals	Description
Input	Input signal.
Output	Output signal.

Continues on next page

## 9 Modeling tab

### 9.4.7 Basic Smart Components

*Continued*

#### Highlighter

The Highlighter temporarily sets the color of the Object to the RGB-values specified in Color. The color is blended with the original color of the objects as defined by Opacity. When the signal Active is reset, Object gets its original colors.

Properties	Description
Object	Specifies the object to highlight.
Color	Specifies the RGB-values of the highlight color.
Opacity	Specifies the amount to blend with the object's original color (0-255).

Signals	Description
Active	True sets the highlight. False restores the original color.

#### Logger

Prints a message in the output window.

Properties	Description
Format	Format string. Supports variables like {id:type}, where type can be d (double), i (int), s (string), o (object)
Message	Formatted message.
Severity	Message severity: 0 (Information), 1 (Warning), 2 (Error).

Signals	Description
Execute	Set to high (1) to print the message.

#### MoveToViewPoint

Moves to the selected viewpoint in the given time, when the input signal Execute is set. The output signal Executed is set when the operation is completed.

Properties	Description
Viewpoint	Specifies the viewpoint to move to.
Time	Specifies the time to complete the operation.

Signals	Description
Execute	Set to high (1) to start the operation.
Executed	Goes high (1) when the operation is completed.

#### ObjectComparer

Determines if ObjectA is the same as ObjectB.

Properties	Description
ObjectA	Specifies the object to compare.
ObjectB	Specifies the object to compare.

Signals	Description
Output	Goes high if the objects are equal.

*Continues on next page*

**Queue**

The Queue represents a FIFO (first in, first out) queue. The object in Back is added to the queue when the signal Enqueue is set. The front object of the queue is shown in Front. The object in Front is removed from the queue when the signal Dequeue is set. If there are more objects in the queue, the next object is shown in Front. All objects in the queue are removed from the queue when the signal Clear is set.

If a transformer component (such as LinearMover) has a queue component as its Object, it will transform the contents of the queue, rather than the queue itself.

Properties	Description
Back	Specifies the object to enqueue.
Front	Specifies the first object in queue.
Queue	Contains unique IDs of the queue's elements.
NumberOfObjects	Specifies the number of objects in the queue.

Signals	Description
Enqueue	Adds the object in Back to the end of the queue.
Dequeue	Removes the object in Front from the queue.
Clear	Removes all objects from the queue.
Delete	Removes the object in Front from the queue and from the station.
DeleteAll	Clears the queue and removes all objects from the station

**SoundPlayer**

Plays the sound specified by Sound Asset when the input signal Execute is set. The asset must be a .wav file

Properties	Description
SoundAsset	Specifies the sound file that should be played. Must be a .wav file.

Signals	Description
Execute	Set to high to play the sound.

**StopSimulation**

Stop a running simulation when the input signal Execute is set.

Signals	Description
Execute	Set to high to stop the simulation.

**Random**

Random generates a random number between Min and Max in Value when Execute is triggered.

Properties	Description
Min	Specifies minimum value.
Max	Specifies maximum value.
Value	Specifies a random number between Min and Max.

*Continues on next page*

## 9 Modeling tab

---

### 9.4.7 Basic Smart Components

*Continued*

Signals	Description
Execute	Set to high to generate a new random number.
Executed	Goes high when the operation is completed.

## 9.4.8 Property Editor

---

### Overview

The Property editor is used to modify the values of dynamic properties and I/O signals for a Smart Component. By default, the Property editor is displayed as a tool window to the left.

Each dynamic property is represented by a control. The type of control that is displayed depends on the property type and property attributes.

Properties with the Hidden flag set to true are not displayed. Read-only properties cannot be modified but are only displayed.

The values are validated according to the property attributes. If an invalid value is entered, an error icon is displayed next to the control and the **Apply** button is disabled.

If you set the **AutoApply** attribute of a property to true, the value is applied whenever you change the value in the control. You can apply the values of other properties by clicking the **Apply** button. If the component has no properties without **AutoApply**, then the **Apply** button will never be enabled.

You can toggle the value of a digital signal by clicking the control. Similarly, you can change the value of an analog or group signal by entering the new value in the text box.

---

### Opening the Property editor

You can open the Property editor dialog box in any one of the following ways:

- Right-click context menu for a Smart Component and select **Properties**.
- Launches automatically when the Smart Component Editor is started.
- Launched when you add a base component. See [Basic Smart Components on page 276](#).

## 9 Modeling tab

### 9.4.9 The Simulation Watch window

#### 9.4.9 The Simulation Watch window

##### Overview

The Simulation Watch allows you to monitor the values of dynamic properties and I/O signals in Smart Components. It specifies the simulation that should be paused when a value changes or meets a condition.

##### Layout of the Simulation Watch window

The Simulation Watch window by default takes the lower tab area in the RobotStudio GUI.

The window contains a list view of four columns with one row for each watch item:

Watch item	Description
Break	Specifies the Simulation break point and Break Condition. For more information, see <a href="#">Setting Breakpoints on page 295</a> .
Object	Specifies the object that owns the property or signal (for station signals the name of the station is displayed).
Property/Signal	Specifies the watched property or signal.
Value	Specifies the current value of the property or signal.

##### Adding and Deleting the Watch items

Use this procedure to add or delete a watch item.



##### Note

As a prerequisite, you should add smart component, its properties and signals. For more information, see [Smart Component Editor on page 265](#).

- 1 In the **Simulation Watch** window, right-click and select **Add** to display the Add submenu.

The Add submenu displays a recursive view of all the smart components, their properties and signals. The top level submenu displays the station signals.



##### Note

Watch items that are already watched are not displayed in the recursive view.

- 2 From the Add submenu, select property or signal to add a single property or signal of a component.
- 3 From the Add submenu, select **Add all** to add all the properties and signals of a component.

*Continues on next page*

- 4 In the **Simulation Watch** window, right-click on the row of the watch item and select **Delete** to delete one or more watch items.

**Note**

The Watch items are saved in the station and are restored when the station is opened.

---

**Setting Breakpoints**

You can set the breakpoint in one of the following ways:

- 1 To set the simulation breakpoint, select the check box next to a watch item.

**Note**

By default, the simulation is paused whenever the value of the property or signal changes.

- 2 In the **Simulation Watch** window, right-click on the row of the watch item and select **Break Condition**.

The **Break Condition** dialog box appears.

- Set the simulation to paused, when the value meets a certain logical condition. The condition can be viewed in the **Break** column of the Watch window.
- When a breakpoint is reached, the simulation is paused which is indicated by both the Play and Stop buttons being enabled.
- If the **Simulation Watch** window is hidden behind the other windows, it is brought to the front and the text of the corresponding watch item turns red.

**Note**

- Break condition can be specified only for the properties of numeric and string types, and not for the object types.
- After the current simulation time step is completed, all the remaining Smart component events are executed before the simulation is actually paused.

## 9.5 Tags

---

### Overview

For a complex RobotStudio station containing many robots, parts, paths, targets and other objects the browser and the 3D graphics view becomes cluttered. Tags function helps the user to identify objects in the cluttered 3D graphics view and in the browser during station modeling and offline programming.

Using tags function, you can group objects in a defined structure by labeling them. It is possible to hide or show these tags independent of the other tags. A hidden tagged object is invisible in the browser and in the 3D graphics window unless it is labeled by a currently visible tag.



#### Tip

For easy and quick access of Tags function, add it to the **Quick Access Toolbar** using the customize Commands.

---

### Creating a New Tag

Use any of the following steps to create a new tag.

- In the **Modeling** tab, select **Tags** and then click **New Tag**.
- Right-click the object and in the context menu, click **Tags** and then click **New Tag**. The object will be labeled by the tag

The new tag appears in the browser with the default name. Press F2 and select **Rename** from the context menu to rename the tag.

Use any of the following steps to hide or show tagged objects.

- In the **Tag** browser, right-click the tag and then check/uncheck **Visible**.
- Click the **Tags** menu and then check/uncheck the tag to show/hide the tag.

---

### Tag Visibility

When a tag is invisible/ unchecked, the corresponding tagged object will be hidden in the 3D graphics view and in the browsers (layout, path and targets, modeling). If an object is labeled by several tags, the object will be visible when at least one of the tags is set visible. Untagged objects are always visible.



#### Note

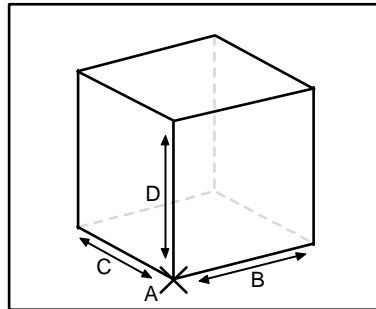
The regular visible property remains and overrides the tag visibility. If regular visible property is unchecked for an object, the object will be hidden in the 3D graphics view but it is visible in the browser. For a tagged object, when the visibility is unchecked, the object will be hidden in the 3D graphics view and in the browser.

## 9.6 Solid

### Creating a solid

- 1 Click **Solid** and then click the type of solid you want to create to bring up a dialog box.
- 2 Enter requested values in the dialog box and click **Create**. For detailed information about the specific dialog box for the curve to create, see below:

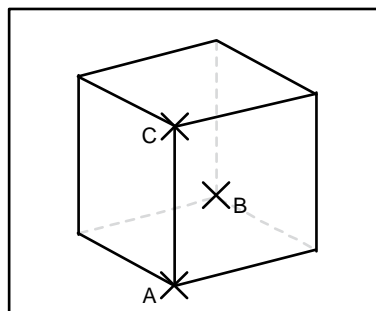
### The Create Box dialog box



xx060000

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Corner Point (A)</b>	Click in one of these boxes, and then click the corner point in the graphics window to transfer the values to the <b>Corner Point</b> boxes, or type the position. The corner point will be the local origin of the box.
<b>Orientation</b>	If the object shall be rotated relative to the reference coordinate system, specify the rotation.
<b>Length (B)</b>	Specify the box dimension along its X axis.
<b>Width (C)</b>	Specify the box dimension along its Y axis.
<b>Height (D)</b>	Specify the box dimension along its Z axis.

### The Create Box from 3-Points dialog box



xx060001

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
------------------	---

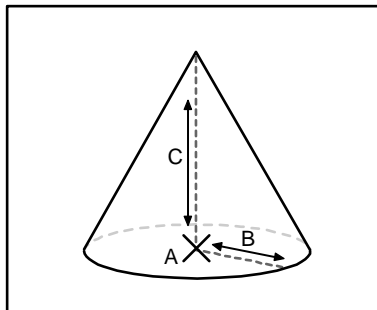
*Continues on next page*

## 9 Modeling tab

### 9.6 Solid Continued

<b>Corner Point (A)</b>	This point will be the local origin of the box. Either type the position, or click in one of the boxes and then select the point in the graphics window.
<b>Point on diagonal of XY-plane (B)</b>	This point is the the corner, diagonal to the local origin. It sets the X and Y directions of the local coordinate system, as well as the dimension of the box along these axes. Either type the position, or click in one of the boxes and then select the point in the graphics window.
<b>Indication Point Z-axis (C)</b>	This point is the corner above the local origin, It sets the Z direction of the local coordinate system, as well as the dimension of the box along the Z axis. Either type the position, or click in one of the boxes and then select the point in the graphics window.

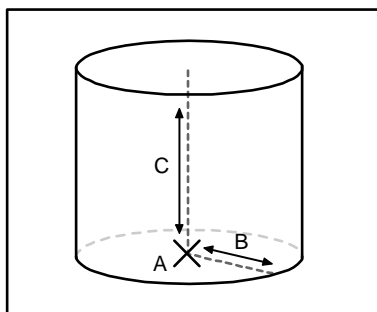
#### The Create Cone dialog box



xx060002

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Base Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Base Center Point</b> boxes, or type the position. The center point will be the local origin of the cone.
<b>Orientation</b>	If the object shall be rotated relative to the reference coordinate system, specify the rotation.
<b>Radius (B)</b>	Specify the radius of the cone.
<b>Diameter</b>	Specify the diameter of the cone.
<b>Height (C)</b>	Specify the height of the cone.

#### The Create Cylinder dialog box

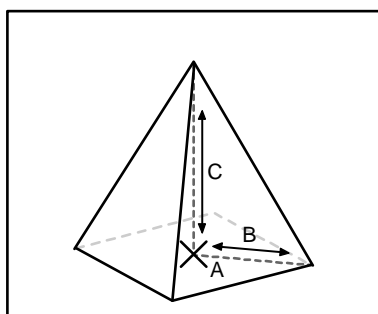


xx060003

Continues on next page

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Base Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Base Center Point</b> boxes, or type the position. The center point will be the local origin of the cylinder.
<b>Orientation</b>	If the object shall be rotated relative to the reference coordinate system, specify the rotation.
<b>Radius (B)</b>	Specify the radius of the cylinder.
<b>Diameter</b>	Specify the diameter of the cylinder.
<b>Height (C)</b>	Specify the height of the cylinder.

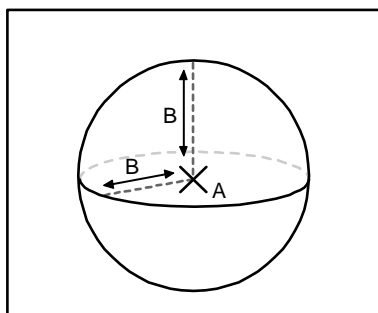
#### The Create Pyramid dialog box



xx060004

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Base Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Base Center Point</b> boxes, or type the position. The center point will be the local origin of the pyramid.
<b>Orientation</b>	If the object shall be rotated relative to the reference coordinate system, specify the rotation.
<b>Center to Corner Point (B)</b>	Either type the position, or click in the box and then select the point in the graphics window.
<b>Height (C)</b>	Specify the height of the pyramid.
<b>Number of Sides</b>	Specify the number of sides of the pyramid. The maximum number of sides is 50.

#### The Create Sphere dialog box



xx060005

Continues on next page

## 9 Modeling tab

---

### 9.6 Solid

*Continued*

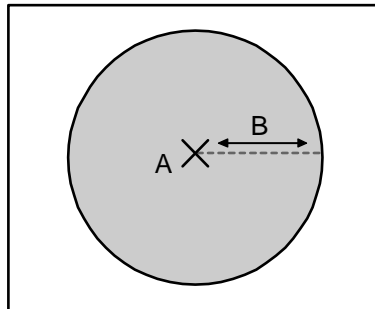
<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes, or type the position. The center point will be the local origin of the sphere.
<b>Radius (B)</b>	Specify the radius of the sphere.
<b>Diameter</b>	Specify the diameter of the sphere.

## 9.7 Surface

### Creating a surface

- 1 Click **Surface** and then click the type of solid you want to create to bring up a dialog box.
- 2 Enter requested values in in the dialog box and click **Create**. For detailed information about the specific dialog box for the curve to create, see below:

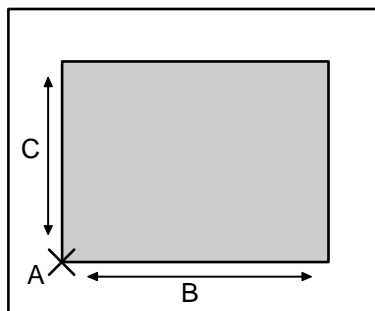
### The Create Surface Circle dialog box



xx060006

<b>Reference</b>	Select the <b>Reference</b> coordinate system, to which all positions or points will be related.
<b>Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes, or type the position. The center point will be the local origin of the circle.
<b>Orientation</b>	If the object shall be rotated relative to the reference coordinate system, specify the rotation.
<b>Radius (B)</b>	Specify the radius of the circle.
<b>Diameter</b>	Specify the diameter of the circle.

### The Create Rectangle dialog box



xx060007

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Start Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Start Point</b> boxes, or type the position. The start point will be the local origin of the rectangle.

*Continues on next page*

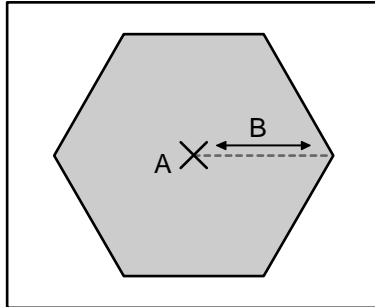
## 9 Modeling tab

### 9.7 Surface

*Continued*

<b>Orientation</b>	If the object shall be rotated relative the reference coordinate system, specify the rotation.
<b>Length (B)</b>	Specify the length of the rectangle.
<b>Width (C)</b>	Specify the width of the rectangle

#### The Create Surface Polygon dialog box



xx060008

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Center Point</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes, or type the position. The center point will be the local origin of the polygon.
<b>First Vertex Point</b>	Either type the position, or click in one of the boxes and then select the point in the graphics window.
<b>Vertices</b>	Specify the number of the vertices here. The maximum number of vertices is 50.

#### The Create Surface from Curve dialog box

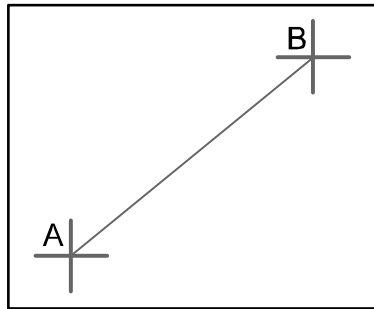
<b>Select Curve from graphics</b>	Select a curve by clicking it in the graphics window.
-----------------------------------	---

## 9.8 Curve

### Creating a curve

- 1 Click **Curve** and then click the curve you want to create to bring up a dialog box.
- 2 Enter requested values in in the dialog box and click **Create**. For detailed information about the specific dialog box for the curve to create, see below:

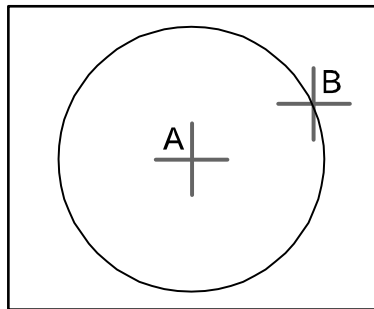
### The Create Line dialog box



xx050034

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Start point (A)</b>	Click in one of these boxes, and then click the start point in the graphics window to transfer the values to the <b>Start Point</b> boxes.
<b>End Point (B)</b>	Click in one of these boxes, and then click the end point in the graphics window to transfer the values to the <b>End Point</b> boxes.

### The Create Circle dialog box



xx050035

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Center point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes.
<b>Orientation</b>	Specify the orientation coordinates for the circle.
<b>Radius (A-B)</b>	Specify the radius of the circle.
<b>Diameter</b>	Alternatively, specify the diameter.

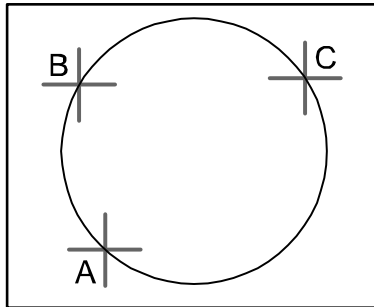
*Continues on next page*

## 9 Modeling tab

### 9.8 Curve

*Continued*

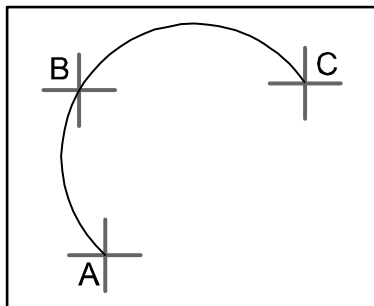
#### The Create Three Points Circle dialog box



xx050036

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>First Point (A)</b>	Click in one of these boxes, and then click the first point in the graphics window to transfer the values to the <b>First Point</b> boxes.
<b>Second Point (B)</b>	Click in one of these boxes, and then click the second point in the graphics window to transfer the values to the <b>Second Point</b> boxes.
<b>Third Point (C)</b>	Click in one of these boxes, and then click the third point in the graphics window to transfer the values to the <b>Third Point</b> boxes.

#### The Create Arc dialog box

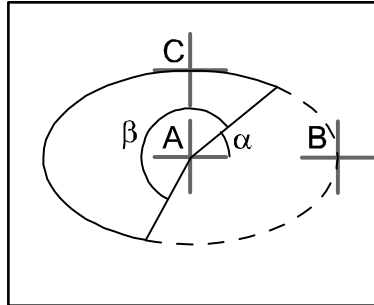


xx050037

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Start Point (A)</b>	Click in one of these boxes, and then click the start point in the graphics window to transfer the values to the <b>Start Point</b> boxes.
<b>Mid Point (B)</b>	Click in one of these boxes, and then click the second point in the graphics window to transfer the values to the <b>Mid Point</b> boxes.
<b>End Point (C)</b>	Click in one of these boxes, and then click the end point in the graphics window to transfer the values to the <b>End Point</b> boxes.

*Continues on next page*

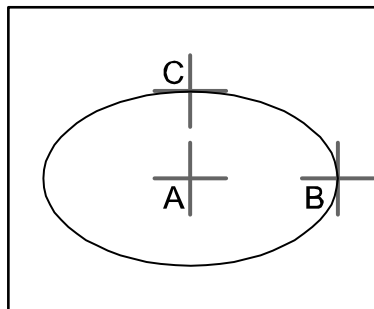
## The Create Elliptical Arc dialog box



xx050038

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes.
<b>Major Axis End Point (B)</b>	Click in one of these boxes, and then click the end point for the major axis of the ellipse in the graphics window to transfer the values to the <b>Major Axis End Point</b> boxes.
<b>Minor Axis End Point (C)</b>	Click in one of these boxes, and then click the end point for the minor axis of the ellipse in the graphics window to transfer the values to the <b>Minor Axis End Point</b> boxes.
<b>Start Angle (<math>\alpha</math>)</b>	Specify the start angle for the arc, measured from the major axis.
<b>End Angle (<math>\beta</math>)</b>	Specify the end angle for the arc, measured from the major axis.

## The Create Ellipse dialog box



xx050039

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Center Point (A)</b>	Click in one of the <b>Center Point</b> boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes.
<b>Major Axis End Point (B)</b>	Click in one of these boxes, and then click the end point for the major axis of the ellipse in the graphics window to transfer the values to the <b>Major Axis End Point</b> boxes.
<b>Minor Radius (C)</b>	Specify the length of the minor axis of the ellipse. The minor radius will be created perpendicular to the major axis.

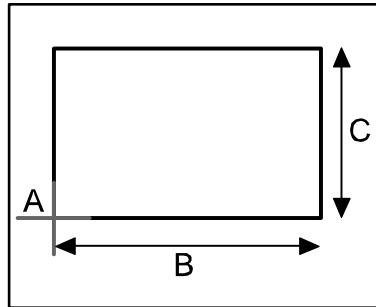
*Continues on next page*

## 9 Modeling tab

### 9.8 Curve

*Continued*

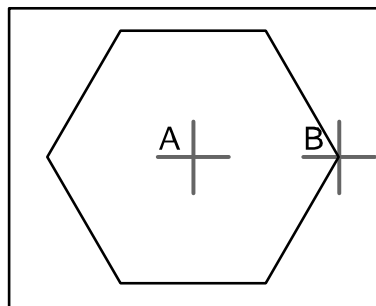
#### The Create Rectangle dialog box



xx050040

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Start Point (A)</b>	Click in one of these boxes, and then click the start point in the graphics window to transfer the values to the <b>Start Point</b> boxes. The rectangle will be created in the positive coordinate directions.
<b>Orientation</b>	Specify the orientation coordinates for the rectangle.
<b>Length (B)</b>	Specify the length of the rectangle along the x axis.
<b>Width (C)</b>	Specify the width of the rectangle along the y axis.

#### The Create Polygon dialog box

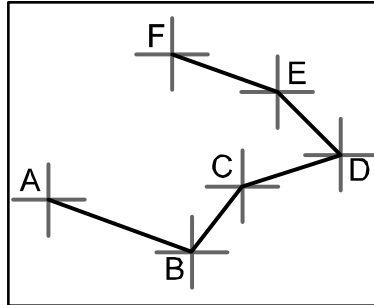


xx050041

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Center Point (A)</b>	Click in one of these boxes, and then click the center point in the graphics window to transfer the values to the <b>Center Point</b> boxes.
<b>First Vertex Point (B)</b>	Click in one of these boxes, and then click the first vertex point in the graphics window to transfer the values to the <b>First Vertex Point</b> boxes. The distance between the center point and the first vertex point will be used for all vertex points.
<b>Vertices</b>	Specify the number of points to be used when creating the polygon. The maximum number of vertices is 50.

*Continues on next page*

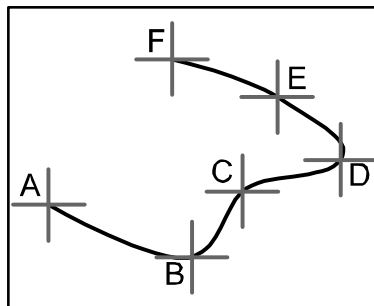
## The Create Polyline dialog box



xx050042

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Point Coordinates</b>	Specify each node of the polyline here, one at a time, by either typing the values, or by clicking in one of these boxes, and then selecting the point in the graphics window to transfer its coordinates.
<b>Add</b>	Click this button to add a point and its coordinates to the list.
<b>Modify</b>	Click this button to modify an already defined point, after you have selected it in the list and entered new values.
<b>List</b>	The nodes of the polyline. To add more nodes, click <b>Add New</b> , click the desired point in the graphics window, and then click <b>Add</b> .

## The Create Spline dialog box



xx050043

<b>Reference</b>	Select the <b>Reference</b> coordinate system to which all positions or points will be related.
<b>Point Coordinates</b>	Specify each node of the spline here, one at a time, by either typing the values, or by clicking in one of these boxes, and then selecting the point in the graphics window to transfer its coordinates.
<b>Ad</b>	Click this button to add a point and its coordinates to the list.
<b>Modify</b>	Click this button to modify an already defined point, after you have selected it in the list and entered new values.
<b>List</b>	This nodes of the spline. To add more nodes, click <b>Add New</b> , click the desired point in the graphics window, and then click <b>Add</b> .

## 9 Modeling tab

---

### 9.9 Border

### 9.9 Border

---

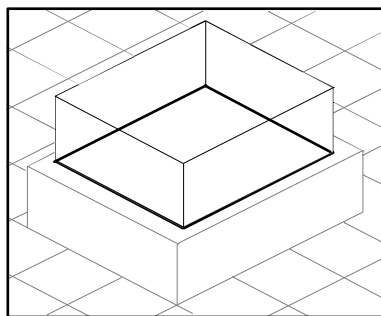
#### Creating a curve

- 1 Click **Border** and then click the border you want to create to bring up a dialog box.
- 2 Enter requested values in in the dialog box and click **Create**. For detailed information about the specific dialog box for the border to create, see below:

---

#### The Create Border Between Bodies dialog box

To use the create border between bodies command, the station must contain at least two objects.



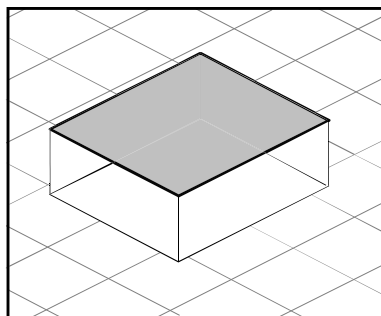
xx050044

<b>First Body</b>	Click in this box and then select the first body in the graphics window.
<b>Second Body</b>	Click in this box and then select the second body in the graphics window.

---

#### The Create Border Around Surface dialog box

To use the create border around surface command, the station must contain at least one object with a graphical representation.



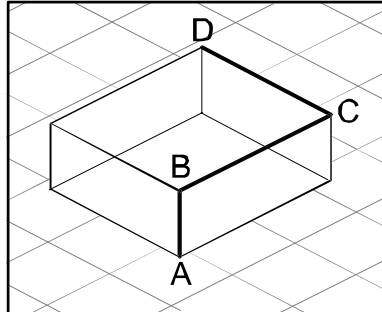
xx050045

<b>Select Surface</b>	Click in this box and then select a surface in the graphics window.
-----------------------	---

*Continues on next page*

**The Create Border From Points dialog box**

To use the create border from points command, the station must contain at least one object.

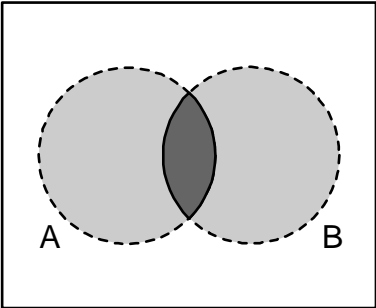


en050000

<b>Selected Object</b>	Click in this box and then select an object in the graphics window.
<b>Point Coordinates</b>	Specify the points that define the border here, one at a time, by either typing the values, or by clicking in one of these boxes, and then selecting the point in the graphics window to transfer its coordinates.
<b>Add</b>	Click this button to add a point and its coordinates to the list.
<b>Modify</b>	Click this button to modify an already defined point, after you have selected it in the list and entered new values.
<b>List</b>	The points that define the borders. To add more points, click <b>Add New</b> , click the desired point in the graphics window, and then click <b>Add</b> .

9.10 Intersect

The Intersect dialog box

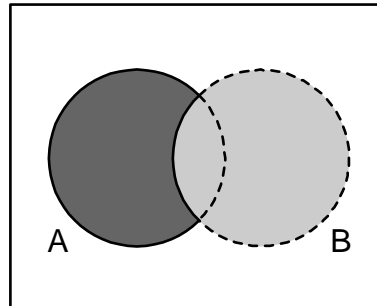


xx060009

<b>Keep Original</b>	Select this check box to keep the original bodies when creating the new body.
<b>Intersect... (A)</b>	Select the body from which you want to make an intersection (A) by clicking it in the graphics window.
<b>...and (B)</b>	Select the body with which you want to make an intersection (B) by clicking it in the graphics window. A new body will be created based on the common area between the selected bodies A and B.

## 9.11 Subtract

### The Subtract dialog box

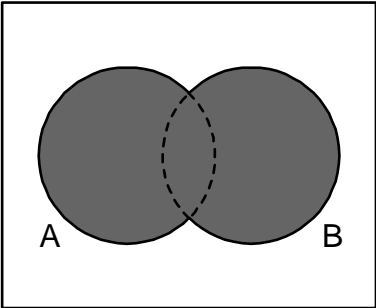


xx060010

<b>Keep Original</b>	Select this check box to keep the original bodies when creating the new body.
<b>Subtract... (A)</b>	Select the body from which you want to subtract (A) by clicking it in the graphics window.
<b>...with (B)</b>	Select the body you want to subtract (B) by clicking it in the graphics window. A new body will be created based on the area of body A subtracted with the common volume between body A and B.

9.12 Union

The Union dialog box



xx060011

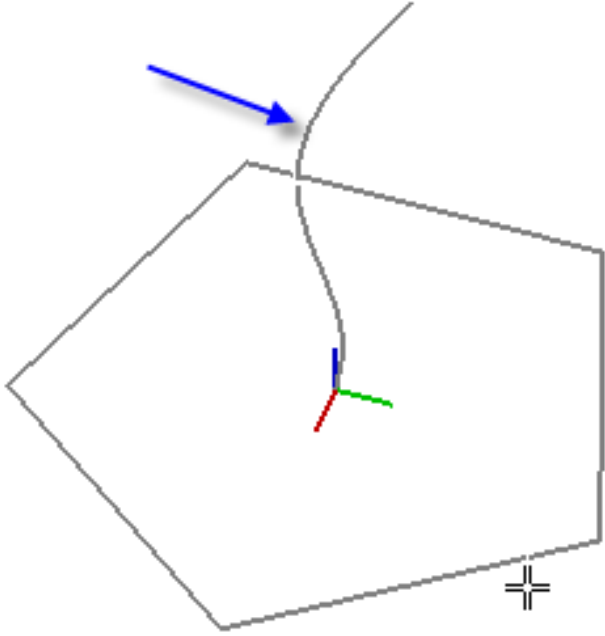
Keep Original	Select this check box to keep the original bodies when creating the new body.
Union... (A)	Select the body that you want to unify (A) by clicking it in the graphics window.
...and (B)	Select the body that you want to unify (B) by clicking it in the graphics window. A new body will be created based on the areas of the two selected bodies A and B.

## 9.13 Extrude Surface or Curve

### Extruding a surface or curve

- 1 From the selection level toolbar, select **Surface** or **Curve**, as appropriate.
- 2 In the graphics window, select the surface or curve you wish to extrude. Click **Extrude Surface** or **Extrude Curve**, as appropriate. The **Extrude Surface or Curve** dialog box opens below the **Modeling** browser.
- 3 For extrusion along a vector, fill in the values.  
For extrusion along a curve, select the **Extrude Along Curve** option. Click the **Curve** box, and select the curve in the **Graphics** window.
- 4 If you wish the form to appear as a surface model, clear the **Make Solid** check box.
- 5 Click **Create**.

### The Extrude Surface or Curve dialog box

Surface or Curve	Denotes the surface or curve to be extruded. To select the surface or curve, first click in the box, then select the surface or curve in the graphics window.
Extrude Along Vector	Enables extrusion along a specified vector.
From Point	The start point of the vector.
To Point	The end point of the vector.
Extrude Along Curve	Enables extrusion along a specified curve.
Curve	Denotes the curve used as a sweep path.  xx0600003076 To select the curve, first click the box, then the curve in the graphics window.

*Continues on next page*

9 Modeling tab

---

9.13 Extrude Surface or Curve  
*Continued*

Make Solid	Select this check box to convert the extruded form into a solid.
------------	--

## 9.14 Line from Normal

---

### Creating a line from normal

- 1 Click **Surface Selection**.
- 2 Click **Line to Normal** to bring up a dialog box.
- 3 Click on a face to select it in the **Select Face** box.
- 4 In the **Length** box, specify a length for the line.
- 5 Optionally, select the **Invert Normal** check box to invert the direction of the line.
- 6 Click **Create**.

### 9.15 The Measure Group



#### Tip

Make sure to select the appropriate snap mode and selection levels before making your measurements.

#### Measuring distances or angles

- 1 Click the type of measurement you want to use:

To measure the	Select
distance between two points you select in the graphics window.	<b>Point to point</b>
angle defined by three points you select in the graphics window. The first point to select is the converging point, thereafter you shall select one point on each line.	<b>Angle</b>
diameter, with the circle defined by three points you select from the graphics window.	<b>Diameter</b>
closest distance between two objects you select in the graphics window.	<b>Minimum distance</b>

The mouse pointer will turn into a ruler when you have activated any of the measurement functions.

- 2 In the graphics window, select the points or objects to measure between. Information about the measuring points is displayed in the **Output** window. The results will be displayed in the **Measurements** tab in the **Output** window when all points have been selected.
- 3 Optionally, repeat step 3 for making a new measurement of the same type.



#### Tip

You can also activate and deactivate the measurement functions from the measurement toolbar.

## 9.16 Create Mechanism

### Create a new mechanism

- 1 Click **Create Mechanism**.  
The Mechanism Modeler opens in create mode.
- 2 In the **Mechanism Model Name** box, enter a mechanism name.
- 3 From the **Mechanism Type** list, select a mechanism type.
- 4 In the tree structure, right-click **Links**, and then click **Add Link** to bring up the **Create Link** dialog box.  
A suggested name appears in the **Link Name** box.
- 5 In the **Selected Part** list, select a part (which will be highlighted in the graphics window) and click the arrow button to add the part to the **Parts** list box.  
The **Selected Part** list then automatically selects the next part, if any more are available. Add these, as required.



#### Note

Parts that are part of a library or mechanism cannot be selected.

- 6 Select a part in the **Parts** list box, enter any values in the **Selected Parts** group boxes, and then click **Apply to Part**.  
Repeat for each part, as required.
- 7 Click **OK**.
- 8 In the tree structure, right-click **Joints**, and then click **Add Joint** to bring up the **Create Joint** dialog box.  
A suggested name appears in the **Joint Name** box.
- 9 Complete the **Create Joint** dialog box, and then click **OK**.
- 10 In the tree structure, right-click **Frame/Tool Data**, and then click **Add Frame/Tool** to bring up the **Create Frame/Tool** dialog box.  
A suggested name appears in the **Frame/Tool Data** name box.
- 11 Complete the **Create Frame/Tool** dialog box, and then click **OK**.  
The validity criteria for the **Frame/Tool** node are as follows:
- 12 In the tree structure, right-click **Calibration**, and then click **Add Calibration** to bring up the **Create Calibration** dialog box.
- 13 Complete the **Create Calibration** dialog box, and then click **OK**.
- 14 In the tree structure, right-click **Dependency**, and then click **Add Dependency** to bring up the **Create Dependency** dialog box.
- 15 Complete the **Create Dependency** dialog box, and then click **OK**.
- 16 If all nodes are valid, compile the mechanism, see [Compiling a mechanism on page 318](#).

### Create Conveyor mechanism

- 1 Click **Create Mechanism**.

*Continues on next page*

## 9 Modeling tab

---

### 9.16 Create Mechanism

*Continued*

The Mechanism Modeler opens in create mode.

- 2 In the **Mechanism Model Name** dialog box, enter a mechanism name.
- 3 From the **Mechanism Type** list, select **Conveyor**.
- 4 From the **Selected Part** list, select **Part**.
- 5 In the **Position of Calibration frame** list, enter the base frame values relative to the local origin of the selected graphic component.
- 6 In the **Conveyor length** box, enter the length of the conveyor.  
The **Compile Mechanism** button is enabled.
- 7 In the **Attachment Points** box, set the **Pitch** and **Count** value.
- 8 Click **Add** to create new attachment points.
- 9 Click **Compile Mechanism**, to compile the mechanism. See [Compiling a mechanism on page 318](#).
- 10 In the **Layout** browser, right-click the conveyor mechanism and select **Save As Library**. Close the station.
- 11 Build a new system. See [Building a new system on page 161](#).  
On the **Modify Options** page of the **System Builder**, scroll down to the **Motion coordination part 3** group and select **606-1 Conveyor Tracking** check box.
- 12 Create new station using this new system. See [Robot System on page 206](#).  
After starting the system, when asked to select the library for the conveyor mechanism browse and select the already saved library.

---

### Compiling a mechanism

When compiling, a new mechanism, created in the create mode of the Mechanism Modeler, is added to the station with the default name "Mechanism\_" followed by an index number.

When compiling, an existing editable mechanism, modified in the modify mode of the Mechanism Modeler, is saved without any poses, joint mapping or transition times.

To compile a mechanism, follow these steps:

- 1 To compile a new or edited mechanism, click **Compile Mechanism**.  
The mechanism is inserted into the active station. The link parts are cloned with new names, but the corresponding links will update their part references. When the Mechanism modeler is closed, these cloned parts will be removed.
- 2 The Mechanism Modeler now switches to modify mode. To complete the mechanism, see below.

---

### Completing or modifying a mechanism

To complete the modeling of a mechanism, follow these steps:

- 1 If the values in the **Joint Mapping** group are correct, click **Set**.
- 2 Configure the Poses grid. To add a pose, click **Add** and then complete the **Create Pose** dialog box. Click **Apply**, followed by **OK**.  
To add a pose, click **Add** and then complete the **Create Pose** dialog box. Click **Apply**, followed by **OK**.

*Continues on next page*

To edit a pose, select it in the grid, click **Edit**, and then complete the **Modify Pose** dialog box. Click **OK**.

To remove a pose, select it in the grid and then click **Remove**.

3 Click **Edit Transition Times** to edit transition times.

4 Click **Close**.

#### The Create Mechanism dialog box

<b>Mechanism Model Name</b>	Specifies the model name of the mechanism.
<b>Mechanism Type</b>	Specifies the mechanism type.
<b>Tree structure</b>	The components of the mechanism in a tree structure. The tree structure will not be visible unless the mechanism is editable. Each node (link, joint, frame, calibration and dependency) can be edited in its own dialog box, see below.
<b>Compile Mechanism</b>	Click this button to compile the mechanism. This button will not be visible unless the mechanism is editable and the mechanism model name is valid.

#### The Create Conveyor Mechanism dialog box

<b>Mechanism Model Name</b>	Specifies the model name of the conveyor mechanism.
<b>Mechanism Type</b>	Specifies the different mechanism types.
<b>Selected Part</b>	Specifies the part to be selected for the conveyor.
<b>Position of Calibration frame</b>	Specifies the baseframe value relative to the local origin of the selected graphic component.
<b>Conveyor Length</b>	Specifies the length of the conveyor.
<b>Attachment Points</b>	Specifies the conveyor position to attach the workpieces.
<b>Compile Mechanism</b>	Click this button to compile the mechanism. This button will not be visible unless the mechanism is editable and the mechanism model name is valid.

#### The Create/Modify Link dialog box

A link is a moving component of a mechanism. Selecting a link node will highlight it in the graphics window.


<b>Link Name</b>	Specifies the name of the link.
<b>Selected Part</b>	Specifies the parts to add to the Part list box.
<b>Set as BaseLink</b>	The BaseLink is where the kinematical chain begins. This must be the parent of the first joint. A mechanism may have only one BaseLink.

*Continues on next page*

## 9 Modeling tab

### 9.16 Create Mechanism

*Continued*

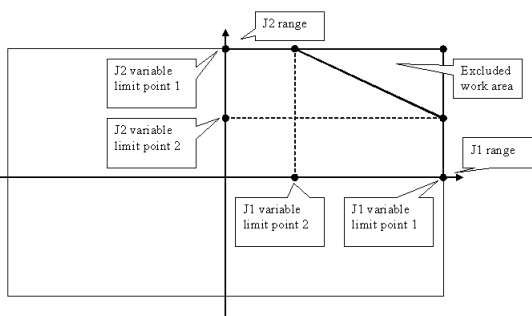

 xx060000	Adds a selected part to the Part list box.
<b>Remove Part</b>	Click this button to remove a selected part from the Part list box.
<b>Selected Part</b>	This group manipulates the transform of the selected part.
<b>Part Position</b>	Specify the position of the part.
<b>Part Orientation</b>	Specify the orientation of the part.
<b>Apply to Part</b>	Click this button to apply the settings to the part.

#### The Create/Modify Joint dialog box

A joint is the axis where two links move in relation to each other, rotationally or prismatically. Selecting a joint node will display a yellow-green line in the graphics window.

<b>Joint Name</b>	Specifies the name of the joint.
<b>Joint Type.</b>	Specifies the joint type. The default option is <b>Rotational</b> . Changing the Joint Type clears the Joint Limits below.
<b>Parent Link</b>	Specifies the parent link, usually the first joint of the mechanism.
<b>Child Link</b>	Specifies the child link. To be valid, the parent and child links may not be identical to each other, and the pair must be unique.
<b>Active</b>	Select this check box to make the joint active. An active joint is one that a user can move, while an inactive joint is a slave to an active joint.
<b>Joint Axis</b>	This group specifies the axis around or along which the child moves.
<b>First Position</b>	Specify the start point of the axis vector.
<b>Second Position</b>	Specify the end point of the axis vector.
<b>Jog Axis</b>	Demonstrates how the child link moves along its axis.
<b>Limit Type</b>	Specifies the limits in each direction to which a joint may move. The options are <b>Constant</b> , <b>Variable</b> and <b>No</b> .
<b>Joint Limits</b>	This group is visible in <b>Constant</b> or <b>Variable</b> mode.
<b>Min Limit</b>	Specifies the minimum joint limit.
<b>Max Limit</b>	Specifies the maximum joint limit.
<b>Joint Limits</b>	This group is visible in <b>Constant</b> or <b>Variable</b> mode.

*Continues on next page*

<b>Variable Limits</b>	<p>In <b>Variable</b> mode, variable limit points may be added as an advanced way of delimiting the area of movement.</p>  <p>xx060012</p>
 left-click	Adds a selected point to the Point list box.
<b>Remove</b>	Click this button to remove a selected point from the Point list box.

#### The Modify Frame/Tool Data dialog box

A frame/tool data node determines the link and location of a frame.

<b>Frame/Tool Data name</b>	Specifies the name of the frame or tool data.
<b>Belongs to Link</b>	Specifies the link to which the frame or tool belongs.
<b>Position</b>	Specify the position of the transform.
<b>Orientation</b>	Specify the orientation of the transform.
<b>Select values from target/Frame</b>	Select this box to select the values from a target or frame, which is selected in the box beneath the check box.
<b>Tool Data</b>	This group is visible if the mechanism is a tool.
<b>Mass</b>	Specifies the mass of the tool.
<b>Center of Gravity</b>	Specify the center of gravity of the tool.
<b>Moment of Inertia <math>I_x, I_y, I_z</math></b>	Specify the moment of inertia of the tool.

#### The Create Calibration dialog box

A calibration contains transforms for calibrating the joints. Two calibrations cannot share the same joint.

<b>Calibration belongs to Joint</b>	Specifies the joint to be calibrated.
<b>Position</b>	Specify the position of the transform.
<b>Orientation</b>	Specify the orientation of the transform.

*Continues on next page*

## 9 Modeling tab

### 9.16 Create Mechanism

*Continued*

#### The Create Dependency dialog box

A dependency is a relationship between two joints, by either a factor or a complex formula.

<b>Joint</b>	Specifies the joint whose motion will be controlled by other joints.
<b>Use LeadJoint and factor</b>	Select this option to specify a lead joint and factor.
<b>LeadJoint</b>	Specifies the lead joint.
<b>Factor</b>	This list holds a double which denotes the extent to which the lead joint will control the main joint.
<b>Use Formula</b>	Select this option to enter a formula in the box.

#### The Modify Mechanism dialog box

The **Modify Mechanism** dialog box contains the objects found in the **Create mechanism** dialog box, as well as the following:

<b>Joint Mapping</b>	These boxes handle the joint mapping of the mechanism. When editing, the mechanism must be disconnected from its library. The values must be integers from 1 – 6 in ascending order.
<b>Set</b>	Click this button to set the joint mapping.
<b>Poses</b>	Displays the poses and their joint values. Selecting a pose will move the mechanism to it in the graphics window.
<b>Add</b>	Click this button to bring up the <b>Create Pose</b> dialog box for adding a pose.
<b>Edit</b>	Click this button to bring up the <b>Modify Pose</b> dialog box for editing a selected pose. A SyncPose cannot be edited unless the mechanism is disconnected from its library.
<b>Remove</b>	Click this button to remove the selected pose. A single SyncPose cannot be removed.
<b>Set Transition Times</b>	Click this button to edit the transition times.

#### The Create/Modify Pose dialog box

<b>Pose Name</b>	Specifies the name of the pose. If the pose is a SyncPose, this box is not editable. The names "HomePosition" and "SyncPosition" are disallowed.
<b>Home Pose</b>	Select this box to specify the home pose of the mechanism. If selected, the non-editable pose name will be "HomePose".
<b>Launch Joint Jog Tool</b>	Click this button to bring up the joint jog tool.
<b>Use Current</b>	Click this button to set the current joint values in the <b>Joint Values</b> group.
<b>Reset Values</b>	Click this button to reset the joint values in the <b>Joint Values</b> group to what they were when the dialog box was opened.
<b>Joint Values</b>	Specify the joint values of the pose.

*Continues on next page*

---

**The Set Transition Times dialog box**

The **Set Transition Times** dialog box is designed like a distance table in a road atlas. The default values are zero.

<b>From Pose</b>	Specifies the start of the transition for the named pose.
<b>To Pose</b>	Specifies the end of the transition for the named pose.

## 9.17 Create Tool

### Creating a tool

You can create a robot hold tool by using the **Create Tool Wizard**. The wizard allows you to easily create a tool from an existing part or by using a dummy part to represent a tool. To create a tool complete with tooldata, follow these steps:

- 1 Click **Create Tool**.
- 2 In the **Tool Name** box, enter a tool name and choose one of the following options:

Option	Action
Use Existing	Select one of the existing parts from the list. The selected part will represent the tool graphics. The selected part must be a single part. Parts with attachments cannot be selected.
Use Dummy	A cone will be created to represent the tool.

- 3 Continue entering the **Mass** of the tool, the **Center of Gravity** and the **Moment of Inertia**  $I_x$ ,  $I_y$ ,  $I_z$ , if these values are known.



#### Note

If you do not know the correct values, the tool can still be used for programming motions, but this data must be corrected before running the program on real robots or measuring cycle times.



#### Tip

If the tool is built from materials with a similar density, you can find the center of gravity by clicking the tool model using the **Center of gravity** snap mode.

- 4 Click **Next**.
- 5 In the **TCP Name** box, enter a name for the Tool Center Point (TCP).



#### Note

The default name is the same as the name of the tool. If creating several TCPs for one tool, each TCP must have a unique name.

- 6 Enter the position of the TCP relative to the world coordinate system, which represents the tool mounting point, by any of the methods below:

Method	Description
Read values from existing target or frame	Click in the <b>Values from Target/Frame</b> box, then select the frame either in the graphics window or the <b>Paths&amp;Targets</b> browser.

*Continues on next page*

Method	Description
Enter position and orientation manually.	In the <b>Position</b> and <b>Orientation</b> boxes, type the values. If <b>Use Dummy Part</b> is selected, the position value can not be 0,0,0. At least one coordinate has to be > 0 in order for a cone to be created.

- 7 Click the arrow right button to transfer the values to the **TCP(s):** box.

If the tool shall have several TCPs, repeat steps 5 to 7 for each TCP.

- 8 Click **Done**.

The tool is created and appears in the **Layout** browser and in the graphics window.

### Creating tooldata for an existing geometry

Ensure to select the robot in which tooldata is created. To create tooldata for an existing geometry, follow these steps:

- 1 Click **Create Tool** and select **Use Existing** and the imported tool from the list.
- 2 Enter the requested data in the boxes in the **Create Tool Wizard**.
- 3 Attach the tool by dragging it to the robot.

### What to do next

To make the tool ready to use, do one of the following:

- To make the robot hold the tool, attach the tool to the robot.
- In the graphics window, check the position and orientation of the TCP. If it is incorrect, modify the values in the tool frame part of the tooldata.
- To simplify future usage of the created tool, save it as a library. On the **File** menu, click **Save As Library**. Browse to the folder where you want to store the tool component, enter a name for the tool component and click **Save**.

**This page is intentionally left blank**

## **10 Simulation tab**

### **10.1 Overview**

---

#### **The Simulation tab**

The Simulation tab contains the controls for setting up, configuring, controlling, monitoring and recording simulations.

## 10.2 Create Collision Set

---

### Overview

A collision set contains two groups, *Objects A* and *Objects B*, in which you place the objects to detect any collisions between them. When any object in *Objects A* collides with any object in *Objects B*, the collision is displayed in the graphical view and logged in the output window. You can have several collision sets in the station, but each collision set can only contain two groups.

### Creating a collision set

- 1 Click **Create Collision Set** to create a collision set in the **Layout** browser.
- 2 Expand the collision set and then drag one of the objects to the **ObjectsA** node to check for collisions.

If you have several objects you want to check for collisions with objects in the **ObjectsB** node, for example, the tool and the robot, drag all of them to the **ObjectsA** node.

- 3 Drag the objects to the **ObjectsB** node to check for collisions.

If you have several objects you want to check for collisions with objects in the **ObjectsA** node, for example, the work piece and the fixture, drag all of them to the **ObjectsB** node.



#### Tip

Selecting a collision set or one of its groups (*Objects A* or *Objects B*) highlights the corresponding objects in the graphical window and the browser. Use this feature to quickly check what objects have been added to a collision set or to one of its groups.

## 10.3 Simulation Setup

### Overview

The Simulation Setup dialog box is used to perform the following two main tasks.

- Setting up the sequence and entry point in the robot program
- Creating simulation scenarios for different simulated objects

### Prerequisites

To set up a simulation, the following conditions must be met:

- At least one path must have been created in the station.
- The paths to be simulated must have been synchronized to the virtual controller.

### The Setup Simulation dialog box

The Setup Simulation dialog box consists of the following two tabs,

- Program Sequence
- Simulation Scenarios

### Program Sequence

From this tab, you can perform the combined task of configuring the program sequence and program execution such as entry point, and running the execution mode.

The Program Sequence tab consists of the following:

<b>Select Active Tasks</b>	Displays all running IRC5 controllers in the station along with the tasks.
<b>Execution Sequence</b> <b>&lt;Task Name&gt;</b>	Displays the procedures in the main entry routine of the task. The sequence of the procedures shows the sequence of execution.
<b>&lt;-</b>	Click the arrow left button to transfer the selected procedure to the <b>Main Sequence</b> box. The procedure will be added last to the sequence.
<b>X</b>	Click this button to remove the selected procedures or sequences from the <b>Main sequence</b> box.
<b>arrow up</b>	Click the arrow up button to move the sequence up in the list in the <b>Main Sequence</b> box or in the <b>Available Procedures</b> box.
<b>arrow down</b>	Click the arrow down button to move the sequence down in the list in the <b>Main Sequence</b> box or in the <b>Available Procedures</b> box.
<b>Available Procedures</b>	Displays all procedures available in the controller. These procedures can be added to the execution sequence.

*Continues on next page*

## 10 Simulation tab

### 10.3 Simulation Setup

*Continued*

<b>Entry point</b>	<p>The task starts its execution in the routine specified by the Entry point. You can setup several simulations at the same time.</p> <ol style="list-style-type: none"><li>1 Click <b>Entry point</b>, the <b>Select entry point</b> dialog box appears.</li><li>2 Click <b>Select entry point</b> for drop down to select the routine to be used as entry point . By default, the value is set to <b>main</b>.</li><li>3 Click <b>Select module</b> drop down to select the module in the task.By default, the value is set to <b>Module1</b>.</li><li>4 Click <b>OK</b>.</li></ol>
<b>Run mode</b>	<p>You can change the run mode between continuous and single cycle mode by toggling the radio buttons.</p> <ul style="list-style-type: none"><li>• <b>Continuous</b> : In this mode, the main routine is executed over and over again until you stop the program.</li><li>• <b>Single</b> : In this mode, the main routine is executed once and thereafter the program is stopped.</li></ul>

#### Simulation Scenarios

From this tab, you can create simulation scenarios containing different simulated objects and connect each scenario with a predefined state to ensure that the correct state is applied to all project objects before running the scenario.

If you want to simulate a specific part or segment of the cell where not all simulated objects in the cell are included, you can set up a new scenario and add only the objects needed for simulation.

The Simulation Scenarios tab consists of the following:

<b>Scenarios</b>	<p>Lists all station scenarios. By default, atleast one scenario is created when you create a station.</p> <p>Select the check box to make the scenario active. Active scenarios cannot be deleted and there should always be one active scenario.</p> <ul style="list-style-type: none"><li>• <b>Add</b> : Click <b>Add</b> to create a new scenario.</li><li>• <b>Remove</b> : Click <b>Remove</b> to delete the selected scenario.</li></ul> <p>Click the scenario in the list view to rename it.</p>
<b>Simulated object</b>	<p>Displays all objects that can be part of a simulation.</p> <p>Objects that utilize simulation time can be part of a simulation. For example, Virtual Controllers and Smart Components.</p> <p>When you create a new scenario, all objects are selected by default.</p>
<b>Saved State</b>	<p>When you set the scenario to active and start the simulation, you can connect a saved state for each scenario and restore this state to all objects that are part of the state.</p> <p>The <b>Saved State</b> drop-down contains all saved states in the station, as well as the entries with no state. By default, no state is connected to the scenario.</p> <p>For more on Saved states, see <a href="#">Saved States on page 267</a>.</p>

#### Setting up a simulation

- 1 Click **Simulation Setup** to bring up the **Setup Simulation** dialog box.
- 2 From the **Program Sequence** tab, select the tasks to be active during simulation in the **Select Active Tasks** box.
- 3 Select the run mode as either **Continuous** or **Single Cycle**.

*Continues on next page*

- 4 In the **Main Sequence** box, select the execution sequence of the procedure in the tasks main entry routine.
- 5 From the **Available Procedures** list, transfer the procedures to be active in the simulation to the **Main Sequence** box by selecting them and clicking the left arrow button between the lists. (This creates a procedure call in the main procedure).
- 6 To start the simulation from procedure other than the actual **Main** procedure, click **Entry point** and set the entry point and module. By default, entry point is set to **Main** and module as **Module1**.
- 7 Click **Apply** to set the simulation. If you click **OK**, the simulation will be set and the dialog box will be closed.

---

**Creating simulation scenarios**

- 1 Click **Simulation Setup** to bring up the **Setup Simulation** dialog box.
- 2 From the **Simulation Scenarios** tab,
  - Click **Add** to create a new scenario in the **Scenarios** box.
  - Click **Remove** to delete the selected scenario from the **Scenarios** box.

When you create a new scenario, by default, all objects are selected in the **Scenarios** box.
- 3 Select a saved state for the scenario from the **Saved State** drop down.

## 10 Simulation tab

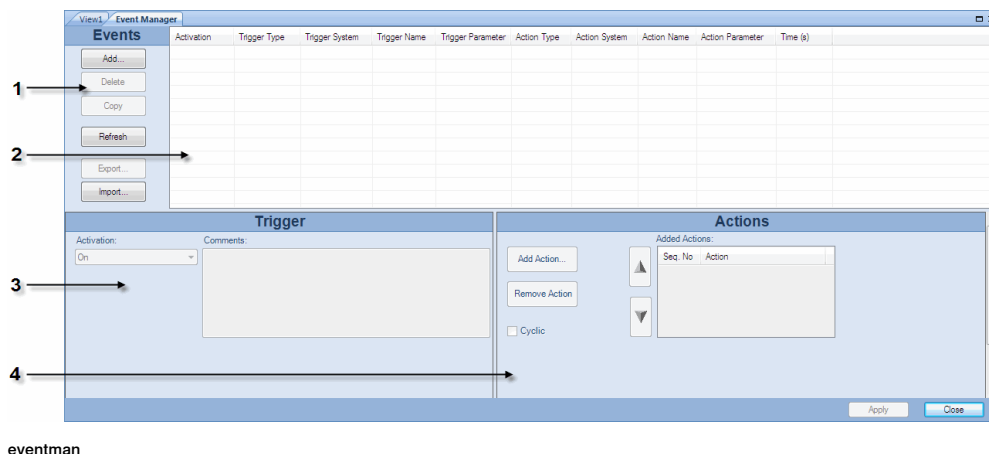
### 10.4 Event Manager

#### 10.4 Event Manager

##### Creating an event

- 1 Click **Event Manager**.
- 2 Click **Add** to open the New Event Wizard.
- 3 Complete the New Event wizard to create the event.

##### Event manager main parts



Part	Description
1	<b>The Task pane.</b> Here you create new events, or copy or delete existing events selected in the Event grid.
2	<b>The Event grid.</b> Displays all events in the station. Here you select events to edit, copy or delete.
3	<b>The Trigger editor.</b> Here you edit the properties of the events trigger. The upper part of the trigger editor is the same for all triggers, and the lower part adapts to the selected trigger type.
4	<b>The Action editor.</b> Here you edit the properties of the events action. The upper part of the action editor is the same for all actions, and the lower part adapts to the selected action type.

##### The task pane parts

Part	Description
<b>Add</b>	Starts the <b>Create New Event Wizard</b> .
<b>Delete</b>	Deletes the event selected in the Event grid.
<b>Copy</b>	Copies the event selected in the Event grid.
<b>Refresh</b>	Refreshes the Event manager.
<b>Export</b>	
<b>Import</b>	

*Continues on next page*

### The event grid columns

In the event grid, each row is a an event and the columns in the grid display their properties.:

Column	Description
Activation	Displays whether the event is active or not. <b>On</b> = The action is always carried out when the trigger event occurs. <b>Off</b> = The action is not carried out when the trigger event occurs. <b>Simulation</b> = The action is only carried out if the trigger event occurs when running a simulation.
Trigger Type	Displays the type of condition that triggers the action. <b>I/O signals changed</b> = Changes a digital I/O signal. <b>I/O Connection</b> = Simulates the behavior of a Programmable Logic Controller (PLC). <b>Collision</b> = Starts or ends a collision or near-miss between objects in a collision set. <b>Simulation time</b> = Sets the activation time. <i>Note:</i> The Simulation time button is enabled once the activation is set to Simulation. The trigger type cannot be changed in the trigger editor. If you want another trigger type than the current one, create a completely new event.
Trigger System	When the trigger type is <i>I/O Signal Trigger</i> , this column displays to which system the signal used as trigger belongs. A dash (-) signifies a virtual signal.
Trigger Name	The name of the signal or collision set used as trigger.
Trigger Parameter	Displays the condition of the event under which triggering occurs. <b>0</b> = The I/O signal used as trigger switches to false. <b>1</b> = The I/O signal used as trigger switches to true. <b>Started</b> = A collision starts within the collision set used as trigger. <b>Ended</b> = A collision ends within the collision set used as trigger. <b>Near miss started</b> = A near-miss starts within the collision set used as trigger. <b>Near miss ended</b> = A near-miss ends within the collision set used as trigger.

*Continues on next page*

## 10 Simulation tab

### 10.4 Event Manager

*Continued*

Column	Description
Action Type	<p>Displays the action type that occurs in conjunction with the trigger</p> <p><b>I/O Signal Action</b> = Changes the value of a digital input or output signal.</p> <p><b>Attach Object</b> = Attaches an object to another.</p> <p><b>Detach Object</b> = Detaches an object from another.</p> <p><b>Turn On/Off Simulation Monitor</b> = Toggles the simulation monitor of a specific mechanism.</p> <p><b>Turn On/Off Timer</b> = Toggles the process timer.</p> <p><b>Move Mechanism to Pose</b> = Moves the selected mechanism to a predefined pose and thereafter sends a station signal. Activates or deactivates the process timer.</p> <p><b>Move Graphical Object</b> = Moves a graphical object to a new position and orientation.</p> <p><b>Show/Hide Graphical Object</b> = Shows or hides the graphical object.</p> <p><b>Do Nothing</b> = No action occurs.</p> <p><b>Multiple</b> = The event triggers multiple actions, either all at once or one at a time, each time the trigger is activated. Each action can be viewed in the action editor.</p>
Action System	<p>When the action type is <i>Change I/O</i>, this column displays the system to which the signal to change belongs.</p> <p>A dash (-) signifies a virtual signal.</p>
Action Name	<p>Displays the name of the signal to change, when the action type is <i>Change I/O</i>.</p>
Action Parameter	<p>Displays the condition after the action has occurred.</p> <p><b>0</b> = The I/O signal will be set to false.</p> <p><b>1</b> = The I/O signal will be set to true.</p> <p><b>On</b> = Turns the process timer on.</p> <p><b>Off</b> = Turns the process timer off.</p> <p><b>Object1 -&gt; Object2</b> = Displays the object to which another will be attached when the action type is Attach object.</p> <p><b>Object1 &lt;- Object2</b> = Displays the object from which another will be detached when the action type is Detach object.</p> <p><b>Ended</b> = A collision ends within the collision set used as trigger.</p> <p><b>Near miss started</b> = A near-miss starts within the collision set used as trigger.</p> <p><b>Near miss ended</b> = A near-miss ends within the collision set used as trigger.</p> <p><b>Multiple</b> = Signifies multiple actions.</p>
Time	<p>Displays the time when the event trigger was executed.</p>

*Continues on next page*

**The trigger editor parts**

In the trigger editor you set the properties of the trigger. The upper part of the editor is common for all types of triggers, and the lower part adapts to the trigger type at hand.

**Parts common to triggers**

Part	Description
<b>Activation</b>	Sets whether the event is active or not. <b>On</b> = The action is always carried out when the trigger event occurs. <b>Off</b> = The action is not carried out when the trigger event occurs. <b>Simulation</b> = The action is only carried out if the trigger event occurs when running a simulation.
<b>Comments</b>	Text box for comments and notes about the event.

**Parts specific to I/O signal triggers**

Part	Description
<b>Active Controller</b>	Select the system to which the I/O to use as a trigger belongs.
<b>Signals</b>	Displays all signals that can be used as triggers.
<b>Trigger Condition</b>	For digital signals, sets whether the event shall trigger when the signals are set as true or false. For analog signals, which are only available for station signals, the event shall trigger under any of the following conditions: <b>Greater than, Greater/Equal, Less than, Less/Equal, Equal to, Not equal to.</b>

**Parts specific to I/O connection triggers**

Part	Description
<b>Add</b>	Opens a dialog box for adding an activator signal to the Activator Signals pane.
<b>Remove</b>	Removes a selected activator signal.
<b>Add &gt;</b>	Opens a dialog box for adding an operator symbol to the Connections pane.
<b>Remove</b>	Removes a selected operator symbol.
<b>Delay (s)</b>	Specifies the delay in seconds.

**Parts specific to Collision triggers**

Part	Description
<b>Collision Type</b>	Set the kind of collision to use as trigger. <b>Started</b> = Triggers when a collision starts. <b>Ended</b> = Triggers when a collision ends. <b>Near miss started</b> = Triggers when a near-miss starts. <b>Near miss ended</b> = Triggers when a near-miss ends.
<b>Collision set</b>	Select the collision set to use as trigger.

*Continues on next page*

## 10 Simulation tab

### 10.4 Event Manager

*Continued*

#### The action editor parts

In the action editor you set the properties of the actions for the event. The upper part of the editor is common to all types of actions, and the lower part adjusts to the selected action.

#### Parts common to all actions

Part	Description
<b>Add Action</b>	Adds a new action that occurs when the triggering condition fulfills. You can add several different actions that either are performed at once or one at a time each time the event triggers. The following types of actions are available: <b>Change I/O</b> = Changes the value of a digital input or output signal. <b>Attach object</b> = Attaches an object to another. <b>Detach object</b> = Detaches and object from another. <b>Turn On/Off Timer</b> = Activates or deactivates the process timer. <b>Do Nothing</b> = No action occurs (might be useful for manipulating sequences of actions).
<b>Remove Action</b>	Removes the action selected in the Added Actions list.
<b>Cyclic</b>	When selected, the actions are performed one at a time each time the trigger occurs. When all actions in the list have been performed, the event will restart with the first action in the list. When cleared, all actions are performed at once every time the trigger occurs.
<b>Added Actions</b>	Lists all actions of the event, in the order they will be executed.
<b>Arrow</b>	Rearranges the order in which the actions are executed.

#### Parts specific to I/O Actions

Part	Description
<b>Active Controller</b>	Displays all systems of the station. Select the system to which the I/O to change belongs.
<b>Signals</b>	Displays all signals that can be set.
<b>Action</b>	Sets whether the event shall set the signals to true or false. If the action is connected to an <i>I/O Connection</i> , this group will not be available.

#### Parts specific to Attach actions

Part	Description
<b>Attach object</b>	Select an object in the station to attach.
<b>Attach to</b>	Select the object in the station to attach to.
<b>Update position / Keep position</b>	<b>Update position</b> = Moves the local origin of the attached object to the attachment point of the other object when making the attachment. For mechanisms, the attachment point is the TCP or the flange; for other objects, it is the local origin. <b>Keep position</b> = Keeps the current position of the object to attach when making the attachment.
<b>Flange index</b>	If the mechanism you attach the object to has several flanges (attachments points), select the one to use.

*Continues on next page*

Part	Description
Offset Position	Optionally, specify an offset between the objects when making the attachment
Offset Orientation	Optionally, specify an offset between the objects when making the attachment

## Parts specific to Detach actions

Part	Description
Detach object	Select an object in the station to detach.
Detach from	Select the object in the station to detach from.

## Parts specific to Turn On/Off Simulation Monitor actions

Part	Description
Mechanism	Selects the mechanism.
Turn Simulation Monitor On/Off	Sets whether the action shall start or stop the simulation monitor function.

## Parts specific to Turn On/Off Timer actions

Part	Description
Turn On/Off Timer	Sets whether the action shall start or stop the process timer.

## Parts specific to Move Mechanism to Pose actions

Part	Description
Mechanism	Selects the mechanism.
Pose	Selects between <b>SyncPose</b> and <b>HomePose</b> .
Station signal to set when Pose reached	Lists the station signals that are sent after the mechanism reaches its pose.
Add Digital	Click this button to add a digital signal to the grid.
Remove	Click this button to remove a digital signal from the grid.

## Parts specific to Move Graphical Object actions

Part	Description
Graphical Object to Move	Select a graphical object in the station to move.
New Position	Sets the new position of the object.
New Orientation	Sets the new orientation of the object.

## Parts specific to Show/Hide Graphical Object actions

Part	Description
Graphical Object	Select a graphical object in the station.
Show/Hide	Sets whether the object is shown or hidden.

## 10.5 Station Logic

### Introduction to Station Logic

The Station Logic has some of the characteristics of a Smart Component. It can be used to work with these characteristics on the station level.

The Station Logic editor consists of the following tabs similar to that of a Smart Component editor:

- Compose
- Properties and Bindings
- Signals and Connections
- View

For more information on the characteristics of a Smart Component editor, see [Smart Component on page 264](#).

### Opening Station Logic

You can launch Station Logic in any of the following two ways:

- In the **Simulation** tab, click **Reset** and select **Manage States**.
- In the **Layout** browser, right-click the station and select **Station Logic**.

### Differences between Station Logic and Smart Component

The following table lists some of the differences while working with Station Logic and a Smart Component:

Smart Component	Station Logic
The Editor window consists of a text box displaying the description of the component that is used for modifying the text.	The Editor window do not have the description text box wherein the description can be modified.
The <b>Compose</b> tab has the following options: <ul style="list-style-type: none"><li>• Child components</li><li>• Saved States</li><li>• Assets</li></ul>	The <b>Compose</b> tab has the following options: <ul style="list-style-type: none"><li>• Child components</li><li>• Saved States</li></ul>
The <b>Properties and Bindings</b> tab has the following options: <ul style="list-style-type: none"><li>• Dynamic Properties</li><li>• Property Bindings</li></ul>	The <b>Properties and Bindings</b> tab has the following options: <ul style="list-style-type: none"><li>• Property Bindings</li></ul>
In the <b>Signals and Connections</b> tab, when working with Add or Edit I/O Connections, you do not have the option of selecting the VCs in the station from the the <b>Source Object</b> and <b>Target Object</b> list.	You can create connections to and from I/O signals in a VC. In the <b>Signals and Connections</b> tab, when working with Add or Edit I/O Connections, you have the option of selecting the VCs in the station from the the <b>Source Object</b> and <b>Target Object</b> list.

## 10.6 Activate Mechanical Units

---

### To activate or deactivate mechanical units manually

- 1 Click **Activate Mechanical Units** to bring up a dialog box.
- 2 In the **Activate Mechanical Units** dialog box, select the check boxes for the mechanical units to set as active. When activating a mechanical unit that shares a common drive unit, the other mechanical unit sharing that drive unit will be deactivated automatically.

## 10.7 Simulation Control

### Running a simulation

- 1 In the **Simulation Control** group,

Click...	to...
<b>Play/Resume</b>	start and resume the simulation. <ul style="list-style-type: none"><li>• The <b>Pause</b> button is enabled once you start the simulation</li><li>• The <b>Play</b> button is changed to <b>Resume</b> once you pause the simulation.</li><li>• Click <b>Resume</b> to resume the simulation.</li></ul>
<b>Play and select Record to Viewer</b>	start the simulation and to record it to a Station Viewer. The <b>Save As</b> dialog box appears where the simulation is saved.
<b>Pause/Step</b>	pause and step the simulation. <ul style="list-style-type: none"><li>• The <b>Pause</b> button is changed to <b>Step</b> once you start the simulation.</li><li>• Click <b>Step</b> to run the simulation in steps.</li></ul> You can set the simulation timestep. See <a href="#">Options:Simulation:Accuracy on page 201</a> .
<b>Reset</b>	reset the simulation to its initial state. See <a href="#">Resetting simulation on page 340</a> .



#### Note

The **Record to Viewer** option is a special recording mode that allow simulations created using Smart Components to be viewed in the Station Viewer. Record to viewer is disabled when Free Run mode is enabled.

### Resetting simulation

- 1 In the **Simulation Control** group, click **Reset** to reset the simulation.
- 2 Click **Reset** and select **Save Current state** to store states of objects and VCs to be used in a simulation scenario. For more information, see [Save Current State on page 267](#).
- 3 Click **Reset** and select **Manage states** to launch Station Logic. For more information, see [Station Logic on page 338](#).

## 10.8 I/O Simulator

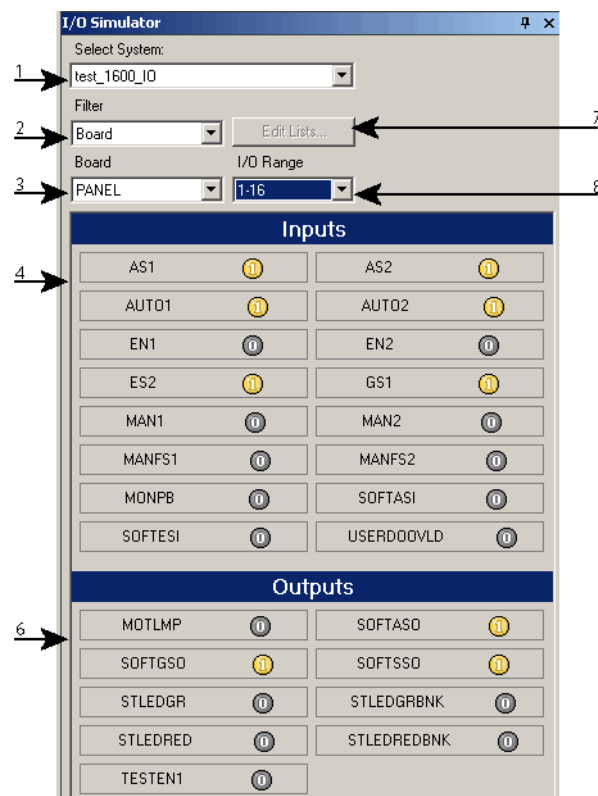
### Setting I/O signals using the I/O Simulator

- 1 Click **I/O Simulator**. This opens the I/O simulator.
- 2 If the station contains several systems, select the appropriate one in the **Select System** list.
- 3 In the **Filter** list and **I/O Range** list, make selections that display the signals to set. Depending on the filter used, you might also set a filter specification.
- 4 To change the value of a digital I/O signal, click it.  
To change the value of an analog signal, type the new value in the value box.

### The I/O Simulator window

With RobotStudio's I/O simulator you view and manually set existing signals, groups and cross-connections during program execution, thus making it possible to simulate or manipulate the signals.

The I/O simulator displays the signals for one system at a time in groups of 16 signals. For handling large sets of signals, you can filter which signals to display and also create custom lists with favorite signals for quick access.



io\_overv

Part	Description
1	<b>Select System.</b> Select the system whose signals you want to view.
2	<b>Filter type.</b> Select the type of filter to use.

*Continues on next page*

## 10 Simulation tab

### 10.8 I/O Simulator






Continued

Part	Description
3	<b>Filter Specification.</b> Select the filter for limiting the signal display. For example, if Board is set as filter type, then you select the board whose signals you want to view.
4	<b>Inputs.</b> Displays all input signals that pass the applied filter. If more than 16 signals pass, only 16 signals at a time are displayed. Then use the I/O range list to select the signals to view.
5	<b>Outputs</b> Displays all output signals that pass the applied filter. If more than 16 signals pass, only 16 signals at a time are displayed. Then use the I/O range list to select the signals to view.
6	<b>Edit Lists.</b> Click this button to create or edit lists of favorite signals.
7	<b>I/O Range.</b> When more than 16 signals pass the filter, use this list to select the range of signals to display.

#### Types of signal filters

Filter	Description
Board	Displays all signals on a specific board. To select a board, use the <b>Filter Specification</b> list.
Group	Displays all signals that belong to a specific group. To select a group, use the <b>Filter Specification</b> list.
User List	Displays all signals in a favorite list. To select a list, use the <b>Filter Specification</b> list.
Digital Inputs	Displays all digital input signals of the system.
Digital Outputs	Displays all digital output signals of the system.
Analog Inputs	Displays all analog input signals of the system.
Analog Outputs	Displays all analog output signals of the system.

#### Signal icons

 value 1	Digital signal with value 1.
 value zero	Digital signal with value 0.
 cross connec	The cross in the upper right corner indicates that the signals are a cross-connection.
 inverted	The -1 in the upper right corner indicates that the signal is inverted.
 value box	Value box for groups or analog signals.

## 10.9 Monitor

### The TCP Trace tab

<b>Enable TCP Trace</b>	Select this check box to activate tracing of the TCP path for the selected robot.
<b>Trace length</b>	Specify the maximum length of the trace in millimeters.
<b>Trace Color</b>	Displays the color of the trace when no alerts are activated. To change the color of the trace, click the colored box.
<b>Alert color</b>	Displays the color of the trace when any of the alerts defined on the <b>Alerts</b> tab exceeds a threshold value. To change the color of the trace, click the colored box.
<b>Clear Trace</b>	Click this button to remove the current trace from the graphics window.

### The Alerts tab

<b>Enable Simulation Alerts</b>	Select this check box to activate simulation alerts for the selected robot.
<b>Log Alerts to Output Window</b>	Select this check box to see a warning message when a threshold value is exceeded. If TCP trace is not enabled, this is the only display of the alert.
<b>TCP Speed</b>	Specify the threshold value for TCP speed alerts.
<b>TCP Acceleration</b>	Specify the threshold value for TCP acceleration alerts.
<b>Wrist Singularity</b>	Specify how close joint five can be to zero rotation before alerting.
<b>Joint Limits</b>	Specify how close each joint can be to its limits before alerting.

### 10.10 Stopwatch

---

#### Stopwatch for measuring process time

The Stopwatch feature is used for measuring the time taken between two trigger points in a process, and also for the process as a whole. The two trigger points are called the Start Trigger and the End Trigger.

When a stopwatch is setup, the timer starts when the Start Trigger occurs, and stops when the End Trigger occurs.

---

#### Setting up a Stopwatch

- 1 On the **Simulation** tab, in the **Monitor** group, click **Stopwatch**.  
The Stopwatch settings dialog appears.
- 2 Specify a **Name** for the stopwatch.
- 3 Select a **Start Trigger** and an **End Trigger** for the stopwatch.

The following parameters are listed for selection as triggers:

- Simulation Start
- Simulation Stop
- Target Changed

Additionally, specify the mechanical Unit and the target.

- I/O Value

Additionally, specify the source mechanical unit from where the signal comes, the type of I/O signal and the value of the signal.

- 4 Click **Add**.

## **10.11 Signal Analyzer**

### **10.11.1 Signal Analyzer for both real and virtual controllers**

---

The Signal Analyzer functionality helps in displaying and analyzing signals from a robot controller. Using the Signal Analyzer, you can optimize the robot program. The Signal Analyzer functionality is present for both virtual and real controllers. The version adapted for real controllers is called Signal Analyzer Online. The following section describes the Signal Analyzer functionality for virtual controllers, though it includes certain common features.

## 10 Simulation tab

---

### 10.11.2 Signal Setup

#### 10.11.2 Signal Setup

---

##### Overview

This feature allows you to configure the signals to be saved for the next simulation. The signals are recorded from the controller information stream and are stored in the station.

---

##### Layout of Signal Setup

The Signal Setup window displays all the signals available for recording. It also displays the signals selected for recording.

The Signal Setup window has the following options:

- Select Signals view
- Current Setup view
- Refresh

##### Select Signals view

Displays all the available source signals. By default, the source tree is expanded. In the source tree, you can select the check-box and add the signal to the Current Setup view.

The signals are organized in a hierarchical tree structure. You can expand or collapse the nodes (except the signal nodes which are at the lowest level) either from the context menu or double-click the node.

##### Current Setup view

Displays all the selected signals.

To remove a signal, Right-click the signal and select **Delete**.

##### Refresh

The Signal Setup window, by default, will be updated automatically, if a signal is added or removed. However, in some cases, a manual refresh may be needed.

In the **Signal Setup** window, click **Refresh** to ensure all signals are displayed in the window.

---

##### Available signals

The following tables show the signals that are available for setting up. You can subscribe to a maximum of 12 signals simultaneously.

Category	Available signals
Controller Signals	Total Motor Power. See the description provided after this table.
	Total Power Consumption. See the description provided after this table.
EventLog	All domains
I/O System	All signals
Joint	J1-J6
	Near Limit. See the description provided after this table.

*Continues on next page*

Category	Available signals
Target	Fine Point
	Target Changed , Tool Changed, Workobject Changed
TCP	Maximum Linear Acceleration in World
	Orientation Q1-Q4 Current Workobject
	Orientation Speed in Current Workobject
	Pos X, Y, Z in Current Workobject
	Robot Configuration cf1, cf4, cf6, cfx
	Speed in Current Workobject
	Zone Entered, Zone Left
TCP	
Smart Components	All signals

**Total Motor Power**

The signal Total Motor Power shows the sum of the instantaneous power for each joint. It may be positive or negative.

The instantaneous power for a specific joint is positive when it accelerates and negative when it decelerates. If one joint is accelerating at the same time as another is decelerating, then the negative energy from the decelerating joint is reused for the accelerating joint. If the sum of the instantaneous power of all joints is negative then the power surplus cannot be reused but is burned off in the bleeder.

**Total Power Consumption**

The signal Total Power Consumption is the integral of the positive part of the Total Motor Power, plus the estimated power consumption of the controller cabinet.

**Near Limit**

Near Limit checks the distance to the closest limit for each joint. If any joint is less than 20 degrees from a limit, the Near Limit signal will show the current value. Otherwise, the value of the signal will be constant at 20 degrees. If more than one joint is below 20 degrees from a limit, then the one that is closest will be looked at.

**Setting up the signals**

Use this procedure to configure the signals to be saved for the next simulation:

- 1 Load a station with system. See [New on page 190](#) .
- 2 In the **Simulation** tab, click **Signal Analyzer** and select **Signal Setup**.  
The Signal Setup window appears.
- 3 In the **Select Signals** view, select the signals to be configured and stored for simulation.  
The selected signals are added in the Current Setup window.
- 4 In the **Current Setup** view, right-click **Station Database** and select **Enabled**.

*Continues on next page*

This ensures that all selected signals will be recorded whenever a simulation is running.



#### Note

- Disabling the station database will stop the recording but stores the configuration and all completed recordings in the station.



#### Note

- You can analyze the recorded signals. See [Layout and usage on page 349](#).
- You can organize the saved signal data. See [History on page 352](#).
- Disable signal recording as soon as the analysis is completed to avoid the station file size to increase.



#### Note

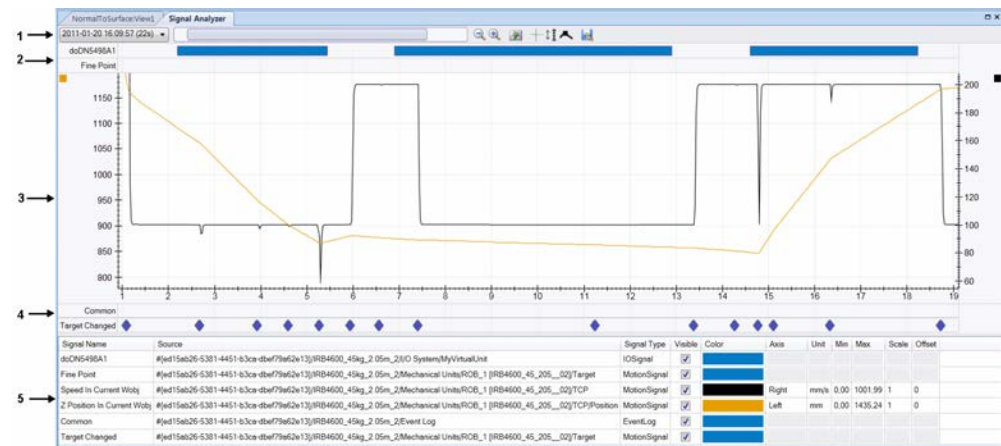
It is not possible to subscribe to signals connected to a unit of type LOCAL\_GENERIC. Attempting to do so produces this error message in the Output window:

Failed to subscribe on signal: ...

### 10.11.3 Layout and usage

#### Layout of Signal Analyzer

The following figure shows the layout of the Signal Analyzer



en1100000034

1	Toolbar	Displays a toolbar with options to configure and to work with the Signal Analyzer.
2	Digital signal values	Displays a colored bar representing a segment where the signal is set.
3	Analog signal values	Displays analog and numeric signal values.
4	Events	Displays discrete events, such as EventLog messages.
5	Signals table	Displays information about all recorded signals for the current data session.

#### Toolbar

The toolbar displays the following options:

Option	Description
Drop-down list	For selecting the signal recording to be displayed. These signals are also available in Signal History. See <a href="#">History on page 352</a> .
Timer slider	For moving the time forward and backward.
Zoom in/out buttons	For zooming in / out the time axis.
Live data button	For enabling data to be shown live, that is, as recorded during a simulation.
Crosshair	For displaying the crosshairs following the mouse.
Auto scale button	For enabling / disabling vertical axis autoscaling.
Line marker button	For displaying the line markers for each sample in the analog / numeric graph.
Save button	For exporting the data to a file. The data can be saved in <i>Microsoft Excel 2007</i> format and <i>tab delimited text</i> format.

*Continues on next page*

## 10 Simulation tab

---

### 10.11.3 Layout and usage

*Continued*

#### Digital signal values

It displays one row per digital signal, showing the signal state over time. A solid colored bar indicates the signal is set (value=1), otherwise the signal is cleared (value=0). The signal name is displayed to the left.

Move the mouse over the colored bars to view additional information like the time stamps when the signal was set and reset.

#### Analog signal values

It displays a 2D line graph for each analog signal. It consists of the following:

- Left-side vertical axis
- Horizontal axis displaying time in seconds
- Plot area displaying the signal graphs
- Optional right-side vertical axis.

You can configure the individual signals to use the right-side vertical axis scale from the signals table at the bottom of the window. This axis is hidden by default.

The following actions can be performed in this segment:

- Scale the vertical axes: If you select the autoscale button in the toolbar, then the vertical axes will automatically scale to ensure that the line graphs are visible. You can modify the vertical scale using the mouse if the cursor is over the axis value area. This automatically deselects the autoscale button.
- Pan and Zoom time axis: If the cursor is over the central, main area of the plot, then you can scale, pan, and zoom the time axis using the mouse.

#### Events

It displays one row per selected event category. Each event is indicated with a diamond shaped icon. Click this icon to display a popup with more information about the event.

#### Signals table

Displays information about each recorded signal. This enables you to configure settings for each signal such as color, visibility, whether to use left or right vertical axis and so on.

---

### Using the Signal Analyzer

Use this procedure to analyze the recorded signal data:

- 1 Set up the signals to be analyzed. See [Setting up the signals on page 347](#).
- 2 Record signal data by running a simulation. See [Setting up the signals on page 347](#).
- 3 In the **Simulation** tab, click **Signal Analyzer**.

*Continues on next page*

The Signal Analyzer window appears.



### Note

- If the station does not contain any saved signal data, you will then have to setup the signals to analyze and record by running a simulation. See [Signal Setup on page 346](#).
- You can organize the saved signal data. See [History on page 352](#).

#### 10.11.4 History

---

##### Overview

This feature displays and helps in organizing saved signal recordings of the current RobotStudio station.

---

##### Layout of Signal History

You can do the following from the *Signal History* window:

- Click the column header to sort the history in ascending or descending order.
  - Click the drop-down to group the history as *View By Today* or *View by Order*.
- 

##### Organizing the Signal History

Use this procedure to organize the signal history:

- 1 Create a saved signal data in the station. See [Setting up the signals on page 347](#).

- 2 In the **Simulation** tab, click **Signal Analyzer** and select **History**.

The Signal History window with all the stored signal history elements are displayed.



##### Note

The signal history elements in the Signal History window is updated automatically whenever the signals are setup and the simulation is started / stopped.

- 3 In the **Signal History** window, right-click a history element and select:

- **Analyze:** To open the Signal Analyzer window.
- **Export:** To save selected history elements to a file.
- **Delete:** To remove the selected signal recording permanently.
- **Rename:** To rename the signal recording.



##### Note

Disable signal recording as soon as the analysis is completed to avoid the station file size to increase.

## 10.12 Record Movie

---

### Prerequisites

For optimal results, first configure the options, see [Options:General:Screen Recorder on page 196](#).

---

### Recording the screen

- 1 In the **Record Movie** group, click **Record application** to capture the entire application window, or **Record graphics** to capture just the graphics window.
- 2 When you are done, click **Stop Recording**. A dialog box appears in which you may choose to save the recording or discard it.
- 3 Click **View Recording** to playback the latest capture.

---

### Recording the simulation

- 1 In the **Record Movie** group, click **Record Simulation** to record the next simulation to a video clip.
- 2 When you are done, click **Stop Recording**.  
The simulation is saved in a default location which is displayed in the output window.
- 3 Click **View Recording** to playback the recording.  
The recording of simulation starts when you click **Play** in the **Simulation** tab.



#### Note

**Record Simulation** gives better output quality than **Record application** or **Record graphics**.

## 10.13 Conveyor Tracking Mechanism

### 10.13.1 Conveyor Tracking

---

#### Overview

Conveyor tracking is the function where the robot follows a workobject mounted on a moving conveyor.

This section describes *how to create a conveyor, add and remove objects to and from the conveyor, create targets during tracking, and simulate conveyor.*

For more information, see the *Application manual - Conveyor tracking.*

---

#### Conveyor tracking mechanism

This procedure describes the workflow for making a conveyor tracking system work in RobotStudio.

- 1 Create a conveyor mechanism. See [Create Conveyor mechanism on page 317](#).
- 2 Setup the conveyor. See [Setting up a conveyor on page 207](#) and [Encoder Unit on page 411](#).

For information on setting up a conveyor tracking station with two robots working on the same conveyor, see [Conveyor tracking station with two robots on page 77](#).

- 3 Jog the conveyor as well as the robot and teach some targets. See [Mechanism Joint Jog on page 469](#).
- 4 Simulate the conveyor. See [Conveyor Simulation on page 355](#).
- 5 Remove objects from conveyor. See [Removing objects from conveyor on page 208](#).

## 10.13.2 Conveyor Simulation

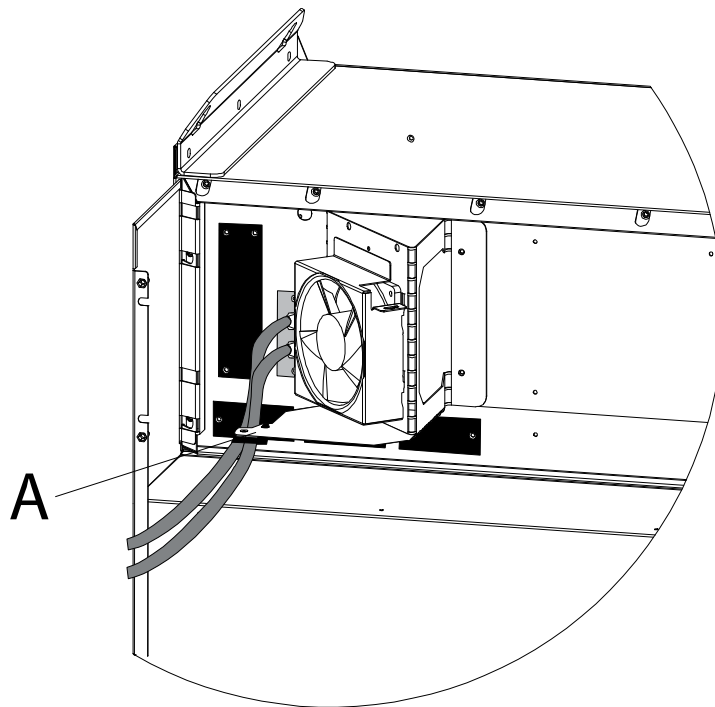
### Running a conveyor simulation

- 1 Create Action Instructions. See [Action Instruction on page 237](#).

Create the following five action instructions along with Move instructions:

ConfL\Off,ActUnit CNV1,WaitWObj Workobject\_1,  
DropWObjWorkobject\_1 and DeactUnit CNV1.

The following program is an example showing how the sequence of instructions appears:



action\_i



#### Note

If an error occurs while executing the program, the controller reaches Guard state. In this state, RobotStudio cannot execute the program during the next simulation. To recover from this state, open the **Control Panel** and switch to **Manual Mod** and then to **Auto Mode**.

For more information, see *Application manual - Conveyor Tracking*.

- 2 Synchronize to VC. See [Synchronize to VC on page 415](#).
- 3 Set up the Simulation. See [Simulation Setup on page 329](#).
- 4 Click **Simulation**.  
The Conveyor Simulation dialog appears.
- 5 In the **Conveyor Speed** box, set the speed during simulation.  
To move the conveyor in the backward direction, select the **Reverse** check box.

*Continues on next page*

## 10 Simulation tab

---

### 10.13.2 Conveyor Simulation

*Continued*

- 6 Click **Apply**.
- 7 Click **Play** to run the simulation.



#### Note

The conveyor speed and direction can be changed while running the simulation. To jump the conveyor back to the start position, click **Reset**. This button remains enabled as long as the station has at least one conveyor.

# 11 Controller tab

## 11.1 Real and virtual controllers

---

The Controller tab contains the controls for managing the real controller and also the controls for synchronization, configuration and tasks assigned to the virtual controller.

RobotStudio allows you to work with an off-line controller, which is a virtual IRC5 controller running locally on your PC. This offline controller is also referred to as the virtual controller (VC). RobotStudio also allows you to work with the real physical IRC5 controller, which is simply referred to as the real controller.

The features on the Controller tab can be categorized as follows:

- Features for both virtual and real controllers
- Features for real controllers
- Features for virtual controllers

For more information on working with a real controller, see [Working online on page 149](#).

## 11 Controller tab

---

### 11.2.1 Add Controller

## 11.2 Features for both virtual and real controllers

### 11.2.1 Add Controller

---

#### Adding and connecting to a controller

You can connect to real or virtual controller using the **Add Controller** button.

To connect to a real controller, on the **Controller** tab click the arrow next to the **Add Controller** icon, and then click one of the following commands as per your requirement:

- One Click Connect - For connecting to the service port of the controller
- Add Controller - For adding available controllers to the network



#### Note

For connecting RobotStudio to a real controller over the Ethernet (LAN), the controller system must have the RobotWare option **PC-interface**. This option is not required when connecting through the service port.

To start and connect to a virtual controller, on the **Controller** tab click the arrow next to the **Add Controller** icon, and then click **Start Virtual Controller**.

---

#### One Click Connect

The One Click Connect feature allows connecting to a robot controller, that is connected to the service port, in a single step. You need to do the following before using this feature:

- Connect the computer to the controller service port.
- Ensure that the network settings on the PC is correct. DHCP should either be enabled or the IP address should have a specific value. For more information on network settings, see [Network settings on page 153](#).

On the **Controller** tab, click the arrow next to the **Add Controller** icon, and then click **One Click Connect**.

---

#### Add Controller

- 1 In the **Controller** tab, click **Add Controller** to bring up a dialog box in which all available controllers are listed.
- 2 If the controller is not found in the list, type its IP address in the **IP Address** box, and then click **Refresh**.
- 3 Select the controller in the list and click **OK**.

*Continues on next page*

---

**Start Virtual Controller**

The **Start Virtual Controller** command allows you to start and stop a virtual controller, using a given system path and without needing a station.

**Tip**

You can use the **Start Virtual Controller** command when you require a virtual controller as emulator while developing PC SDK or RobotWare Additional Options. You can also use this command when you need to use the Configuration editor or the RAPID editor without requiring a station.

Clicking **Start Virtual Controller** under **Add Controller** opens the *Start Virtual Controller* dialog box. In this dialog box, specify the following:

- 1 In the **System Pool** drop-down list specify the location and folder of your PC where the required virtual controller system are stored.  
To add a folder to this list, click **Add** and then browse to and select the folder to be added. To remove a folder from the list, click **Remove**.
- 2 The *Systems Found* table lists the virtual controller systems found in the selected system folder. Click a system to select to select it for starting.
- 3 Select the required check boxes:
  - I-Start, to start the VC with the current system and the default settings
  - Local login
  - Handle Write Access automatically

### 11.2.2 Events

---

#### Events log

To view the events log of the controller, on the **Controller** tab in the **Controller Tools** group, click **Events**. This opens the *Event* log. The severity of each event is indicated by its background color; blue for information, yellow for warning and red for an error which needs to be corrected in order to proceed.

You can perform the following operations on the *Event* log.

- Click any event to view a brief description about the event.
- The **Auto Update** check box is selected by default, so that new events appear in the list as they occur.

Clearing the check box to disable automatic update. Selecting it again, fetches and displays the events missed while it was cleared.

- You can filter the event log list based on the category of the event or based on any text in its displayed details.

To filter the list based on any required text, specify it in the **Text** box.

To filter based on the events categories, use the **Category** drop-down list.

The list contains the following different event categories.

- Common (the default category, includes all categories)
  - Operational
  - System
  - Hardware
  - Program
  - Motion
  - IO & Communication
  - User
  - Internal
  - Process
  - Configuration
  - RAPID
- To clear the current event record, click **Clear**. This does not affect the event log of the controller, which can be retrieved again by clicking the **Get** button.
  - To retrieve and display all events currently stored in the controller, click **Get**
  - To save the event records of the selected event categories to log files on the computer, click **Save**.
  - To enable all events currently in the Common Event Log to be saved to a log file on the computer, select the **Log to file** check box.

The log file will be updated with all new events as they occur.

## 11.2.3 Inputs / Outputs

### I/O System

You can view and set input and output signals in the *I/O System* window. To open this window, on the **Controller** tab, in the **Controller Tools** group, click **Inputs/Outputs**.

The following details of I/O signals are available in the *I/O System* window:

- The **Name** column

This column shows the name of the signal. The name is set by the I/O unit's configuration and cannot be changed from the I/O system.

- The **Type** column

This column shows which type of signal it is, by using any of the abbreviations described below. The signal type is set by the I/O unit's configuration and cannot be changed from the I/O system.

Abbreviation	Description
DI	Digital input signal
DO	Digital output signal
AI	Analog input signal
AO	Analog output signal
GI	Group of signals, working as one input signal
GO	Group of signals, working as one output signal

- The **Value** column

This column shows the value of the signal. The value can be changed by double-clicking the signal row.

- The **Min Value** column

This column displays the minimum value that the signal can have.

- The **Max Value** column

This column displays the maximum value that the signal can have.

- The **Logical State** column

This column shows whether the signal is simulated or not. When a signal is simulated, you specify a value that overrides the actual signal. Changing the logical state by turning the simulation on or off can be done from the I/O system.

- The **Unit** column

This column shows to which I/O unit the signal belongs. This is set by the I/O unit's configuration and cannot be changed from the I/O system.

- The **Bus** column

This column shows to which I/O bus the signal belongs. This is set by the I/O bus' configuration and cannot be changed from the I/O system.

- The **Label** column

*Continues on next page*

## 11 Controller tab

---

### 11.2.3 Inputs / Outputs

*Continued*

This column displays the Signal Identification Label as defined in the I/O Configuration database.

You can filter the I/O system window to view only a subset of all signals. You can filter the view using the following parameters:

- **Name and Label**- Use the free-text edit boxes above these columns. The resulting view will show signals that contain the entered text string in the respective field.
- **Simulated** - Select this check box to view simulated signals only
- **Unit, Bus, and Category** - Use the drop-down list boxes above each column to select the required option for the respective parameter. The resulting view will show only those signals that have the selected option.
- **Clear filter** - Click this button to reset the view, and show all signals again.

## 11.2.4 ScreenMaker

### Overview

ScreenMaker is a tool in RobotStudio for creating customized FlexPendant user interfaces without the need to learn Visual Studio development environment and .NET programming.

For more information on ScreenMaker, see [ScreenMaker tab on page 499](#).

### Prerequisites



#### Note

In RobotStudio 5.61, ScreenMaker is only available for the 32-bit edition and is therefore disabled in the 64-bit edition.

To use ScreenMaker, you need to fulfill the following requirements:

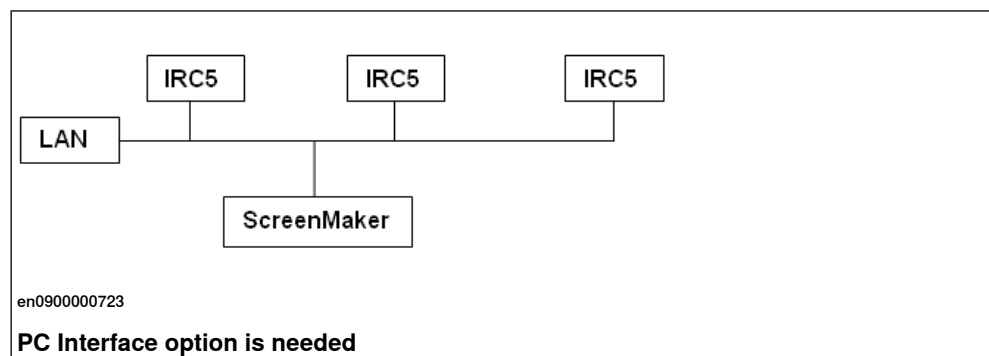
- RobotStudio with Premium license.
- The IRC5 controller(s) require the **FlexPendant Interface** option in RobotWare to run ScreenMaker applications.
- Install Microsoft .NET Compact Framework 2.0. Download it from <http://www.microsoft.com/download>
- Install FlexPendant SDK with the same version as the RobotWare on the target controller. Download it from <http://developercenter.robotstudio.com/>

For more information on the system requirements, hardware requirements, and the supported Operating Systems, see *RobotStudio Release Notes*.

### Testing on Virtual controller/Real controller

RobotWare FlexPendant Interface option is required for ScreenMaker applications.

**NOTE!** RobotWare PC Interface option is required only when using ScreenMaker for Robots on a LAN (to get the data from the controller, bind, and deploy). If there is no PC Interface option, service port can be used to design and deploy screens.



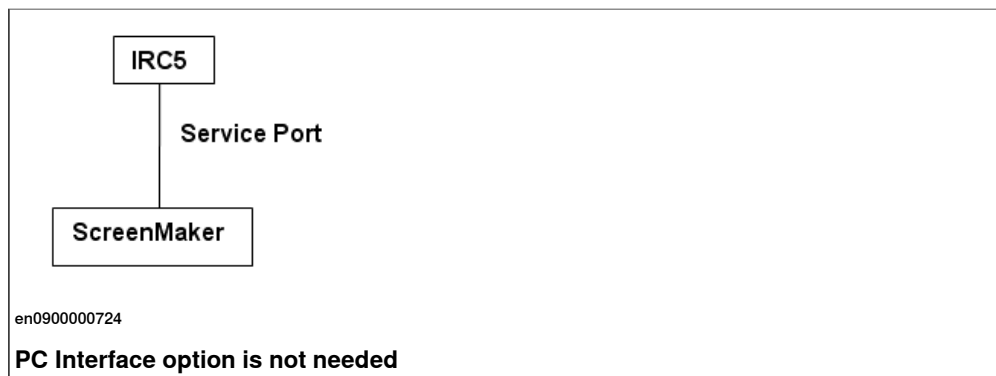
*Continues on next page*

## 11 Controller tab

---

### 11.2.4 ScreenMaker

*Continued*



---

### Launching ScreenMaker

You can launch ScreenMaker either from the **Controller** tab.

- 1 In the **Controller** tab, click the arrow next to the **FlexPendant** icon.
- 2 Click **ScreenMaker**.

ScreenMaker is launched as a new tab.

The connection to all connected virtual and real controllers can be established.



#### Note

For more information, see [Managing projects on page 507](#).

## 11.2.5 Restart a controller

### When to restart a controller

Some operations require a restart of the controller to take effect. When working in RobotStudio, you will be notified when a restart is necessary.

### Warm restart

Typically, you need to make an ordinary warm restart of a controller when:

- You have changed the baseframe of any of the robots belonging to that controller.
- You have changed the robot's configuration, either with the Configuration Editor or by loading new configuration files.
- You have added new options or hardware to the system.
- A system failure has occurred.

### Advanced restart options

The controller can be restarted with the following advanced restart options:

Option	Description
I-Start	Restarts the controller with the current system and the default settings. This restart discards the changes made to the robot's configuration. It reverts the current system to the state it had when it was installed on the controller (an empty system). This restart deletes all RAPID programs, data and custom configurations that have been added to the system.
P-Start	Restarts the controller with the current system and reinstall RAPID. This restart deletes all RAPID program modules. It can be useful if the system has changed in such a way that the programs no longer are valid, for instance if system parameters used by the program are changed.
X-Start	This restart applies only to real controllers. This restart saves the current system, with the current settings, and starts the boot application on the FlexPendant from which you can choose a new system to start with. You can also configure the controller's network settings from the boot application.
C-Start	This restart applies only to real controllers. This restart deletes the current system and starts the boot application on the FlexPendant from which you can choose a new system to start with. You can also configure the controller's network settings from the boot application.
B-Start	This restart applies only to real controllers. Restarts the controller with the current system and the last known good settings. This restart restores changes made to the robot's configuration to a previously good state.

### Restarting a virtual controller

- 1 In the **Controller browser**, select the controller to restart.

*Continues on next page*

## 11 Controller tab

---

### 11.2.5 Restart a controller

*Continued*

- 2 In the **Controller Tools** group, click the arrow next to the **Restart** icon, and then select one of the following options:

<b>Warmstart</b>	Restarts the VC and activates the changes made to the system. This is the default option if you directly click <b>Restart</b> .
<b>I-start</b>	Restarts the VC with the current system and the default settings.
<b>P-start</b>	Restarts the VC with the current system and reinstalls RAPID.

The options for **Restart** are also present in the context menu when you right-click a controller in the **Controller** browser.

---

### Restarting a real controller

The following are the prerequisites for restarting a real controller:

- You must have Write access to the controller you are restarting.
- For the advanced restart methods X-start and C-start, you must have access to the controller's FlexPendant.

To restart a real controller:

- 1 In the **Controller** browser, select the controller to restart.
- 2 In the **Controller Tools** group, click the arrow next to the **Restart** icon, and then select one of the following options:

<b>Warmstart</b>	Restarts the real controller and activates the changes made to the system.
<b>Advanced</b>	The controller can be restarted with the following advanced restart options: <ul style="list-style-type: none"><li>• <b>I-start</b></li><li>• <b>P-start</b></li><li>• <b>X-start</b> (FlexPendant required)</li><li>• <b>C-start</b> (FlexPendant required)</li><li>• <b>B-start</b></li></ul>

The options for **Restart** are also present in the context menu when you right-click a controller in the **Controller** browser.

## 11.2.6 Back up a system

### Overview

When backing up a system you copy all the data needed to restore the system to its current state:

- Information about software and options installed on the system.
- System's home directory and all its content.
- All robot programs and modules in the system.
- All configuration and calibration data of the system.

### Prerequisites

To backup a system you must have:

- Write access to the controller
- Logged on to the controller with appropriate grants. For more information, see [User Authorization on page 156](#).

### Creating a Backup

To create a backup, follow these steps:

- 1 In the **Controller** browser, select the system you want to backup from the browser.
- 2 Right click and select **Create Backup**.  
The Create Backup dialog box appears.
- 3 Enter a new backup name and specify a location for the backup, or keep the default ones.
- 4 Click **OK**.

The progress of the backup is displayed in the Output window.

### Backup folder

When the backup is complete you will have a folder with the name of the backup in the specified location. This folder contains a set of subfolders which together comprise the backup.



#### CAUTION

If the contents of the Backup folder are changed, then it will not be possible to restore the system from backup.

Subfolders	Description
BACKINFO	Contains information necessary for re-creating the system's software and options from the mediapool.
HOME	Contains a copy of the system's home directory content.
RAPID	Contains one subfolder for each task in the system's program memory. Each of these task folders contains separate folders for program modules and system modules.
SYSPAR	Contains the system's configuration files.

*Continues on next page*

## 11 Controller tab

---

### 11.2.6 Back up a system

*Continued*



#### Note

The contents of the PIB board of a IRC5P system (a controller system for painting) will not be included with the regular RobotStudio backup. Please use the backup function of the FlexPaintPendant to include the PIB content.

## 11.2.7 Restore a system

### Overview

When restoring a system from backup, the current system gets the same content as when the backup was performed. Restoring a system replaces the following contents in the current system with the content from the backup:

- All RAPID programs and modules in the system.
- All configuration and calibration data of the system.



#### Note

The system's home directory and all its content are copied from the backup to the current system.

### Prerequisites

To restore a system you must have:

- Write access to the controller.
- Logged on to the controller with appropriate grants. For more information, see [User Authorization on page 156](#).

### Restoring a system



#### Note

Before proceeding, make sure that the system from the backup is compatible with the controller you are restoring.

To restore a system, follow these steps:

- 1 In the **Controller** browser, select the system you want to restore.
- 2 Click **Backup** and select **Restore Backup**.  
The Restore from Backup dialog box appears.
- 3 In the **Restore from Backup** dialog box, select which backup to use for restoring the system.
- 4 Click **OK**.

The progress of the restore appears in the Output window. The controller is automatically restarted to load the restored system.



#### Note

If the system from the backup does not originate from the controller you are restoring, you get the following message about the mismatch.

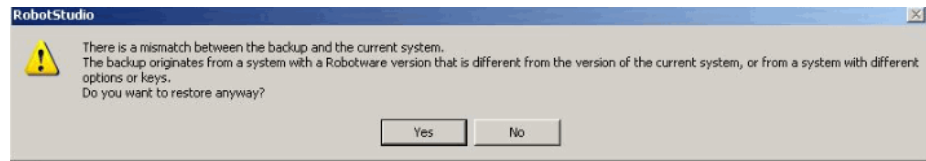
*Continues on next page*

## 11 Controller tab

---

### 11.2.7 Restore a system

*Continued*



en0900001061

## **11.2.8 System Builder**

---

### **Functions of the System Builder**

For procedures using the various functions of the System Builder, see [System Builder Overview on page 158](#).

#### 11.2.9 Configuration editor

---

##### Configuration editor

From the Configuration Editor you view and edit the system parameters of a specific topic in a controller. The Instance Editor is a complementary editor with which you edit the details of a type instance (a row in the Configuration Editor's instance list). The Configuration Editor has a direct communication with the controller. This means that changes you make are applied to the controller as soon as you complete the command.

With the Configuration Editor, including the Instance Editor, you can:

- view types, instances, and parameters
- edit instances and parameters
- copy and paste instances within a topic
- add and delete instances

---

##### Layout of the Configuration Editor

The Configuration Editor consists of the Type name list and the Instance list.

The **Type name** list displays all available configuration types for the selected topic. The list of types is static. This means you cannot add, delete or rename types.

The **Instance** list displays all system parameters of the type selected in the **Type name** list. Each row in the list is an instance of the system parameter type. The columns show each specific parameter and its value for each instance of the parameter type.

The Configuration editor has the following options:

- Controller
- I/O
- Communication
- Motion
- Man-machine communication
- Add Signals

##### Add Signals

You must have write access to the controller to be able to open the add signal window.

Type of Signal	Defines the type of signal.
Signal Base Name	Defines the name for one or more signals.
Assigned to Unit	Defines the I/O unit to which the signal belongs.
Signal Identification Label	Optionally, offers filtering and sorting based on this category.
Number of Signals	Defines the number of signals to add in a range
Start Index	Defines the index (number) to start the range with.
Step	Defines the number which the index should increase with.
Unit Mapping Start	Defines the bits in the I/O memory map of the assigned unit to which the signal is mapped.

*Continues on next page*

Category	Optionally, offer filtering and sorting based on this category.
Access Level	Defines the write access to I/O signals for categories of I/O controlling clients connected to the robot controller. This field is enabled only when Advance checkbox is selected. Not necessarily Write access. Options are Internal, Default, ReadOnly, All
Default Value	Specifies the I/O signal value to be used at the start.
Value at System and Power Failure	Specifies whether the output I/O signal should keep its current value or take the I/O signal's default value in case of system failure or at a power fail.
Invert Physical Value	Applies an inversion between the physical value of the signal and its logical representation in the system.
Store Value at Power Failure	Specifies if the I/O signal should be set to the value stored in the permanent memory pool or not at the start.

**Layout of the Instance editor**

The Instance Editor lists the parameters and their values in the open instance.

In the **Value** column you can view and edit the value of the parameter.

When you click a row, the lower section of the Instance Editor window displays the type of parameter, restrictions for the parameter value and other conditions for the parameter.

#### 11.2.10 Load Parameters

##### Prerequisite

You must have write access to the controller.

##### Loading a configuration file

- 1 In the **Controller** browser, select the system and expand the **Configuration** node.
- 2 Click **Load Parameters** to bring up a dialog box.
- 3 In the dialog box, select how you want to combine the parameters in the configuration file to load with the existing parameters:

If you want to...	then select...
replace the entire configuration of the topic with the one in the configuration file.	<b>Delete existing parameters before loading</b>
add new parameters from the configuration file to the topic, without modifying the existing ones.	<b>Load parameters if no duplicates</b>
add new parameters from the configuration file to the topic and update the existing ones with values from the configuration file. Parameters that only exist in the controller and not in the configuration file will not be changed at all.	<b>Load parameters and replace duplicates</b>

- 4 Click **Open** and browse to the configuration file to load. Then click **Open** again.
- 5 In the information box, click **OK** to confirm that you want to load the parameters from the configuration file.
- 6 When the loading of the configuration file is finished, close the Select mode dialog box.

If a restart of the controller is necessary for the new parameters to take affect, you will be notified of this.

---

## **11.2.11 Save Parameters**

---

### **Overview**

The system parameters of a configuration topic can be saved to a configuration file, stored on the PC or any of its network drives.

The configuration files can then be loaded into a controller. They are thereby useful as backups, or for transferring configurations from one controller to another.

---

### **File-naming conventions**

The configuration files should be named with a name that relates to their corresponding topics. When saving configuration files, the correct name for each file will be suggested by default.

---

### **Saving a configuration file**

- 1 In the **Controller** browser, select the system and expand the **Configuration** node.
- 2 Click **Save Parameters** and select the topic to save to a file and click **Save**.
- 3 In the **Save As** dialog box, browse for the folder to save the file in.
- 4 Click **Save**.

---

### **Saving several configuration files**

- 1 Select the **Configuration** node.
- 2 Click **Save System Parameters**.
- 3 In the **Save System Parameters** dialog box, select the topics to save to files. Then click **Save**.
- 4 In the **Browse for Folder** dialog box, browse for the folder to save the files in.  
Then click **OK**.  
The selected topics will now be saved as configuration files with default names in the specified folder.

#### 11.2.12 Transfer

---

##### Overview

The transfer function allows easy transfer of offline-created RAPID programs to the real robot on the shop floor. This means that you can transfer data from a virtual controller (which is offline) to a real controller (which is online). As part of the transfer function you can also compare the data present in the virtual controller with that present in the real controller and then select which data to transfer.

You can also use the transfer function to transfer data from a virtual controller to another virtual controller.

---

##### Relations for transfer of data

To transfer data, you must first set up a **Relation** between the two controllers. A Relation defines the rules for the transfer of data between the two controllers.

---

##### Creating a Relation

When you have two controllers listed in the Controller browser, you can create a Relation between them. To create a Relation:

- 1 On the **Controller** tab, in the **Transfer** group, click **Create Relation**.  
The Create Relation dialog box is displayed.
- 2 Enter a **Relation Name** for the relation.
- 3 Specify the **First Controller**, from the list. This must be a virtual controller  
The First Controller, also called the Source, owns the data being transferred.
- 4 Specify the **Second Controller**, from the list. This can either be a real controller or another virtual controller.  
The Second Controller, also called the Target, receives the data being transferred.
- 5 Click **Ok**.

The relation between the controllers is now created.

After this, the *Relation* dialog box opens, using which you can configure and execute the transfer. Relations of a controller are listed under its Relations node in the Controller browser.



##### Note

The properties of the relation are saved in a XML file under INTERNAL in the owner controller's system folder.

---

##### Transferring data

You can configure the details of the transfer of data and also execute the transfer, in the *Relation* dialog box.

*Continues on next page*

To open the *Relation* dialog box, double-click a relation. Alternatively, select a relation in the **Controller** browser, and then in the **Transfer** group, click **Open Relation**.

### Configuring the transfer

Before executing a transfer, you can configure the data to be transferred, under the *Transfer Configuration* heading. Configure using these guidelines:

- Use the check boxes in the *Included* column to include or exclude the corresponding items shown in the tree structure. All items in a module that are included will be transferred. Other non-listed items of a module such as comments, records and so on will be automatically included in the transfer.
- The *Action* column shows a preview of the transfer's result, based on the items you include or exclude.
- If a module exists both in the source and the target controllers, and the *Action* column shows *Update*, then click **Compare** in the *Analyze* column. This opens the *Compare* box which shows two versions of the module in different panes. The affected lines are highlighted and you can also step through the changes. You can choose one of the following options for the comparison:
  - **Source with target** - Compares the source module with the target module
  - **Source with result** - Compares the source module with the module that will be the result of the transfer operation
- BASE (module), wobjdata and tooldata are excluded by default.
- wobjdata wobj0, tooldata tool0, and loaddata load0 of the BASE module are unavailable for inclusion.

A task can be transferred only if:

- Write access to the target controller is present (must be manually retrieved).
- Tasks are not running.
- Program execution is in the stopped state.

### Executing the transfer

After you have configured the transfer, you can execute it.

Under the *Transfer* heading, the Source and Target modules are shown along with the arrow showing the direction of the transfer. You can change the direction of the transfer by clicking **Change Direction**. This also switches the source and target modules.

To execute the transfer click **Transfer now**. A dialog showing a summary of the transfer appears. Click **Yes** to complete the transfer. The result of the transfer is displayed for each module in the output window.

The **Transfer now** button is disabled if:

- None of the included tasks can be transferred.
- Write access is required but not held.

Continues on next page

## 11 Controller tab

---

### 11.2.12 Transfer

*Continued*



#### Note

If one of several modules fail, then the following error message is displayed.

Module xxx.zzz has failed. Do you want to continue?

### 11.2.13 Signal Analyzer Online

---

#### Analyzing signals from the controller

The Signal Analyzer Online functionality helps in displaying and analyzing signals from a robot controller. Using the Signal Analyzer, you can optimize the robot program.

The Signal Analyzer functionality is present for both virtual and real controllers. The following section describes the Signal Analyzer functionality for real controllers. For information on the Signal Analyzer Online functionality for virtual controllers, see [Signal Analyzer on page 345](#).

To open the Signal Analyzer Online, on the **Controller** tab, in the **Controller Tools** group, click **Signal Analyzer Online**. Alternatively, you can open the Signal Analyzer Online using the context menu in the **Signal History** window.



#### Note

The Signal Analyzer Online command in the Controller Tools group is enabled only if the selected controller is a real controller, or if the controller tree has only one real controller.

For information about the layout of the Signal Analyzer Online, see [Layout of Signal Analyzer on page 349](#).

---

#### Turning the signal recording on and off

To turn the recording of signals on or off, use the **Start recording** and **Stop recording** buttons.

To start the recording, click **Start recording**.

To stop the recording, click **Stop recording**. The recording stops and the recorded session is saved.

---

#### Configuring signals for the next recording

To configure the signals which are to be saved during the next signal recording session, use the **Signal Setup** window. For this, click the arrow next to the **Signal Analyzer** icon, and then click **Signal Setup**. The **Signal Setup** window appears.

The signals that are available for configuration are shown in the **Signal Setup** window. For the list of available signals, see [Available signals on page 346](#).

For information on setting up the signals, see [Setting up the signals on page 347](#).

*Continues on next page*

## 11 Controller tab

---

### 11.2.13 Signal Analyzer Online

*Continued*

---

#### History

The signal data from each signal recording session is saved. To view these, click the arrow next to the **Signal Analyzer** icon, and then click **History**. For more information, see [History on page 352](#).



#### Note

The signal data from each signal recording session is saved as a .sdf file at the following location. The History feature uses these files.

C:\Users\<your user name>\AppData\Local\ABB Industrial  
IT\Robotics IT\RobotStudio\SignalAnalyzer

This path is for a PC with a standard installation of Windows 7 or 8 (English version). The path may differ if you have a customized installation or use Windows XP.

#### 11.2.14 Safety Configuration

---

##### Overview

For information on safety configuration, see:

- *Application manual - SafeMove*
- *Application manual - Electronic Position Switches*

## 11 Controller tab

---

### 11.3.1 Request Write Access

## 11.3 Features for real controllers

### 11.3.1 Request Write Access

---

#### Overview

You need Write access for editing programs and configurations or in any other way to change data on the controller.

---

#### Prerequisites for Write access

You can get Write access to any controller as long as the prerequisites are fulfilled.

When the Controller is in Mode:	This has to be fulfilled:
Auto	The Write access must not be taken by any other user.
Manual	The remote Write access must be granted on the FlexPendant. For safety reasons, a FlexPendant user can also recall this remote Write access in manual mode.

If the prerequisites are not fulfilled you will be denied, or lose, the Write access. This means that if you have Write access in auto mode and the controller is switched over to manual mode you will lose the Write access without any warning. This is because the FlexPendant unit by default has the Write access in manual mode, for safety reasons. The same will happen if the remote Write access in manual mode is recalled from the FlexPendant unit.

---

#### Result

The Controller Status window will be updated when the request for Write access is granted.

If the Write access is denied, a message is displayed.

### 11.3.2 Release Write Access

---

#### Overview

Several users can be logged on to a single controller but only one can have the write access. You can release the write access when you do not need it anymore.

#### Result

The Controller Status window will be updated when your access right has changed from read/write to read only.

#### 11.3.3 Authenticate

---

##### Overview

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

You can perform the following functions from the **Authenticate** menu:

- Login as a Different User
- Log off
- Log off all controllers
- Edit User Accounts
- UAS Grant Viewer

##### Login as a Different User

- 1 In the **Authenticate** menu, click **Login as a Different User**. The **Add new user** dialog box appears.
- 2 In the **User Name** box, enter the user name you want to log on as.
- 3 In the **Password** box, enter the password for the user name you are logging on as.
- 4 Click **OK**.

**Note:** If you have previously logged on as a different user and wish to revert as default user, click **Login as Default User**.

##### Log off

In the **Authenticate** menu, click **Log off** to log the user off from the controller.

##### Login off all controllers

In the **Authenticate** menu, click **Log off** to log the user off from all the controllers.

##### Edit User Accounts

For more information on User Accounts, see [User Accounts on page 395](#).

##### UAS Grant Viewer

For more information on UAS Grant Viewer, see [UAS Grant Viewer on page 400](#).

## 11.3.4 File transfer

### Overview

You can transfer the files and folders between the PC and a controller through the File Transfer window.

### Prerequisites

The following are the prerequisites to be met:

- The PC must be connected to the same network as the controller, or connected to the service port of the controller.
- You must be logged on to the controller as a user with UAS grants that allows file transferring.

### Transferring files and folders

Use this procedure to transfer files and folders between the PC and a controller:

- 1 In the **Controller Tools** group, click **File Transfer**.  
The **File Transfer** window appears.
- 2 In the **PC explorer**, browse to the folder from or to which you want to transfer the data.
- 3 In the **Controller explorer**, browse to the folder from or to which you want to transfer the data.
- 4 Select the item to transfer from the list.

To select several items at once, do one of the following:

To select	then press
several adjacent items	the <b>SHIFT</b> key and select the first and the last item.
several non-adjacent items	the <b>CTRL</b> key and select each item.
all items in the list	the keys <b>CTRL + A</b>

- 5 When the files and folder to transfer are selected, do one of the following:

To	then press
cut the files	<b>CTRL + X</b>
copy the files	<b>CTRL + C</b> , or click <b>Arrow</b>

*Continues on next page*

- 6 Place the insertion point either in the **PC explorer** or the **Controller explorer** and click CTRL + V.



#### Note

In the **PC explorer** or **Controller explorer** window, right-click to view the following context menu:

- Transfer
- One level up
- Open
- Refresh
- Cut
- Copy
- Paste
- Delete
- Remove

---

## 11.3.5 FlexPendant Viewer

---

### Overview

FlexPendant Viewer is an add-in to RobotStudio that retrieves and displays a screenshot from the FlexPendant. The screenshot is generated automatically at the moment of the request.

---

### Prerequisites

The controller you want to retrieve screen shots from must be added to your robot view.

A FlexPendant must be connected to the controller. If no FlexPendant is currently connected (option *Hot plug* is installed and the jumper plug is used) then no screen shot can be retrieved.

---

### Using FlexPendant Viewer

- 1 Make sure you are connected to the controller.
- 2 In the **Controller Tools** group, click the arrow next to the **FlexPendant** icon, and then click **FlexPendant Viewer**.  
A screen shot will be displayed in the workspace.
- 3 To reload the screen shot, click **Reload** in the workspace.
- 4 To set an automatic reload period for the screen shot, click on the menu **Tools**, point to **FlexPendant Viewer** and click **Configure**.  
Set the desired reload period and select the check-box **Activated**. Then click **OK**.

---

### Results on the controller

The screenshot will automatically be saved as a file on the controller. When a new request is sent, a new screenshot is generated and saved, overwriting the previous file.

No message will be displayed on the FlexPendant.

### 11.3.6 Import Options

---

#### Importing system options

- 1 In the **Configuration** group, click **Import Options** to bring up a dialog box.
- 2 In the **Option source** box, enter the path to the folder where the options to import are located. You can also click the **Browse** button and browse to the folder.
- 3 In the **Media Pool destination** box, enter the path to the media pool you want to store the options in. You can also click the **Browse** button and browse to the media pool folder.
- 4 Select the options to import and click **Import**.

To select several options at once, do one of the following:

To select	then hold down
several adjacent options	the SHIFT key and select the first and the last option.
several non-adjacent options	the CTRL key and select each option.

- 5 Click **OK**.

---

#### Removing system options

- 1 In the **Configuration** group, click **Import Options** to bring up a dialog box.
- 2 In the **Media Pool destination** list, enter the path to the media pool from which you want to delete the options. You can also click the **Browse** button and browse to the media pool folder.
- 3 Select the options to delete and click **Remove**.

To select several options at once, do one of the following:

To select	then hold down
several adjacent options	the SHIFT key and select the first and the last option.
several non-adjacent options	the CTRL key and select each option.

- 4 Click **OK**.

## 11.3.7 Properties

### Overview

You can perform the following from the **Properties** menu:

- Renaming the controller
- Setting the controller date and time
- Setting the Controller ID
- Viewing controller and system properties
- Handling the Device Browser

### Renaming the controller

The controller name is an identification of the controller that is independent of the system or the software running on the controller. Unlike the controller ID, the controller name does not have to be unique for each controller.



#### Note

The controller name must be written with characters from the ISO 8859-1 (Latin 1) character set.

- 1 In the **Configuration** group, click **Properties**, and then click **Rename**.  
The **Rename Controller** dialog box appears.
- 2 Enter the new name of the controller in the dialog box.
- 3 Click **OK**.

The new name will be activated when the controller is restarted.

You will be prompted to either click **Yes** to restart the controller immediately or click **No** to restart later.

### Setting the controller date and time

You can either set the date and time to the same as the computer you are working from, or you can specify the date and time manually.

Use this procedure to set the controller date and time:

- 1 In the **Configuration** group, click **Properties**, and then click **Date and Time**.  
The **Set Date and Time** dialog box appears.
- 2 In the **Set Controller's date and time**, click the arrow next to the date and time list to set the date and time of the controller.



#### Note

Click **Get local computer's time** to set the date and time of the controller to the same as the computer you are working on.

### Setting the Controller ID

The Controller ID is by default set to the serial number of the controller and is thereby a unique identifier of the controller.

*Continues on next page*

## 11 Controller tab

### 11.3.7 Properties

*Continued*

The Controller ID is a unique identifier for the controller and should not be changed. However, if the hard disk of the controller is replaced, the ID will be lost and you must set it back to the serial number of the controller.



#### Note

You must **Request Write Access** to the controller before setting the controller ID.

- 1 In the **Configuration** group, click **Properties**, and then click **Controller ID**. The **Set Controller ID** dialog box appears.
- 2 Enter the Controller ID and then click **OK**.



#### Note

Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 40 characters.

### Viewing controller and system properties

You can view the following properties for a controller and its running system.

Controller Properties	System Properties
Boot Application	Control Module
Controller ID	Drive Module #1
Controller Name	Serial Number
Installed Systems	System Name
Network Connections	

- 1 In the **Configuration** group, click **Properties**, and then click **Controller and System Properties**.  
The **Controller and System Properties** window appears.
- 2 In the tree view at the left of the window, browse to the node for which you want to view the properties.  
The properties of the selected object are displayed in the Properties list to the right of the window.

### Viewing the Device Browser

The Device Browser displays the properties and trends of the various hardware and software devices in a robot controller. To open the Device Browser, in the **Configuration** group, click **Properties**, and then click **Device Browser**.

#### Displaying the properties of a device

In the tree view, browse to the node for which you want to view the properties and then click it. The properties of the selected object, along with their corresponding values, are listed to the right of the tree view.

#### Updating the tree view

Press **F5**, to update the tree view.

*Continues on next page*

Alternatively, right-click inside the tree view pane, and then click **Refresh**.

#### Displaying a trend

Select a device in the tree view and then double-click any property, that has a numerical value, in the right-hand panel. This opens a trend view. The trend view collects data at a rate of one sample per second.

#### Hiding, stopping, starting or clearing a trend

Right-click anywhere on the trend view and then click the required command.

---

#### Saving system diagnostics

You can create a System Diagnostics data file from RobotStudio.

To save a System Diagnostics data file to your PC, in the **Configuration** group, click **Properties**, and then click **Save system diagnostics**.

### 11.3.8 Go Offline

---

#### Overview

The main purpose of this feature is to create a new station with a VC similar to the connected real controller. This helps a robot technician to work offline, and not just when connected to the real controller.

#### Using Go Offline

- 1 Connect the PC to a real controller.
- 2 On the **Controller** tab, click **Request Write Access**.  
For more information on Request Write Access, see [Request Write Access on page 382](#).
- 3 Click **Go Offline**.  
The **Go Offline** dialog box is displayed.
- 4 Enter a name for the system and browse for the location to save the system.  
A new station is created with a VC with the same configuration as the real controller.



#### Note

Go Offline transfers additional options from a real controller and installs them on the PC. A Relation is automatically created between the virtual controller and the real controller.

For more information on Relations, see [Transfer on page 376](#)

### 11.3.9 Online Monitor

This feature allows you to remotely monitor the robot connected to a real controller. It displays a 3D layout of the connected robot controller and enhances user's current perception of reality by adding motion visualization augmentation.

**Note**

The Online Monitor shows TCP robots and TCP robots with track. When connecting the Online Monitor to a virtual controller, the motion is shown only if the virtual controller is using Free-Run mode, not the Time Slice mode.

#### Using Online Monitor

The following procedure describes the Online Monitor feature in RobotStudio:

- 1 Connect the PC to a controller and add the controller. See [Add Controller on page 358](#).
- 2 Click **Online Monitor**.

The 3D view of the mechanical units of the controller system is displayed in the graphics window.

**Note**

The robot view is refreshed every second with the current jointvalues of all the mechanical units.

#### Indication of TCP

A cone is automatically created to indicate the active tool data being used. The cone has its base in the robot wrist and its tip at the location of the tool data.

#### Kinematic Limitations

When the Kinematic Limitation button is enabled, the graphical 3D viewer indicates whether the robot is at a joint limit or at a singularity.

For joint limits, the corresponding link is highlighted in yellow to indicate a warning and in red to indicate an error. The tolerance limits are defined in RobotStudio Options - Online - Online Monitor.

For singularity, a markup indicates when the axis 5 is close to singularity. The singularity level is also defined in RobotStudio Options.

#### Visualizing Safety Zones in Online Monitor

This feature allows the user to visualize the current status of the manipulators in a robot system and provides an augmented reality of the robot cell. The user can visualize a failure scenario of a robot, that is, when the robot makes an unplanned stop. When the robot enters a restricted zone, the safeMove supervision feature stops the robot. To give the user an idea of the physical layout and the safety zone that has caused the stop, the safety zones are visualized in online monitor.

*Continues on next page*

#### Features

- A **Show Safety Zones** button is available in the Online Monitor for each manipulator in the system, for example, four buttons in a MultiMove system with four manipulators.



#### Note

The eight tool points defined in the SafeMove configuration for monitoring the tool position and speed are not visualized in Online Monitor.

- The name of each tool zone and the corresponding manipulator is shown as a markup, for example, Rob1 STZ1, ..., Rob4 STZ8, Rob1 MTZ1, ..., Rob4 MTZ8 and so on.
- Zones that are defined as **Allow inside** in the definition of the zone are visualized as a green semitransparent hollow shape.
- Zones that are defined as **Allow outside** in the definition of the zone must be visualized as a red semitransparent solid shape.
- A message is displayed in the output window if no STZ or MTZ is defined for the manipulator.
- The controller event log message **20468 SC STZ violation** is displayed in the output window if it is present in the controller event log.

### 11.3.10 User Accounts

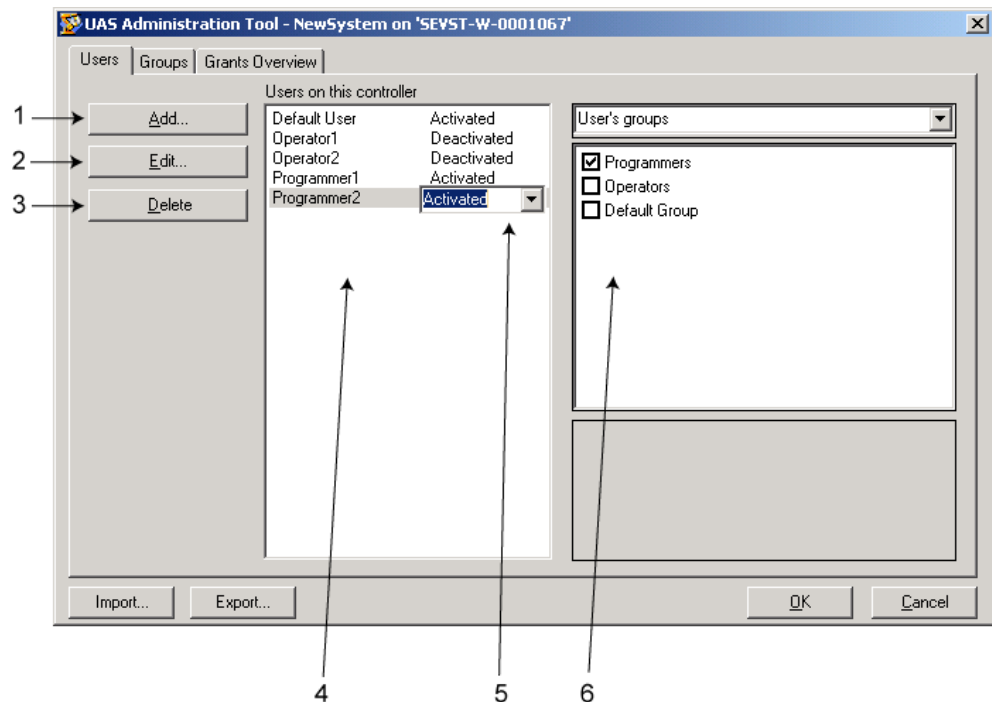
#### Overview

All the procedures below require the following steps to be taken before managing the details:

- 1 In the **Controller** browser, select the controller to which you want to manage a user or group
- 2 From the **Controller** tab, click **Request Write Access** to provide write access to the controller.
- 3 In the **Controller** tab, click **Authenticate** and select **Edit User Accounts**, for administering UAS accounts, grants, and groups.

#### Users tab

With the Users tab you set which users will be able to log on to the controller and which groups the users shall belong to.



users-ta

#### Users tab Parts

- 1 The **Add** button. Opens a dialog box for adding new users.
- 2 The **Edit** button. Opens a dialog box for changing the log on name and password of the user.
- 3 The **Delete** button. Deletes the selected user account from the controller.

*Continues on next page*

## 11 Controller tab

### 11.3.10 User Accounts

*Continued*

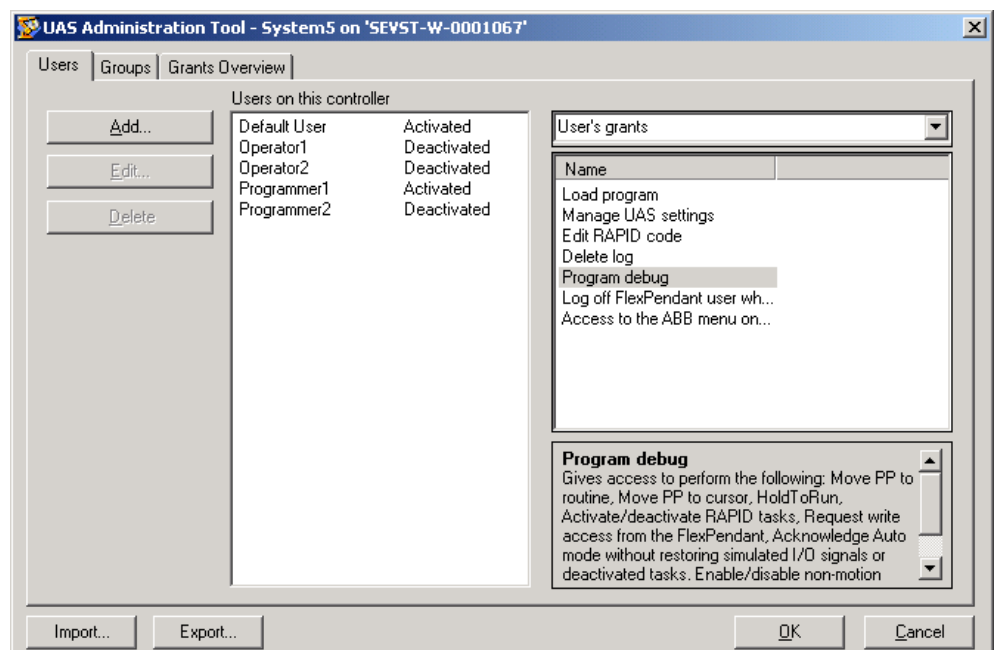
- 4 The **Users on this Controller** list. Shows the user accounts defined on this controller. The list has two columns:

Column	Description
User	The name of the user account
Status	Shows if the account is activated or deactivated. When deactivated, it is not possible to log on using that account.

- 5 The **Activated/Deactivated** item box. Changes the status of the user account.
- 6 The **User's groups/User's grants** list.

The **User's groups** list shows which group(s) the user is a member of. For changing the membership of a group, select or clear the checkbox in front of the group name.

The **User's grants** list shows the available grants for the selected User's group(s). When selecting a grant from the User's grants list, a description of the selected grant appears.



users-t0

### Adding a user

- 1 On the **Users** tab, click **Add** to bring up a dialog box.
- 2 In the **User Name** box, enter the user name. Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 16 characters.
- 3 In the **Password** box, enter the user's password. The password you type in will not be visible. Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 16 characters.
- 4 In the **Retype Password** box, enter the user's password again.
- 5 Click **OK** to add the new user and close the dialog box.
- 6 Click **OK**.

*Continues on next page*

#### Deleting a user

- 1 On the **Users** tab, select the user to delete from the **Users on this controller** list and click **Delete**.
- 2 To the question **Are you sure you want to remove this user?**, answer **Yes**.
- 3 Click **OK**.

---

#### Setting up group membership

- 1 On the **Users** tab, select the user from the **Users on this controller** list.
- 2 In the **User's groups** list, select the groups the user shall be a member of.
- 3 Click **OK**.

---

#### Changing a name or password

- 1 On the **Users** tab, select the user to edit from the **Users on this controller** list and click **Edit user**.  
This opens the **Edit** dialog box.
- 2 To change the user name, enter the new name in the **User Name** box. Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 16 characters.
- 3 To change the password, enter the new password in the **Password** box, then retype the password in the **Retype Password** box. Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 16 characters.
- 4 Click **OK** to save the changes to the user and close the dialog box.
- 5 Click **OK**.

---

#### Activating or deactivating a user

- 1 On the **Users** tab, select the user from the **Users on this controller** list and click the status text (Activated or Deactivated). An item box appears and you can change the status.  
The user's new state is now displayed in the status column of the **Users on this controller** list.
- 2 Click **OK**.

---

#### Exporting a user list

On the **Users** tab, select the user from the **Users for this Controller** list and click **Export**.

This opens a **Save as** dialog box, in which you specify the name and location for the file with the user list.

---

#### Importing a user list

On the **Users** tab, select the user from the **Users for this Controller** list and click **Import**.

This opens an **Open file** dialog box, in which you browse to the file with the list to import.

*Continues on next page*

## 11 Controller tab

### 11.3.10 User Accounts

*Continued*

When you have selected the file, the ImportOptionsForm dialog appears.

Select ...	Description
Delete existing users and groups before importing	Earlier groups and users will be deleted.
Advanced options	A new dialog appears. Import users but don't replace duplicates means that you will not replace the existing users. Import users and replace duplicates means that you will replace the existing users. Import groups but don't replace duplicates means that you will not replace the existing groups. Import groups and replace duplicates means that you will replace the existing groups.

#### Adding a group

- 1 On the **Groups** tab, click **Add**.  
This opens the **Add new group** dialog box.
- 2 In the **Group Name** box, enter the name of the group. Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 16 characters.
- 3 Click **OK** to add the new group and close the dialog box.
- 4 Click **OK**.

#### Renaming a group

- 1 On the **Groups** tab, select the group to rename from the **Groups on this controller** list and click **Rename**.  
This opens the **Rename Group** dialog box.
- 2 In the **Group Name** box, enter the name of the group. Use only characters from the ISO 8859-1 (Latin 1) character set and no more than 16 characters.
- 3 Click **OK** to rename the group and close the dialog box.
- 4 You will now be asked if you want the users who belong to this group to continue belonging to a group with the old name as well as the new one.

Click	To
<b>Yes</b>	Let the users of the group be members to groups with both the old and the new name. However, the old group will no longer be defined in the controller's UAS since it is replaced by the new group.  This option might be useful if you plan to recreate the old group, or copy the user's settings to another controller who has the old group defined.
<b>No</b>	Delete the user's memberships to the old group. This is to just replace the old group name with the new one.
<b>Cancel</b>	To cancel the change and keep the old group name, with its user's memberships.

- 5 Click **OK**.

*Continues on next page*

---

**Deleting a group**

- 1 On the **Groups** tab, select the group to delete from the **Groups on this controller** list and click **Delete**.
- 2 You will now be asked if you want the users who belong to this group to continue belonging to it though it is not valid.

Click	To
<b>Yes</b>	Let the users of the group remain members to it even if it is no longer defined in the controller's UAS.  This option might be useful if you plan to recreate the group, or copy the user's settings to another controller who has the group defined.
<b>No</b>	Delete the user's memberships to the group.
<b>Cancel</b>	Cancel the change and keep the group, with its user's memberships.

- 3 Click **OK**.

---

**Giving grants to a group**

- 1 On the **Groups** tab, select the group from the **Groups on this controller** list.
- 2 In the **Controller grants/Application grants** list, select the grants to give to the group.
- 3 Click **OK**.

## 11 Controller tab

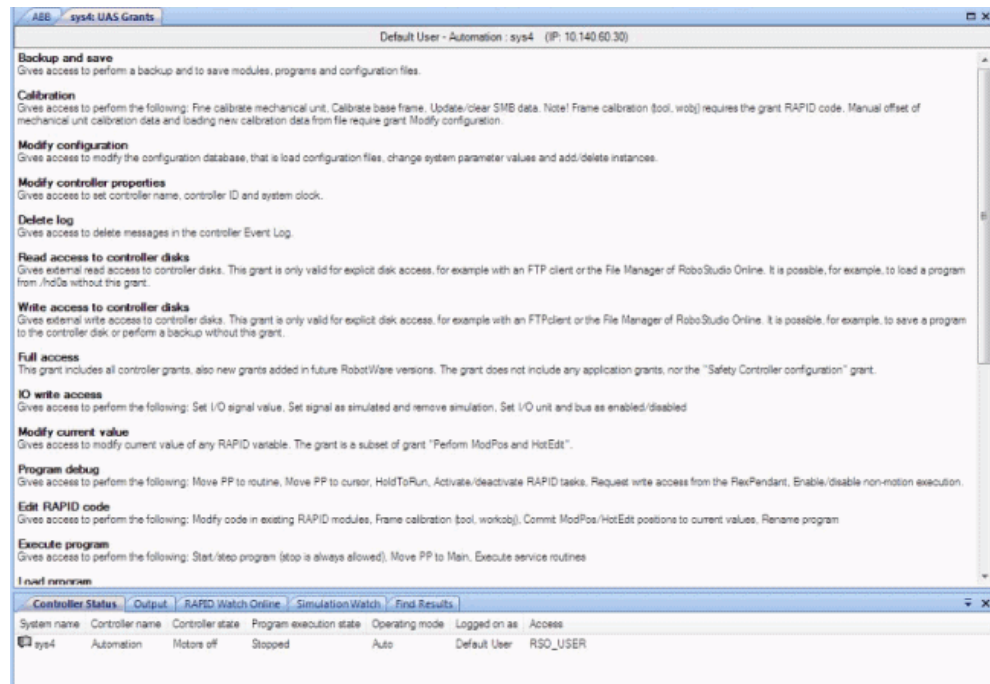
### 11.3.11 UAS Grant Viewer

### 11.3.11 UAS Grant Viewer

#### Overview

The UAS Grant Viewer page displays information about the grants provided to the user currently logged in and the groups owning them.

- 1 In the **Authenticate** menu, click **UAS Grant Viewer**. The **UAS Grants** window appears.



en0900000852

#### Examples of common actions to perform

Action	Necessary grants
Rename the controller (A restart of the controller is necessary)	Modify controller properties Remote warm start
Change system parameters and load configuration files	Modify configuration Remote warm start
Install a new system	Administration of installed system
Perform a backup (A restart of the controller is necessary)	Backup and save Remote warm start
Restore a backup (A restart of the controller is necessary)	Restore a backup Remote warm start
Load/delete modules	Load program
Create new module.	Load program
Edit code in RAPID modules	Edit RAPID code
Save modules and programs to disk	Backup and save
Start program execution from Task Window	Execute program

*Continues on next page*

Action	Necessary grants
Create a new I/O signal, that is, add a new instance of the type Signal ( A restart of the controller is necessary)	Modify configuration Remote warm start
Set the value of an I/O signal	I/O write access
Access to controller disks from File Transfer window	Read access to controller disks Write access to controller disks

### Controller grants

Full access	This grant includes all controller grants, also new grants added in future RobotWare versions. The grant does not include any application grants or the <i>Safety Controller configuration</i> grant.
Manage UAS settings	Gives access to read and write the UAS configuration, that is to read, add, remove and modify UAS users and groups.
Execute program	Gives access to perform the following: <ul style="list-style-type: none"> <li>Start/step program (stop is always allowed)</li> <li>Move PP to Main</li> <li>Execute service routines</li> </ul>
Perform ModPos and HotEdit	Gives access to perform the following: <ul style="list-style-type: none"> <li>Modify or teach positions in RAPID code (ModPos)</li> <li>During execution modify positions in RAPID code as single points or as a path (HotEdit)</li> <li>Restore ModPos/HotEdit positions to original</li> <li>Modify current value of any RAPID variable</li> </ul>
Modify current value	Gives access to modify current value of any RAPID variable. This grant is a subset of the grant <i>Perform ModPos and HotEdit</i> .
I/O write access	Gives access to perform the following: <ul style="list-style-type: none"> <li>Set I/O signal value</li> <li>Set signal as simulated and remove simulation</li> <li>Set I/O unit and bus as enabled/disabled</li> </ul>
Backup and save	Gives access to perform a backup and to save modules, programs and configuration files. The grant gives full FTP access to the current systems BACKUP and TEMP directory.
Restore a backup	Gives access to restore backup and perform B-start.
Modify configuration	Gives access to modify the configuration database, that is to load configuration files, change system parameter values and add/delete instances.
Load program	Gives access to load/delete modules and programs.
Remote warm start	Gives access to perform warm start and shutdown from a remote location. No grant is required to perform warm start via a local device, as for example the FlexPendant.
Edit RAPID code	Gives access to perform the following: <ul style="list-style-type: none"> <li>Modify code in existing RAPID modules</li> <li>Frame calibration (tool, workobj)</li> <li>Commit ModPos/HotEdit positions to current values</li> <li>Rename program</li> </ul>

Continues on next page

## 11 Controller tab

### 11.3.11 UAS Grant Viewer

*Continued*

Program debug	Gives access to perform the following: <ul style="list-style-type: none"><li>• Move PP to routine</li><li>• Move PP to cursor</li><li>• HoldToRun</li><li>• Activate/deactivate RAPID tasks</li><li>• Request write access from the FlexPendant</li><li>• Enable/disable non-motion execution</li></ul>
Decrease production speed	Gives access to decrease speed from 100% in Auto mode. This grant is not required if speed is already below 100%, or controller is in Manual mode.
Calibration	Gives access to perform the following: <ul style="list-style-type: none"><li>• Fine calibrate mechanical unit</li><li>• Calibrate base frame</li><li>• Update/clear SMB data</li></ul> Frame calibration (tool, wobj) requires the grant <i>Edit RAPID code</i> . Manual offset of mechanical unit calibration data and loading new calibration data from file require the grant <i>Modify configuration</i> .
Administration of installed systems	Gives access to perform the following: <ul style="list-style-type: none"><li>• Install new system</li><li>• P-start</li><li>• I-start</li><li>• X-start</li><li>• C-start</li><li>• Select System</li><li>• Install system from device</li></ul> This grant gives full FTP access, that is, the grant gives the same rights as <i>Read access to controller disks</i> and <i>Write access to controller disks</i> .
Read access to controller disks	Gives external read access to controller disks. This grant is only valid for explicit disk access, for example with an FTP client or the File Manager of RoboStudio. It is possible, for example, to load a program from /hd0a without this grant.
Write access to controller disks	Gives external write access to controller disks. This grant is only valid for explicit disk access, for example with an FTP client or the File Manager of RoboStudio. It is possible, for example, to save a program to the controller disk or perform a backup without this grant.
Modify controller properties	Gives access to set controller name, controller ID and system clock.
Delete log	Gives access to delete messages in the controller Event Log.
Revolution counter update	Gives access to update the revolution counter.
Safety Controller configuration	Gives access to perform a configuration of the Safety Controller. This is valid only for the PSC-option and is not included in the <i>Full access</i> grant.

*Continues on next page*

---

**Application grants**

Access to the ABB menu on FlexPendant	Value <b>true</b> gives access to the ABB menu on the FlexPendant. This is the default value if a user does not have the grant. Value <b>false</b> means that the user cannot access the ABB menu when the controller is in Auto mode. The grant has no effect in Manual mode.
Log off FlexPendant user when switching to Auto mode	A user having this grant is automatically logged off from the FlexPendant when switching from Manual mode to Auto mode.

#### 11.3.12 Integrated Vision

The Integrated Vision system provides a robust and easy-to-use vision system for general purpose Vision Guided Robotics (VGR) applications. The system features a complete software and hardware solution that is fully integrated with the IRC5 robot controller and the RobotStudio programming environment. The vision capability leverages on the Cognex® In-Sight® smart camera family, with embedded image processing and an Ethernet communication interface.

RobotStudio has been equipped with a vision programming environment that exposes the full palette of Cognex EasyBuilder® functionality with robust tools for part location, part inspection and identification. The RAPID programming language has been extended with dedicated instructions and error tracing for camera operation and vision guidance.

For more information, see *Application manual - Integrated Vision*

## **11.4 Features for virtual controllers**

### **11.4.1 Virtual FlexPendant**

---

#### **Opening a Virtual FlexPendant**

You can open a virtual FlexPendant in one of the following ways:

- 1 On the **Controller** tab, in the **Controller Tools** group, click the arrow next to the **FlexPendant** icon, and then click **Virtual FlexPendant**.
- 2 Press the keyboard shortcut, **CTRL + F5**.



#### **Note**

The Virtual FlexPendant is applicable while running a virtual controller. For information on specifying the appearance and placement of virtual FlexPendant, see [Options on page 195](#).

## 11 Controller tab

---

### 11.4.2 Control Panel

### 11.4.2 Control Panel

---

#### The Control Panel dialog box

<b>Operation Mode</b>	This group contains the three operational modes of the controller represented by option buttons.
<b>Auto</b>	This option corresponds to the Auto mode on the FlexPendant. Moving between the <b>Auto</b> and <b>Manual Full Speed</b> options must proceed via the <b>Manual</b> option.
<b>Manual</b>	This option corresponds to the Manual mode on the FlexPendant.
<b>Manual Full Speed</b>	This option corresponds to the Manual 100% mode on the FlexPendant. Moving between the <b>Auto</b> and <b>Manual Full Speed</b> options must proceed via the <b>Manual</b> option.
<b>Motors On</b>	Click this button to turn on the motors.
<b>Enable Device</b>	In a manual mode, click this button to simulate holding the enabling device to turn on the motors.
<b>Release Device</b>	In a manual mode, click this button to turn off the motors.
<b>Reset Emergency Stop</b>	If the controller enters the emergency stop state, click this button to reset the state.

### 11.4.3 Shutdown

---

**Shutting down a controller**

- 1 In the **Controller** browser, select the controller to shut down.
- 2 Right-click the controller, and then click **Shutdown**.

**Note**

Suppose you want to restart the controller, select Warmstart. For more information on restarting a controller, see [Restart a controller on page 365](#).

#### 11.4.4 Set Task Frames

---

##### Modifying Task frame

- 1 On the **Controller** tab, in the **Virtual Controller** group, click **Task Frames**. The **Modify Task Frames** dialog box appears.
- 2 Set the reference to **World**, **UCS**, or **Local**.
- 3 Edit the position and orientation of task frames in the **Task Frames** coordinate box.
- 4 Click **Apply**.

To the question, *Do you also want to move the Base Frames(s)?*

- Click **Yes** to move the base frame, but keeps its relative placement to the task frame.
- Click **No**. The following question appears **Do you want to update the controller configuration and restart?**. Click **Yes** to restart the controller and update the base frame configuration of the connected VC.



##### Note

If there are any stationary RAPID objects (tooldata, workobjects) connected to the robot, the following question appears **Do you want to keep the positioning of all stationary RAPID objects?**

- Click **Yes** to keep all the stationary RAPID objects in their global coordinates.
- Click **No** to move all the stationary RAPID objects along with the base frame (same coordinates relative to base frame).

## 11.4.5 Edit System

### Overview

The Edit System window contains functions for making and viewing advanced system configurations, such as changing controller and baseframe positions, calibrating and setting up external axes.

The left side of the Edit System window contains a hierarchical tree with which you browse to different aspects of the system. The right side contains a property sheet for the aspect selected in the tree. Below are short descriptions of the property sheets for each aspect node of the tool.



#### CAUTION

Editing the system may result in corrupted systems or unexpected robot behaviors. Be sure to understand the effects of the changes before proceeding.

### The System node

The system node contains a box with information about the system and a button for loading new parameters (configuration files) to the system.

### The task node

The task node has no property page.

### The mechanism folder node

The property page of this node contains controls for mapping and setting axis and joints. It is from this page you set up external axes.

### The mechanism library node

The property page of this node contains controls for changing the baseframe of the robot or mechanism. Here, too, you specify whether the baseframe is moved by another mechanism (coordinated motion), like a track external axis.

### Updating the baseframe position

- 1 Move the mechanical unit (robot or external axis) to its new location using the ordinary tools for moving and placing objects.
- 2 In the **Controller** browser, select the controller for the mechanical unit.
- 3 On the **Controller** tab, in the **Virtual Controller** group, click **Edit System**. This opens the **System Configuration** dialog.
- 4 Select the node for the mechanical unit in the hierarchical tree. The baseframe property sheet for the robot is now displayed.
- 5 Select the baseframe position values to use after restarting the robot.

Select	To
Controller values	Reset all changes to the baseframe made since the last time the system was started.

*Continues on next page*

## 11 Controller tab

---

### 11.4.5 Edit System

*Continued*

Select	To
Stored station values	Reset all changes made to the baseframe since the last time the station was saved. Optionally, you can enter new values in the baseframe coordinate boxes (relative to the controller world coordinate system).
Use current station values	Read and use the current location of the baseframe. Optionally, you can enter new values in the baseframe coordinate boxes (relative to the controller world coordinate system).

6 Click OK.



#### Note

For information on adding a track from the Edit System tool, see [Track motion of type RTT or IRBTx003 on page 83](#).

## 11.4.6 Encoder Unit

### Configuring an Conveyor Encoder Unit

- 1 Click **Encoder Unit**.

The Configure Conveyor Encoder Unit dialog box appears.

Alternatively, the Configure Conveyor Encoder Unit dialog box can be opened from the **Paths&Targets** browser. Right-click a station in the browser, select **Configuration**, and then click **Encoder Unit**.

- 2 Select **CNV1** from the **Mechanical Unit** list.
- 3 In the **Parameters** box, enter the values for **Maximum Distance**, **Minimum Distance**, **Queue Tracking Distance** and **Start Window Width**.



#### Note

If any of the parameter values are changed, the controller must be restarted.

- 4 Click **OK**.
- 5 Click **Yes** to restart the controller.

**This page is intentionally left blank**

## **12 RAPID tab**

### **12.1 Overview of the RAPID tab**

---

The RAPID tab provides tools and functionalities for creating, editing, and managing RAPID programs. You can manage RAPID programs which are online on a real controller, offline on a virtual controller, or standalone programs which are not part of a system.

## 12.2 Synchronize to Station

---

### Synchronizing to the station

- 1 On the **RAPID** tab, in the **Access** group, click the arrow next to the **Synchronize** icon, and then click **Synchronize to Station**.
- 2 Select the paths to be synchronized to the station from the list.
- 3 Click **OK**.

The message **Synchronization to Station completed** is displayed in the Output window.



#### Note

This function is also present in the **Controller** group on the **Home** tab.

## 12.3 Synchronize to VC

### Synchronizing to the virtual controller

- 1 On the **RAPID** tab, in the **Access** group, click the arrow next to the **Synchronize** icon, and then click **Synchronize to VC**.
- 2 Select the elements to be synchronized to the VC from the list.
- 3 Click **OK**.

The message **Synchronization to VC completed** is displayed in the Output window.



#### Note

This function is also present in the **Controller** group on the **Home** tab.

## 12.4 Edit RAPID code

---

### Editing RAPID code using RAPID Editor

The RAPID editor enables you to view and edit programs loaded into a controller, both real and virtual. The integrated RAPID editor is useful for editing all robot tasks other than robot motion. With the RAPID editor you can edit the RAPID code of the program modules and system modules. Each module you open appears in an editor window of its own, where you can add or edit RAPID code.

For examples of using the RAPID editor, see [Examples of using the RAPID editor on page 441](#).

### General RAPID Editor features

The following are the general features of the RAPID Editor:

- **Read-only documents** - If the document is read-only (for example, due to lack of mastership), then the background of the editor area will be light gray instead of the normal white. Typing in an editor that is in the read-only state results in a dialog asking you whether RobotStudio should acquire write access.
- **Syntax highlighting** - Text is highlighted in different colors depending on their token classification (such as keyword, identifier and so on). You can configure these colors in the *File* tab, under *Options:Robotics:RAPID Editor*. For more information, see [Options:Robotics:RAPID Editor on page 197](#).  
In addition to token classification, the editor also shows different colors for built-in and installed identifiers (such as MoveL) and also for identifiers declared in user code.
- **Quick-Info tooltips** - When you hover the mouse pointer over a symbol (such as a data declaration or procedure call), a tooltip is displayed describing the symbol. For many built-in symbols (such as MoveJ) a short description is also displayed. For symbols corresponding to a data declaration, the current value is also displayed.
- **Context-sensitive help** - Pressing F1 when the cursor is on a RAPID programming construct, such as an instruction, opens the related section in the RAPID reference manual, instead of the main RobotStudio help.
- **Auto-indent cursor on return** - When you press Enter, the cursor is automatically indented by the appropriate amount on the following line. For example, after typing a PROC header, pressing ENTER will indent the cursor one tab (or the corresponding number of spaces, depending on settings).
- **Completion list** - When you type in code in the editor, a pop-menu which lists possible code suggestion maybe displayed depending on the kind of RAPID code construct being written. The suggestions listed also depend on where in the document the cursor is.

Pressing comma (,), semi-colon (;), colon (:), equal sign (=), Spacebar, Tab, or Enter keys automatically inserts the selected item. Press Esc to cancel the list.

*Continues on next page*

- **Auto-completion** - After typing or completing a procedure call (such as MoveJ), pressing the Tab key will fill in all required parameters. Note that this is only available for certain built-in procedures, such as those listed in the *Insert Instruction* menu.
- **Argument information** - While typing in procedure calls and function calls, tooltips showing argument information are displayed.
- **Collapsible regions** - Certain regions of the code can be collapsed. For example, Data declarations area, routines, IF/WHILE/FOR statements and so on.
- **Error highlighting** - Red squiggly lines appear under errors in the code. All syntax errors and a subset of semantic errors are indicated in this manner.
- **Zooming in and out** - In the RAPID editor you can zoom in and zoom out of the code display. Click the plus (+) and minus (-) buttons at the top right corner of the RAPID editor window to zoom in and zoom out.

**Tip**

The Zoom in Zoom out feature is also present in the *RAPID Tasks*, *Rapid Editor*, *Configuration Editor*, *Event viewer*, and *I/O* windows.

- **Cut, copy, paste and drag and drop** - These standard commands for clipboard handling of text are supported.
- **Undo and redo** - Standard commands for undo and redo operations are supported.
- **Selection modes** - You can select text by character, row and column.
- **Line numbers** - Line numbers for the RAPID code lines are displayed in the left margin of the editor.
- **Keyboard shortcuts** - For keyboard shortcuts in the RAPID Editor, see [Keyboard shortcuts on page 72](#).

### Starting the RAPID Editor

To open a RAPID module in the RAPID editor, in the **Controller** browser right-click on a RAPID module, and then click **RAPID Editor**.

The RAPID code of the module opens in the editor window.

**Tip**

You can view the graphical layout, without closing the editor, by clicking the graphics window tab.

### Editing a RAPID program

The Edit group on the RAPID tab has commands which help in editing the lines of code in the RAPID Editor. Other than standard functions such as Cut, Copy, and Paste, the following functions are present in the Edit group:

- **Comment** – To comment out selected lines

*Continues on next page*

## 12 RAPID tab

---

### 12.4 Edit RAPID code

*Continued*

Uncomment – To uncomment commented lines

The comment and uncomment buttons in the ribbon will add/remove comment characters (“!”) at the beginning of the selected line(s).

- Indent – To increase the indent of selected line(s) by four white spaces  
Unindent – To decrease the indent of selected line(s) by four white spaces  
The indent and unindent buttons in the ribbon will move the selected code line(s) one tab position to the right/left.
- Format Document - Auto-formats the active document by arranging the spaces and tabs in the RAPID code.
- Uppercase Keywords - To change RAPID keywords from lowercase to uppercase. This function operates on the current document.
- Format Selection – This function is similar to Format Document, but with the difference that it operates on currently selected text.
- Convert Spaces to Tabs - Converts consecutive spaces to the corresponding number of tabs. This function operates on the current selection.
- Convert Tabs to Spaces - Does the opposite of function above.



#### Note

To make formatting easier, the tabs and white spaces can be indicated by arrows and dots, respectively. To enable this go to Options:Robotics:RAPID Editor ([Options:Robotics:RAPID Editor on page 197](#)) and then select the **Show whitespace** check box.

By default a Tab consists of four whitespaces. To change this, go to Options:Robotics:RAPID Editor and set the Tab size as your require.

Edited lines are denoted by change bars which remain until the edits are applied. Also, the RAPID Editor's tab sports an asterisk (\*) until the edits are applied.

---

### Adding code snippets

Code Snippets are pieces of code which you can insert into the RAPID Editor. To view and select a code snippet, in the Insert group, click Snippet.

The list which appears show two kinds of code snippets:

- Predefined code snippets
- User defined code snippets

The following are the predefined code snippets in RobotStudio:

- Array of num, 2x2x4
- Array of num, 2x4
- Array of num, 2x4x2
- Array of num, 4x2
- Module header
- Procedure with parameters
- Procedure with error handler
- Robtarget declaration

*Continues on next page*

- Tooldata declaration
- Workobject declaration

You can also create your own code snippets or save a section of existing code from the RAPID editor as a code snippet. Such user created code snippets are also listed along with the predefined snippets.

To save a section of existing code, from the RAPID editor, as a code snippet:

- 1 Select the code you wish to save as a snippet.
- 2 In the **Insert** group, click the arrow next to the **Snippet** icon, and then click **Save Selection as Snippet**.

The **Save As** dialog box appears. Specify a name for the snippet and save it. The RobotStudio .snippet files are saved in the following folder.

*C:\<Documents and Settings>\<user name>\RobotStudio\Code Snippets*

To insert a snippet in the RAPID editor, click the arrow next to the **Snippet** icon, and then click the required snippet from the listed snippets.



#### Note

The folder *<Documents and Settings>* may be configured with different names, for example, *Data*. It may also be translated on localized versions of Windows.

Snippets can also be edited in an XML editor such as Microsoft Visual Studio.

For information on creating customized code snippets, see <http://msdn.microsoft.com/>.

### Inserting instructions

To insert a predefined instruction into the code:

- 1 Place the cursor at the required point in the RAPID code.
- 2 In the **Insert** group, click **Instruction**.

A list of pre-defined instructions is shown.

The instruction is inserted into the code where the cursor is placed.

RobotStudio generates and inserts default arguments to the instruction, using similar rules as the FlexPendant.

### Applying and verifying the edits

To apply the changes made in the editor to the system and also to check the program go to the **Controller** group, and click the arrow next to the **Apply** icon. Then:

- To apply only the changes in made in the module, currently shown in the editor, click **Apply Changes**.

Alternatively, you can also directly click the **Apply** icon

- To apply the changes made in all modified modules, click **Apply All**.

*Continues on next page*

## 12 RAPID tab

---

### 12.4 Edit RAPID code

*Continued*



#### Note

The Apply commands are enabled only if there are changes waiting to be applied. When possible, RobotStudio will try to commit the changes without losing the program pointer. If this is not possible, you will be asked if it is OK to lose the program pointer.

To verify the syntactic and semantic correctness of the modules, in the **Test and Debug** group, click **Check Program**.

## 12.5 Find and replace RAPID code

---

### Overview

The Find group, on the RAPID tab contains commands for performing Find and Replace actions on the code in the RAPID editor.

---

### Quick Find

Enter the search string in the **Quick find** box and then press Enter or F3. If an instance is found, it is highlighted. Press F3 again to search for the next instance.

---

### Go to line

Enter a line number in the **Go to line** box and press Enter. The cursor moves to the corresponding line in the RAPID editor.

---

### Jump To

The Jump To list has an item for each routine and data declaration in the program module. Click an item to move to its location in the code.

---

### Find or Replace

Click **Find/Replace** to open the Find/Replace dialog box. This dialog box provides standard find/replace functionality in addition to the following:

Use the **Look in** list box to specify where to look for in a find/replace operation. You can choose to search in the Current Document, Current System or in a folder on your PC (you can browse to a folder to specify it).

The *Search Results* window displays the results of a *Find* operation. Double-click a search result to go to the corresponding instance in the RAPID editor. If the instance is from a module which is not in the RAPID editor, then the module automatically opens in the editor.

---

### Go To Definition

The **Go To Definition** command is enabled for an identifier in the RAPID Editor context menu if the source code for the corresponding symbol definition is available.

Click **Go To Definition** to move the cursor to (and select) the corresponding symbol definition. This action detects symbol definitions such as routine declarations, data declarations and record definitions.

---

### Find Unused References

Click **Find unused references in Task** to see all data declarations in the task of the active module document that are not used anywhere. The results are shown in the *Search Results* window. Click **Find unused references in Module** to see unused data declarations in the current module.

---

### Find All References

The **Find All References** command is enabled for identifiers in the editor code.

For a given identifier, click **Find All References** to search through the entire task for uses of the same identifier (including its definition). Note that this is not just a

*Continues on next page*

## 12 RAPID tab

---

### 12.5 Find and replace RAPID code

*Continued*

string search. It takes RAPID scoping rules into account. For PERS and syncident data, this function searches the other tasks for a matching global symbol and return the uses of those.

## 12.6 Manage RAPID modules

### Managing file based RAPID modules

File based RAPID modules can be opened in the editor in four different ways:

- Using the *Open* command on the File tab
- Using the *New:RAPID Module File* command on the File tab. For more information, see [Creating a new RAPID module file on page 190](#).
- Double-clicking a module in the File browser of the RAPID tab. For more information on the File browser, see [Manage RAPID files and backups on page 426](#).
- Right-clicking the Files nodes and selecting **Open** in the File browser of the RAPID tab. For more information on the File browser, see [Manage RAPID files and backups on page 426](#).



#### Tip

For file based modules, the standard file commands are applicable: **Save/Save As** will save the module; **Open** will open a module and **Close** will close the module.

The **Apply Changes** command is disabled for file based modules. It is applicable only for controller based modules.

### Creating a new RAPID module

- 1 On **RAPID** tab, in the **Controller** browser, right-click a task and then click **New Module**.  
The Create Module dialog box opens.
- 2 Enter a module name.
- 3 Select the **Module type** as Program or System, as required.
- 4 Select one of the following options:
  - No Step-In - The module cannot be entered during step-wise execution.
  - Read-only - The module cannot be modified.
  - View-only - the module cannot be modified, but the attribute can be removed.
- 5 Click **Create**.

### Loading a RAPID module

- 1 On **RAPID** tab, in the **Controller** browser, right-click a task and then click **Load Module**.
- 2 Browse to and select the module to be loaded to your station, and then click **Open**.

### Saving a RAPID module as another

- 1 On **RAPID** tab, in the **Controller** browser, right-click a module and then click **Save Module As**.

*Continues on next page*

## 12 RAPID tab

---

### 12.6 Manage RAPID modules

*Continued*

- 2 Browse to the location where the new module is to be saved and then click **Save**.

## 12.7 Edit RAPID data

### RAPID Data Editor overview

The RAPID Data Editor allows you direct access to RAPID data values, which you can view and edit.

To open the RAPID Data Editor, on the **RAPID** tab go to the **Controller** browser, right-click a RAPID module, and then click **RAPID Data Editor**. This opens the Data window which shows the data declarations in that particular module.

Data declarations are grouped according to their data types. All data declarations belonging to a data type are shown in a table below it. Each row corresponds to a data declaration and shows the contents of the declaration.

### Using the RAPID Data Editor

- Editing the values of a row opens the changed value in the RAPID Editor window. The new value is shown in both the Data editor and also the RAPID editor. This means that the changes made in the RAPID data editor are seen in the RAPID editor, and vice-versa.



#### Tip

A asterisk (\*) on the window tab indicates that the changes have not been saved.

- You can select multiple cells and edit them together.
- You can create, edit or delete a data declaration from the RAPID Data Editor.
- To delete a data declaration, select the row and click the Delete button beside it.
- To add a new declaration, click **New Declaration** next to the required data type. This adds a new row to the table below it having some default properties and values, which can be edited. However, you cannot add a declaration of a data type that is not already present in the module. In such as case you need to manually add the declaration to the module using the RAPID Editor.



#### Note

The RAPID Data Editor only shows data declarations that contain editable values.

## 12.8 Manage RAPID files and backups

---

### Managing RAPID files

In the Files browser, right-click the **File** node and then click **Open**. The Open File dialog box appears, using which you can browse to and open system module (\*.sys), RAPID modules (\*.mod), and Configuration files (\*.cfg) which are residing on your PC or on a network.

Upon opening a RAPID or system module file, it opens in the RAPID editor. The system parameters file (\*.cfg) open in a notepad-like editor but is unlike the RAPID editor. To save the changes made in an editor, click the **Save** button on the Quick Access Toolbar.



#### Note

When you open standalone RAPID modules, the editor may show the code as having syntax errors if the variable declarations exist in another module.

### Managing system backups

Right-click **Backup** and click **Browse**, to browse to and open system backups.

The structure of the backup is reflected in the Files browser under the **Backups** node. There is one node for each task defined in the system. The RAPID modules of each task are shown as its child nodes in the tree view. The editor will find data declared in other modules and correctly mark the code as being syntactically and semantically correct.

The contents of the **HOME** folder are shown in a separate folder. RAPID modules of the **HOME** folder will be edited in the standalone mode, which means that the editor will not find data declared in other modules. The reason is that the editor cannot know in which context (task) the module should be treated.

The **SYSPAR** folder will show the configuration files.



#### Note

There is no syntax check or intellisense for editing configuration files.

## 12.9 Manage RAPID code on the controller

### 12.9.1 Manage RAPID programs

---

#### Loading a RAPID program

To load a RAPID program to a station:

- 1 On the RAPID tab, in the **Controller** group, click **Program** icon and then select **Load Program**.  
Alternatively, in the Controller browser, right-click the active task under the station, and click **Load Program**.
  - 2 In the *Open* dialog box that appears, browse to the location of the program to be loaded to your station and click **Open**.
- 

#### Saving a program

- 1 On the RAPID tab, in the **Controller** group, click **Program** icon and then click **Save Program As**.  
Alternatively, in the Controller browser, right-click the active task under the station, and select **Save Program As**.
  - 2 In the *Save As* dialog box that appears, browse to the location where you want to save your program, and click **Save**.
- 

#### Renaming a program

- 1 On the RAPID tab, in the **Controller** group, click **Program** icon and then click **Rename Program**.  
Alternatively, in the Controller browser, right-click the active task under the station, and select **Rename Program**.
  - 2 In the *Rename* dialog box that appears, enter a new name for your program, and click **Ok**.
- 

#### Deleting a program

- 1 On the **RAPID** tab, in the **Controller** group, click **Program** and select **Delete Program**.  
A confirmation dialog is displayed.
- 2 Click **Yes**.  
The selected program is deleted.

To delete the entire program under a task in a station, in the **Controller** group, click **Program** and then click **Delete Program**.

Alternatively, in the Controller browser, right-click the task under the station, and then click **Delete Program**.

### 12.9.2 RAPID Tasks

#### Overview

The RAPID Tasks window shows the configured tasks of the selected controller and their state, in a tabular form. To open the RAPID Tasks window, in the **Controller** group click **RAPID Tasks**.

The following table describes the columns displayed for each task.

Task Name	The task name, as defined by the controller configuration in topic Controller, type Task. For information on topic Controller type Task, see the <i>Technical Reference Manual for System Parameters</i> .
Type	A task can be of type Normal, Static, or SemiStatic. This is defined by the controller configuration in topic Controller, type Task. For information on topic Controller type Task, see the <i>Technical Reference Manual for System Parameters</i> .
Mechanical Unit	Shows which mechanical unit group is used for the task. This is defined by the controller configuration in topic Controller, type Task For information on topic Controller type Task, see the <i>Technical Reference Manual for System Parameters</i> .
Run Mode	Defined by the Run Mode setting in RobotStudio. For more information on Run Mode, see <a href="#">Run mode of the controller on page 431</a> .
State	Displays the task execution state. A task can be in state Ready, Running, or Stopped. <ul style="list-style-type: none"><li>Ready: The program has no PP (program pointer). To set the program pointer, use the Program Pointer menu of the RAPID Tab. Alternatively, use the FlexPendant.</li><li>Running: The program is running.</li><li>Stopped: The program has stopped.</li></ul> For more information on the Program Pointer (PP), see <a href="#">Using the Program Pointer on page 436</a> .
TrustLevel	Handles the system behavior when a SemiStatic or Static task is stopped or not executable. The possible values here are NoSafety, SysFail, SysHalt or SysStop. A SemiStatic or Static task can only be stopped if it has TrustLevel as NoSafety. The TrustLevel is defined by the controller configuration in topic Controller, type Task. For information on topic Controller type Task, see the <i>Technical Reference Manual for System Parameters</i> .
Program Name	The name of the program in the specific task.
Module Name	The current module name.
Routine Name	The current routine name.
Task in Foreground	Use this to set priorities between tasks. The current task will execute only if the foreground task is idle. This is defined by the controller configuration in topic Controller, type Task.

*Continues on next page*

**Task execution state**

A task can be activated, started, and stopped from the Controller browser, with the following limitations:

- Only Normal tasks can be activated and deactivated. Background tasks will always be automatically activated.
- Background tasks of type Static and SemiStatic can only be started and stopped if they have TrustLevel NoSafety.

For detailed information about the different TrustLevel values, see the *Technical reference manual - system parameters*.

- You need to have write access and the appropriate grant.
- The limitations concerning task execution that hold for the FlexPendant also apply to RobotStudio.

The following table shows cases where task execution state cannot be changed.

If...	RobotStudio gives a message that informs the user that...
the user does not have the grant <i>Execute program</i> or <i>Full access</i>	the operation is not possible.
the user changes from manual mode to automatic mode, or vice versa, the user loses the write access and	the operation is not possible.
the motors are in off state	the Start operation is not possible.

**Note**

It is not possible to override the controller's safety system, that is, you cannot stop a background task (Static and SemiStatic) that has the TrustLevel set to a value other than NoSafety.

For detailed information about the different TrustLevel values, see *Technical reference manual - system parameters*.

**Activating, starting and stopping tasks**

To activate a task, right-click the task in the Controller browser and then turn on the **Active** command.

If the prerequisites are met, you can operate the task, such as start and stop the task, move the program pointer to main and set the run mode.

To start a task, right-click the task in the Controller browser and then click **Start Task**. You can start Normal tasks, but you can only start a Static or SemiStatic task if the TrustLevel is set to NoSafety.

**CAUTION**

When starting a task, the manipulator axes may move very quickly and sometimes in unexpected ways! Make sure no personnel is near the manipulator arm!

*Continues on next page*

## 12 RAPID tab

---

### 12.9.2 RAPID Tasks

*Continued*

To stop a task, right-click the task in the Controller browser and then click **Stop Task**. You can stop Normal tasks, but you can only stop a Static or SemiStatic task if the TrustLevel is set to NoSafety.

### 12.9.3 Run Mode

---

#### Run mode of the controller

The Run Mode indicates the mode of the controller. It has the following two options:

- Continuous
- Single

You can set the run mode of the controller in the following ways:

- On the **RAPID** tab, in the **Controller** group, click **Run Mode** and then click either **Continuous** or **Single**.
- This method is only applicable to virtual controllers in a station.

On the **Simulation** tab, in the **Configure** group, click **Simulation Setup** and then select **Continuous** or **Single** in the **Setup Simulation** dialog box.

#### 12.9.4 Adjust Robtargets

---

##### Overview

The Adjust Robtargets feature helps in recalculating and changing the robtarget data (tooldata and workobject data) while maintaining the joint angles of the robot. The robtarget data related to the specified source tooldata and workobject will be adjusted for usage with the new tooldata and workobject.

##### Prerequisites

- You should have a controller (virtual or real) running with one or more modules containing procedures with a sequence of move instructions expressed with a defined tool and workobject.
- You should have RobotStudio Premium license to use this feature.
- The **Execute** button of the feature Adjust Robtargets will be enabled only if the selected tool data or workobject data have the same properties, such as robhold, ufprog, ufmech and so on.



##### Note

Inline targets, arrays, event records, and offsets are not supported. Relative tool is also not supported. Circular move instruction (MoveC) is supported.

##### Using Adjust Robtargets



##### Note

Take a backup copy of your modules before adjusting your robtargets.

The following procedure describes the Adjust Robtargets feature in RobotStudio:

- 1 On the **RAPID** tab, in the **Controller** browser, select a RAPID task or module under the RAPID icon. Then click **Adjust Robtargets** on the **RAPID** tab.

Alternatively, right-click the RAPID task or module in the **Controller** browser, and then click **Adjust Robtargets** in the context menu.

The Robtarget Adjust dialog box appears.



##### Note

You can access **Adjust Robtargets** from the **Controller** tab also. Right-click the RAPID task or module in the **Controller** browser and then click **Adjust Robtargets** in the context menu.

- 2 If the module you want to adjust is selected, then go to Step 4. Otherwise continue with the next step.

*Continues on next page*

- 3 Select a task from the **Task** drop-down list and module from the **Module** drop-down list.

**Note**

In the **Module** drop-down list, you can either select a particular module or **<ALL>** to update.

- 4 Select the source robtarget data (that is, the data defined in the selected task) from **Old tooldata** and **Old wobjdata** drop-down list.
- 5 Select the destination robtarget data (that is, new tooldata and workobject) from **New tooldata** and **New wobjdata** drop-down list.
- 6 Click **Execute**.

The **Execute** button is enabled only if source robtarget data (that is, old tooldata and workobject) and destination robtarget data (that is, new tooldata and workobject) are different.

The module searches for move instructions that use the specified old tooldata or workobject and recalculates the robtarget data for the new tooldata and workobject.

For example,

- 1 Select "tool0" as the source tool and "wobj0" as the source workobject.
- 2 Select "toolb" as the new tool and "wobjb" as the new workobject.
- 3 Click **Execute**.

Robtargets of "tool0" and "wobj0" will be replaced with re-calculated robtargets which correspond to the same robot configuration (all joint angles will be the same), and with the new tool "toolb" and "wobjb". Note that both the tooldata and the wobjdata are replaced independently.

---

**Update instruction**

By default, the **Update instruction** check box is selected. This means that move instructions using the specified source (old) tooldata and workobject will be updated to use the target (new) tooldata and workobject in addition to recalculating the robtargets.

If the **Update instruction** check box is cleared, the robtargets will be recalculated, but the move instructions will not be updated. They will still use the source tooldata and workobject.

This feature is useful after the calibration of tooldata and workobject. After calibration, you might still want to use the old names of the tooldata and workobject, but update their values and recalculate the robtargets accordingly. The following sample procedure illustrates how this can be accomplished.

**Sample procedure**

**Prerequisite:** RAPID module with robtargets and move instructions that use uncalibrated tooldata tool1, and workobject wobj1.

- 1 Calibrate your tooldata tool1, and workobject wobj1. Store the new values in tool1\_calib and wobj1\_calib, respectively. Keep the old values of the uncalibrated tooldata and workobject in tool1, and wobj1.

*Continues on next page*

## 12 RAPID tab

---

### 12.9.4 Adjust Robtargets

*Continued*

- 2 Open the Adjust robtargets tool and clear the **Update instruction** check box. Select your RAPID module, enter tool1, and wobj1 as your old tooldata and wobjdata, and tool1\_calib, and wobj1\_calib as the new tooldata and wobjdata, respectively.
- 3 Click **Execute**, and apply the changes to the controller from the RAPID Editor
- 4 In the RAPID Editor, rename your tooldata tool1 to tool1\_uncalib, and tool1\_calib to tool1 and apply the changes to the controller. Also, perform the same for wobj1.

Now, your robtargets are updated to match the calibrated values of tool1 and wobj1.

---

### Limitations

If a robtarget is used more than once but with different tools or workobjects, then a message *Target is referenced* is displayed in the output window.

## 12.10 Test and debug

### 12.10.1 Commands for testing and debugging

The Test and Debug group on the RAPID tab consists of the following commands.

Command	Description
<b>Start</b>	Starts the execution of all normal RAPID tasks in the system.
<b>Stop</b>	Stops the execution of all normal RAPID tasks in the system.
<b>Step over</b>	Starts and executes one statement in all normal tasks in the system.
<b>Step in</b>	Starts and executes into a routine, while stopping at the beginning of the routine.
<b>Step out</b>	Executes all remaining statements of the current routine, and stops after the call to the current routine.
<b>Breakpoint : Ignore breakpoints</b>	Ignores all breakpoints during simulation.
<b>Breakpoint : Toggle breakpoint</b>	Toggles a breakpoint at the cursor.
<b>Check Program</b>	Verifies the syntactic and semantic correctness of the RAPID modules.

Other tools such as the Program Pointer (PP) and the RAPID Profiler, which aid in testing and debugging RAPID code, are explained in detail in the following sections.

#### 12.10.2 Using the Program Pointer

---

##### How the Program Pointer helps

During program execution, the Program Pointer (PP) points to the line of code that is currently executing.

The function **Follow Program Pointer** keeps the program pointer visible during program execution by automatically scrolling the RAPID editor window according to the movements of the program pointer. To enable the function, in the Test and Debug group on the RAPID tab, click the arrow next to the Program Pointer icon and then select **Follow Program Pointer**.



##### Note

While the program executes, you can see the program pointer jump across modules only if those modules are already open in the editor. Hence, you can decide in which modules you wish to follow the program pointer in, and keep them open.

The other commands in the Program Pointer menu are:

- Go To Program Pointer – To show the current location of the program pointer in the RAPID editor
- Go To Motion Pointer – To show the current location of the motion pointer in the RAPID editor
- To set the program pointer at a particular line code or code segment and then start program execution from that point, use the **Set Program Pointer** options. You can choose from the following options:
  - Set Program Pointer to Main in all tasks
  - Set Program Pointer to Cursor
  - Set Program Pointer to Routine

---

##### Maintaining the Program Pointer

The RAPID code can only be edited when the controller is not running, that is when it is in state Ready or Stopped. In Ready state the program pointer is not set, but in Stopped state the Program Pointer is set to a specific location of the program. For limited changes to the RAPID code of a controller in Stopped state, the current location of the program pointer can be maintained. After such an edit you can resume program execution from where it was without having to reset the program pointer.



##### Note

If the edit is too large for the program pointer to be maintained then a warning message is displayed to convey this.

The program pointer cannot be maintained, for example, when editing the line of code at which the program pointer is located. Editing that line of code results in

*Continues on next page*

resetting the program pointer. In effect, the program will start from the beginning when the controller is started after the edit.

**WARNING**

Starting program execution after the program pointer has been reset will cause the robot to move along the shortest path from its current location to the first point of the program.

## 12 RAPID tab

---

### 12.10.3 Using the RAPID Profiler

#### 12.10.3 Using the RAPID Profiler

---

##### What's RAPID Profiler

The RAPID Profiler analyzes the execution times on procedure level, identifies critical procedures and reports these during the execution of RAPID code.

##### Prerequisites for using the RAPID Profiler

- You should have RobotStudio Premium license to use this feature.
- You should have a controller with one or more executable tasks running.
- When using the RAPID Profiler with a real controller, you require more than 25 MB of free controller disk space.



##### Note

The RAPID Profiler will automatically stop if either of the following controller events is generated by the controller. This is to avoid disturbing the robot operation.

- **20192, Disk memory low** (less than 25 MB free storage left)
- **20179, Disk memory critically low** (less than 10 MB free storage left, program execution is stopped)

##### How to use the RAPID Profiler

To use the RAPID Profiler:

- 1 Set the Program Pointer at a desired point in the RAPID code from where you wish to start your analysis. For example, set the program pointer to Main in all tasks.
- 2 On the **RAPID** tab, in the **Test and Debug** group, click the arrow next to the **RAPID Profiler** icon and then click **Start**.
- 3 Start the simulation.  
In the background the RAPID Spy feature logs the data about the program's execution.
- 4 After the program execution ends, click the arrow next to the **RAPID Profiler** icon and then click **Stop**.
- 5 On the **RAPID** tab, in the **Test and Debug** group, click the arrow next to the **RAPID Profiler** icon and then click **Analyze**.

The RAPID Profiler window appears, showing the results of the analysis.

Click **Export to Excel** to export the results to a Microsoft Excel spreadsheet file.

To view the log file of the analysis, click the arrow next to the **RAPID Profiler** icon and then click **Open log file**.

##### Execution of the Rapid Profiler based on RobotWare version

Depending on the RobotWare version, the RAPID Profiler is executed in one of the following methods:

- For controller systems with RobotWare versions prior to 5.14, the RAPID instructions *SpyStart* and *SpyStop* must be inserted at the RAPID execution

*Continues on next page*

start and end, respectively. When the program is run, a Spy log file is generated. You can open the file for analysis by the RAPID Profiler. Use the RAPID Profiler menu option *Browse for Spy log* to open the log file.

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

**Note**

When the RAPID Profiler is used to analyze a log file, there is no information about in which procedure the *SpyStart* command is executed. The triggering procedure defaults to *<SpyStart Procedure>*.

- For RobotWare version 5.14 or later, the log file can be generated automatically. Activate the RAPID Profiler and run the program of the controller. When the program execution stops, the results are presented to the user.

#### 12.11 RAPID Watch window

##### Viewing variables and I/O signals

The RAPID Watch window displays the following details of selected variables and I/O signals during program execution.

Column	Description
Name	Displays variable name
Value	Displays variable value
Type	Displays type of datatype
Source	Displays system name

You can view and edit the RAPID data of the variables in the RAPID watch window, both during program execution and when the controller is stopped. However, you can only view, but not edit, I/O signals in the watch window.

To view a variable or I/O signal in the RAPID Watch window, you need to first add it to the window. In the RAPID editor, right-click the required variable or I/O signal, and then click **Add Watch**.

By default, during program execution the values of the variables are automatically refreshed in the watch window every 2 seconds. You can also manually refresh the values.

To enable or disable automatic refresh, in the context menu, select or clear the **Auto Refresh** command.

To do a manual refresh, in the context menu, click **Refresh** (keyboard shortcut F5).



##### Note

CONST variables cannot be edited.

On closing RobotStudio, the variables and signals added to the watch window are removed.

On the **RAPID Watch** window, right-click to display the following context menu:

Item	Used for
Copy	Copying the value
Paste	Pasting the copied value
Delete	Deleting the watch item
Select All	Selecting all the items
Clear All	Clearing all the variables and signals from the watch window
Refresh	Manually updating the values of the variables and signals
Auto Refresh	Automatically refreshing the values displayed in the watch window at regular intervals

## 12.12 Examples of using the RAPID editor

### Overview

This section provides examples illustrating several useful functions of the RAPID editor including IntelliSense, code snippets and the watch window.

### Editing

Assume that you wish to create an infinite loop whereby the controller receives commands from a line PLC. The controller communicates with the PLC using digital I/O signals, but you have forgotten the exact name of the function that reads an input signal.

- 1 Using code snippets, create a new procedure.
- 2 On the **Rapid** tab, in the **Insert** group, click **Instruction**.  
A drop-down list of available instructions is displayed.
- 3 On the **Instruction** menu, point to **I/O**, and then click **DOutput**.
- 4 Press the spacebar to display the parameter information ToolTip. As you enter parameters, the ToolTip is updated, displaying the current argument in bold. The ToolTip is closed either by concluding the instruction with a semicolon (;), or by pressing **ESC**.



#### Tip

At any time you may press **CTRL + Spacebar** to complete what you have begun typing. This will either bring up a narrowed-down list of selectable parameters, or, if only one selection remains, will automatically complete your text.



#### Tip

After typing the name of an identifier or an instruction, press the **TAB** key to automatically fill in the default arguments or parameters. For instructions, the last used argument of each type will be used.

### Searching

Assume that you have programmed targets and motion instructions and synchronized them to the controller. The number of targets is large, so you decide to distribute them among several modules.

You may have forgotten in which module your main procedure is found.

- 1 Press **CTRL + F** to bring up the **Find and Replace** dialog box.
- 2 In the **Find what** box, type "PROC main". Since no modules are open, in the **Look In** list, select **Current System**, and then click **Find All**.  
The search result is displayed in the *Search Results* window.
- 3 Double-click the line matching your search to launch the RAPID editor.

*Continues on next page*

## 12 RAPID tab

---

### 12.12 Examples of using the RAPID editor

*Continued*

---

#### Adding breakpoints

Now that you have finished editing, you may want to test your loop and add some breakpoints.

- 1 Place the insertion on the new statement and press **F9** to set a breakpoint.
- 2 Ensure that the **Ignore breakpoints** button in the editor toolbar is not clicked, and click the **Play** button on the **Simulation** toolbar.  
The program will run and then stop at the breakpoint.
- 3 To run the program statement by statement, click the **Step over** button in the editor toolbar.

---

#### Executing

You might want to debug your loop or monitor a specific variable.

- 1 In the RAPID editor browser, right-click the procedure you want to set as entry point, and then click **Set Program Pointer to Routine**.
- 2 In the **RAPID** tab, click the **Play** button.  
The program will run and then stop at the next breakpoint.
- 3 Select a variable for monitoring and drag it to the watch window.
- 4 Restart the loop and monitor the variable at each iteration.

## 13 Add-Ins tab

### Overview of the Add-Ins tab

The Add-Ins tab contains the control for PowerPacs, Migrate backup and Gearbox Heat Prediction. The Add-Ins browser shows the installed PowerPacs, General add-ins.

For instruction on building General add-ins visit the ABB Robotics Developer Center web site at <http://developercenter.robotstudio.com>.

General add-ins are loaded from the following folder on your PC: C:\Program Files (x86)\Common Files\ABB Industrial IT\RoboticsIT\RobotStudio\Addins



#### Note

For the RobotStudio 5.61:

- Add-ins will be loaded from the following folder: C:\Program Files (x86)\ABB Industrial IT\RoboticsIT\RobotStudio 5.61\Bin64\Addins

This is the path on a PC with Microsoft Windows 7 64-bit English version, on default installation. For customized installations and for operating system versions in other languages, this path may differ.

### Gearbox Heat Prediction

#### Overview

The Gearbox Heat Prediction Tool is an add-in for RobotStudio which helps predict heat problems in the gearboxes. When the temperature is above a predefined value, you can adjust the cycle to reduce the temperature or order a fan that can cool down the gear.

Robots with compact gearboxes have a risk of getting overheated under certain circumstances. The gearbox temperature is supervised by Service Information System (SIS). It is a software function within the robot controller, that 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. It also supervises large robots from damaging the motors during high load operations with a safety shutdown.

The temperature supervision is based on an algorithm that predicts the stationary temperature of the gearboxes and motors of the robot. The algorithm predicts the heat based on the character of the robot motion and also the room temperature. Intensive motion (high average speed and /or high average torque and/or short wait time) will increase the heat of the gearbox and motors.

*Continues on next page*

To avoid overheating, SIS stops the robot if the temperature becomes too high. For large robots, there is an option to add a cooling fan to axis 1, 2 and sometimes axis 3 to allow the robot to run even with a heavy duty program.



### Note

Gearbox Heat Prediction is not supported for Tool and External axis.

### Prerequisites

- 1 RobotStudio 5.14.02 or later.
- 2 RobotWare 5.14.01 or later.
- 3 RobotStudio station with controller having a programmed cycle that includes payload for the robot.

### Calculating the Gearbox Heat

Use the following procedure for predicting the heat generated by the robot:

- 1 Create a new station or open a saved station. The Gearbox heat button is now visible in the **Add-Ins** tab.
- 2 In the **Add-Ins** tab, click Gearbox Heat. The **Gearbox Heat Prediction** window opens.
- 3 In the **Add-Ins** tab, select **Enabled** to enable the **Gearbox Heat Prediction** tool .



### Note

For a manipulator without compact gear, **Gearbox Heat Prediction** is disabled.

- 4 Run a simulation.



### Note

If the RobotStudio license is expired, the **Play** button in the **Simulation** tab will be disabled. As such, you will be unable to run the simulation from the Simulation tab. In such a scenario, use the **Play** button which will now be visible in the **Gearbox Heat Prediction** tab window to run the simulation.



### Note

The data is recorded during the simulation only if the **Gearbox Heat** tool is enabled. After the recording is finished, you can perform another recording or perform a calculation for heat related problems.

- 5 In Cycles, define the behavior of the cycle for predicting the heat generated by the robot:
  - **Continuous:** Select this option if you want the robot to continuously calculate the predictions without waiting time between two consecutive cycles.

*Continues on next page*

- **Number of cycles per hour:** Select this option if you want to manually specify the number of cycles per hour for calculation.
- **Waiting time between cycles (sec):** Select this option to specify the waiting between cycles. Specify the waiting time in seconds.

6 In **Ambient Temperature**, define the ambient temperature.

- Use the slider to change the temperature.
- Select **Use temperature from controller(s)** to reset the the ambient temperature.



**Note**

The ambient temperature used in the calculations should be the same that is used in the configuration of the actual robot in its real environment.

7 Calculate the result in either of the following ways:

- In the **Recordings** section, either double-click a recording or select a recording and click **Calculate**.
- In the **System** section, either double-click a controller or select a controller and click **Calculate**.



**Note**

- The **Recordings** section displays the recordings to be analyzed when **Gearbox Heat Prediction** is enabled.
- The **System** section displays all the available controllers. Data for all controllers are recorded at all times and you can select the controller to be analysed from the list.

The results are displayed for each joint and with fans for the joints that can have fans installed as an option.



**Note**

The following factors influence the heat accumulated:

- Axis speed
- Payload
- Room temperature (ambient temperature)
- Waiting time (to allow robot to cool down)



### Note

The calculated energy is displayed as different heat levels:

- **Green:** Indicates no heat problem
- **Orange:** Indicates it is recommended to install a fan.
- **Red:** Indicates a fan must be installed.
- **Grey:** Indicates it is not possible to calculate the possible energy level for this joint.
- **Not available:** Indicates the joints that cannot have a fan installed.



### Note

Recommended action is displayed along with the warning level for each joint.

- **Joint:** Represents the joint.
- **Without fan:** Displays the percentage of heat levels calculated to the corresponding joint without fan.
- **With fan:** Displays the percentage of heat levels calculated to the corresponding joint with fan.
- **Action:** Displays the recommended action.

---

### Migrate backup feature

The Migrate backup feature is an add-in to RobotStudio. This helps in migrating IRC5 backups from RobotWare versions 5.15 to RobotWare versions 5.60 and later.

To use the tool, press **Migrate backup** in the **Add-Ins** tab and then select the controller backup you want to migrate. Click **OK** to start the migration.

Information about the migration progress will be displayed in the output window. After migration, configuration files that contain incompatibilities are opened in a text editor. Configuration parameters that are not migrated automatically will be commented out together with an explanation. These parameters must be migrated manually.

The backup is ready to be restored when migration is completed for all incompatible files. Incompatible configuration files are retained in the SYSPAR folder with \*.bak extension.

## **14 Context menus**

### **14.1 Add to Path**

---

**Creating a move instruction based on an existing target**

- 1 Select the target for which to create the move instruction.
- 2 From the **Home** menu, in the **Path Programming** group, select the type of move instruction to create.
- 3 Click **Add to Path**.  
The move instruction will appear under the path node as a reference to the original target.

## 14 Context menus

---

### 14.2 Align Frame Orientation

### 14.2 Align Frame Orientation

---

#### The Align Frame Orientation dialog box

<b>Reference</b>	Specify the frame or target for which you want to align the selected objects here.
<b>Align Axis</b>	The axis you specify here will be aligned as on the reference target/frame for all selected objects.
<b>Lock Axis</b>	The axis you specify here will not be changed on the selected objects by the align function, but will keep its orientation.

## 14.3 Align Target Orientation

### Aligning target orientation

- 1 Select the targets whose orientation you wish to change.
- 2 Click **Align Target Orientation** to bring up a dialog box.
- 3 In the **Reference** box, specify the target whose orientation you want to use as reference, by first click in the box and then selecting the target either from the graphics view or the **Layout** browser.
- 4 In the **Align Axis** box, select the axis whose orientation you want to copy from the reference target to the selected ones.
- 5 In the **Lock Axis** box, select the axis to rotate the target around. The orientation of this axis will not be changed on the targets. For example, if the Z axis of all targets are orientated normally to the surface of the work piece and you want to keep it this way, you should lock the Z axis.
- 6 Click **Apply**.



#### Tip

You can change the Align and Lock axis and click Apply again to reorientate the targets until you deselect them.

## 14 Context menus

---

### 14.4 Attach to

### 14.4 Attach to

---

#### Attaching an object

- 1 In the **Layout** browser, right-click the child object, click **Attach to** and click the parent object in the list.
- 

#### Attaching an object by drag and drop

- 1 In the **Layout** browser, drag the child object to the parent object.
- 2 In the displayed message, click the corresponding button:

To	Click
attach the child object and move it to the attachment point	<b>Yes</b>
attach the child object and keep its position	<b>No</b>
not perform the attachment	<b>Cancel</b>

## 14.5 Configurations

### Auto Configuration

Use this procedure for setting the configuration of all targets in the path that are marked as *The configuration is not verified* :



#### Note

For all targets in the path, the function will ignore any existing unverified configuration and replaces it with optimal configuration with respect to the configuration of the preceding target.

- 1 In the **Paths&Targets** browser, right-click a path, select **Configurations** and then select **Auto Configuration**.

The robot now steps through each target in the path and sets the configurations.



#### Note

- If the first target in the path has no configuration assigned, then the configurations tool appears.
- If the first target has a configuration assigned, then the one assigned will be used.

The result of auto configuration varies depending on the configuration of the first target.

The configuration for targets in the path that have a verified configuration, will not be re-assigned.

### Reset Configurations

The configuration data which is a part of the target, when reset, is optimized by Auto Configuration. As a result, the target / move instruction icon will change and is marked as *The configuration is not verified*.

Use this procedure for resetting the configuration:



#### Note

You can reset the configuration of a path, target, or move instruction.

- 1 In the **Paths&Targets** browser, right-click a path, select **Configurations** and then select **Reset Configurations**.



#### Note

To reset the configuration of a target or move instruction,

In the **Paths&Targets** browser, right-click a target or move instruction, and select **Reset Configuration**.

*Continues on next page*

## 14 Context menus

---

### 14.5 Configurations

*Continued*

---

#### Verify Configurations

Use this procedure for verifying the existing configuration:



#### Note

Targets and Move instructions marked as *The configuration is not verified* can be verified with respect to the configuration.

- 1 In the **Paths&Targets** browser, right-click a path, select **Configurations** and then select *Verify Configurations*.



#### Note

If the existing configuration is correct, then the move instruction is set as verified.

If the configuration is incorrect, then the target is set as unreachable.

## 14.6 Check Reachability

### Checking the reachability

You can use the Check Reachability function to check whether targets are reachable or not. If you select a path for the check, then the reachability of all move instructions in the path is checked. This function provides an easy reachability check which you can use for initial positioning of the robot, its workobject, paths and targets.

The Check Reachability function ignores the robot axis configuration. The function indicates a target as being reachable if it can be reached with any robot axis configuration, and ignores the defined robot axis configuration.



#### Note

The Check Reachability feature does not verify whether a path can be executed or not.

- 1 In the **Paths&Targets** browser, right-click the workobject or target or path to be checked for reachability.
- 2 Click **Reachability** to see the reachability status of the selected object.

The frames for the selected object change color in the graphic window based on the reachability status.

Color	Means
Green	The object can be reached.
Red	The object cannot be reached at its current position.

## 14.7 Configurations

---

### Manually setting a robot axis configuration for single targets

- 1 In the **Paths&Targets** browser, select a target and then click **Configurations** to bring up a dialog box.
- 2 If more than one configuration solution exist, examine them by clicking them, one at a time.

The position of the robot with the selected configuration will be displayed in the graphics window, and the joint values for the configuration will be displayed in the joint values list below the configurations list.

In most cases, selecting a configuration similar to the previous one is the best choice.

- 3 Select the configuration to use and click **Apply**.

## 14.8 Convert Frame to Workobject

---

### Converting a frame to a workobject

- 1 In the **Layout** browser, select a frame.
- 2 Click **Convert Frame to Workobject**. The new workobject will appear in the **Paths&Targets** browser.
- 3 Optionally, rename or edit the workobject in any way.

## 14 Context menus

---

### 14.9 Convert to Move Circular

#### 14.9 Convert to Move Circular

---

##### Prerequisites

At least two targets, the via-point target and the end point target, must have been created.

A path containing at least the via-point target and the end point target, in correct order, must have been created.

---

##### Converting to Move Circular

- 1 In the **Paths&Targets** browser, expand the path node that contains the move instruction to be converted.
- 2 Select the move instruction that contains the via-point of the circular motion together with the succeeding move instruction, which will serve as the end point. You can select several instructions by holding down the **SHIFT** key while clicking the instructions.
- 3 Click **Convert to Move Circular**. The two selected move instructions will be converted to a circular move instruction, which includes the via-point and the end point.



##### Tip

To convert two move instructions to a circular motion, you can also select and right-click both move instructions at once and then click **Convert to Circular**.

#### 14.10 Copy / Apply Orientation

---

##### Copying and applying an orientation

- 1 In the browser, select the object or target from which to copy the orientation.
- 2 On the **Modify** menu, click **Copy Orientation**.
- 3 In the browser, select the object or target to which to apply the orientation.
- 4 On the **Modify** menu, click **Apply Orientation**. This can be performed on several targets or a group of selected targets.

## 14 Context menus

---

### 14.11 Detach

#### 14.11 Detach

---

##### Detaching an object

- 1 In the **Layout** browser, right-click the attached object (child) and then click **Detach**. The child will be detached from the parent and return to its position before the attachment.

---

**14.12 Execute Move Instruction**

---

**Prerequisites**

The move instruction must exist.

A virtual controller must be running for the robot with the move instruction.

---

**Executing a move instruction**

- 1 In the **Paths&Targets** browser, browse to the motion instruction to execute through the **Controller**, **Tasks** and **Paths** nodes.
- 2 Click **Execute move instruction**. The TCP of the active robot will move from the current location to the motion instruction according to the programmed motion properties. If the target for the motion instruction does not have a stored configuration, the robot will use the configuration nearest the current one.

#### 14.13 External Axis Interpolation

##### Prerequisites

You need to have a path selected and a robot with external axis configured.

##### Interpolating external axis

- 1 In the **Paths&Targets** browser, select a path, right-click and select **Interpolate External Axis**.

The Interpolate External Axis dialog box appears.

- 2 Select the mechanical unit from the **Mechanical Unit** drop-down list.
- 3 Select the axis to interpolate from the **Axis** drop-down list.
- 4 In the **Interpolation** drop-down list,

select...	to...
Constant	set a constant value for the axis in each robtarget. You can set the value from the <b>Value</b> drop-down list.
TCP Offset	calculate an axis value such that the <ul style="list-style-type: none"><li>• For a linear axis, robot base is translated the offset distance relative to the target along the axis direction.</li><li>• For a rotating axis, external axis value is calculated so that the angle between the TCP approach direction and the rotational axis zeroposition is kept constant at the offset angle.</li></ul>

- 5 Click **Apply**.

## 14.14 Graphic Appearance

### Overview

With the graphic appearance dialog box you set the graphic properties for an individual object. These settings override the generic settings made in the options dialog box. To launch the Graphic Appearance dialog box right-click a part in the browser and select Graphic Appearance from the context menu.

The right part contains a preview of the part and controls for setting the behavior and appearance of the preview. The user can work on the entire part, or on individual bodies or faces by selecting them in the preview.

### Graphic Appearance: The Material tab

The Material tab contains controls for setting material parameters or selecting a material from the list of user defined and predefined materials. It is also possible to save the current material to the user defined materials.

The Material tab contains two groups: Colors and Textures. The Color group contains parameters for controlling the color properties of an object. The Textures group contains the settings for textures. The color and texture setting together define a material. You can save your current settings as a new material for later re-use or apply an existing material.



The following options are available:

- **Apply material** : Provides a list of predefined material that can be applied to the object.
- **Save material** : Saves the specified combination of color and texture settings as a material.

### Color group

<b>Simple Color</b>	Click this color box to select another color for the object.
<b>Opacity</b>	Controls the transparency of the object.
The color boxes	Set the color of the object for different light situations here.
<b>Shininess</b>	Specify the reflectiveness of the object here.

### Textures group

<b>Base texture</b>	<p>Specifies the basic structure of the selected part. It is a standard 24-bit image displayed on a 3D surface.</p> <div>  <b>Note</b> </div> <p>Transparency of textures is provided only for .png images.</p>
<b>Specify texture size</b>	<p>Check this option to specify the texture size. When a material with a specified texture size is applied to a geometry, RobotStudio tries to adjust the texture coordinates so that the texture image covers the specified area.</p> <div>  <b>Note</b> </div> <p>This option works best on flat surfaces; on curved surfaces the size will be approximate.</p>

*Continues on next page*

## 14 Context menus

### 14.14 Graphic Appearance

*Continued*

<b>Blend Mode</b>	Specifies how the texture is combined with the specified object color.
<b>Environment map</b>	Provides a highly reflective appearance to the surface.
<b>Normal Map</b>	Specifies a texture that defines the bumpiness of the surface

#### Graphic Appearance: The Properties tab

The Properties tab contains controls for surface properties and texture coordinates.

##### Surface Properties

<b>Render both sides</b>	By selecting this option surfaces will be rendered regardless of orientation.
<b>Invert surface(s)</b>	Selected surfaces can be inverted by this button. This process is simplified by checking <b>Highlight inverted faces</b> , which will cause the backside of surfaces to be displayed in red.
<b>Generate new normals</b>	Generate new normals. If the surface normals of an imported geometry are of poor quality, they can be recreated by clicking this button.

##### Texture coordinates

<b>Swap u/v</b>	Click this button to swap the horizontal and vertical directions of the texture.
<b>Modify</b>	Select along which directions the commands listed below shall be applied. <b>u</b> is the horizontal axis of the texture. <b>v</b> is the vertical axis of the texture.
<b>Normalize</b>	Click this button to set the ratio between the dimensions of the object and the texture to 1.
<b>Flip</b>	Click this button to invert the coordinates along the selected axes. This is the same as mirroring around the other axis.
<b>Stretch</b>	Click this button to stretch the texture along the selected axes.
<b>Shrink</b>	Click this button to shrink the texture along the selected axes.
<b>Shift&lt;</b>	Click this button to move the texture along the selected axes.
<b>Shift&gt;</b>	Click this button to move the texture along the selected axes.

#### 14.15 Go to Visualization and Go to Declaration

##### Go to Visualization

The **Go to visualization** context menu command is available for targets in the RAPID editor. It takes you to the 3D graphical window to show you where the target is found.



##### Note

This command requires that the RAPID code has been synchronized to the station.

##### Go to Declaration

In the Paths & Targets browser, the **Go To Declaration** context menu command is available for targets. This command takes you back to the target in the RAPID editor.



##### Note

This command requires that the RAPID code has been synchronized to the virtual controller.

### 14.16 Interpolate Path

---

#### Reorienting targets in a path by interpolation

- 1 In the **Layout** browser or the graphics window, select the path with the targets to reorient.
- 2 Click **Interpolate Path.** to bring up a dialog box.
- 3 With the **Interpolate type** options, select whether to use **Linear** or **Absolute** interpolation.

Linear interpolation distributes the difference in orientation evenly, based on the targets positions along the length of the path. By contrast, absolute interpolation distributes the difference in orientation evenly, based on the targets' sequence in the path.

- 4 If using the **Select Start/End** option, select the start and end targets for the interpolation in the **Start target** and **End target** boxes, respectively.
- 5 Optionally, with the **Lock Axis** options, select an axis to lock.
- 6 Click **Apply**.

## 14.17 Invert

---

### Inverting the direction of a face

- 1 Right-click in the **Modeling** browser, point to **Filter** and make sure that both **Show Bodies** and **Show Faces** are selected.
- 2 In the **Modeling** browser, expand the node for the object and browse down to and select the face which direction you want to invert.
- 3 In the **Modeling** browser, expand the node for the object and browse down to and select the face whose direction you want to invert.
- 4 Click **Invert**. If the option backface culling is activated, the face will now shift from visible to not visible, or the other way around, depending on from which direction you view the face. If backface culling is deactivated, there will be no visible indication that the direction of the face has been inverted.

### 14.18 Jump to Target

---

#### Jumping to a target

- 1 In the **Paths&Targets** browser, browse to the target to jump to through the **Controller**, **Tasks** and **WorkObjects** nodes.
- 2 Click **Jump to target**.  
If the target has a valid configuration for the robot axes stored, the active TCP of the robot will immediately be positioned at the target. If no valid configuration is stored, the **Select Robot configuration** dialog box is displayed.
- 3 In the **Select Robot Configuration** dialog box, select a suitable configuration solution and click **Apply**. The selected configuration is now stored with the target.



#### Note

You can deactivate the configuration check when jumping to targets. The robot will then use the configuration solution closest to the current one when reaching the target. For more information, see [Options on page 195](#).

---

## 14.19 Linked Geometry

---

### Overview

The Linked Geometry feature allows you to load geometry from a shared repository. If the source file is updated, then the station will be updated with a single click.

---

### Adding Link

You can add a link to a geometry in two ways:

- 1 In the **Home** tab, click **Import Geometry** to open a dialog box.  
Select the option **Link to Geometry**.
- 2 In the **Layout** browser, right-click an existing part in the station and select **Add Link**.

A dialog box opens where you can select the CAD file to be linked.

---

### Editing Link

To edit an existing link:

- 1 In the **Layout** browser, right-click an existing part in the station.
- 2 Select the option **Link to Geometry** and click **Edit Link**.

---

### Deleting Link

To delete an existing link:

- 1 In the **Layout** browser, right-click an existing part in the station.
- 2 Select the option **Link to Geometry** and click **Delete Link**.

---

### Updating Linked Geometry

To update a linked geometry:

- 1 In the **Layout** browser, right-click an existing part in the station, component group or the station.
- 2 Select the option **Link to Geometry** and click **Update Linked Geometry**.

The update result is displayed in the output window.



#### Note

When you select a component group or a station, all linked geometries within the group or station is updated. If the timestamp on the file is newer than the timestamp stored in the station, all corresponding parts will be updated from the source location.

## 14 Context menus

---

### 14.20 Modify Library Component

#### 14.20 Modify Library Component

---

##### Modifying a library component

- 1 In the **Layout** browser, select the library you wish to modify.
- 2 Click **Disconnect Library**.
- 3 Select the library and then make any modifications to it.
- 4 Select the modified library, and then click **Save As Library**.

## 14.21 Mechanism Joint Jog

### Jogging the joints of a robot

- 1 In the **Layout** browser, select the robot.
- 2 Click **Mechanism Joint Jog** to bring up a dialog box.
- 3 Each row in the **Jog Joints** dialog box represents a joint of the robot. Jog the joints either by clicking and dragging the bar at each row, or by using the arrows to the right of each row.  
Set the length of each step in the **Step** box.

### The Mechanism Joint Jog dialog box

<b>Joint</b>	Move the joints of the objects by dragging the slider on the row corresponding to each joint. Alternatively, click the buttons to the right of the row, or type a value.
<b>Cfg</b>	The current configuration value.
<b>TCP</b>	The current position of the TCP.
<b>Step</b>	Specify the length of the joint movements for each click of the buttons to the right of each joint row.
<b>External Axis</b>	If the robot uses external axes, you can select an axis to jog from this list. The external axes must belong to the same task as the object you are jogging for occurring in this list. If no external axes are present in the same task, this list is not available.
<b>Lock TCP</b>	Select this check box to reposition the robot according to the jogging of the external axis. For track-external axes the robot will reposition so that the TCP is locked relative to the world coordinate system. For positioner-external axes the robot will reposition so that the position of the robot's TCP is locked relative to the attachment point of the positioner. The robot will move with the positioner the same way as when using multi-robot jog. If no external axes are present in the same task, this check box is not available.
<b>External axes joint</b>	Move the joint of the external axes by dragging the slider on the row corresponding to each joint. Alternatively, click the buttons to the right of the row, or type a value. If no external axes are present in the same task, this check box is not available.

### Jogging a conveyor

- 1 Create **Empty Path**. See [Empty Path on page 224](#).
- 2 In the **Layout** browser, select the conveyor.
- 3 Right-click **Conveyor Mechanism** and select **Mechanical Joint Jog**.  
The Joint Jog dialog box appears.
- 4 Jog the conveyor by moving the slider and click **Teach Instruction**.  
A move instruction is added to the path.

*Continues on next page*

## 14 Context menus

---

### 14.21 Mechanism Joint Jog

*Continued*



#### Note

When you jog the conveyor mechanism, objects on the conveyor are also moved.

- If you jog the conveyor mechanism beyond the maximum distance, the workobject will be dropped.
- If you jog the conveyor mechanism beyond the zero position, the workobject that belongs to the first part is attached to the conveyor attachment point.

If the workobject is dropped in Teach Mode, you can jog the conveyor backwards to connect it again.

## 14.22 Mechanism Linear Jog

---

### Jogging the TCP of a robot by using the Linear Jog dialog box

- 1 In the **Layout** browser, select the robot.
- 2 Click **Mechanism Linear Jog** to bring up a dialog box.
- 3 Each row in the **Linear Jog** dialog box represents a direction or rotation for the TCP. Jog the TCP along the preferred direction or rotation, either by clicking and dragging the bar at each row, or by using the arrows to the right of each row.
- 4 From the **Reference** list, you can select the coordinate system that you want to jog the robot relative to.
- 5 In the **Step** box, specify the step movement per deg/rad.

## 14 Context menus

### 14.23 Mirror Path

#### 14.23 Mirror Path

##### The Mirror Path dialog box

<b>Duplicate</b>	Select this option to keep the existing path when mirroring.
<b>Replace</b>	Select this option to remove the existing path after mirroring.
<b>X-Y, X-Z and Y-Z</b>	Select the plane to mirror the path around with these options. The plane is defined by the selected axes and position of the reference coordinate system selected below.
<b>Reference</b>	Select the frame or coordinate system to define the mirror plane in. To use another frame than any of the predefined ones, select <b>Select Frame</b> from the list and specify the frame in the box below.
<b>Select Frame</b>	If <b>Select Frame</b> is used as Reference frame, specify the frame to use here by first clicking in the box and then selecting the frame from the graphics window or the <b>Layout</b> browser.
<b>Flip axis X/YZ</b>	Select one of these options to mirror the orientation of the targets. When any of these are selected, the robot will approach the targets in a mirrored way.  The axis you select will change the most for achieving the mirrored orientation, while the other one will be kept as near to its current direction as possible.  The axis that is set to the robot's approach vector cannot be selected.
<b>Keep orientation</b>	Select this option to keep the orientation of the targets. When selected, the robot will go to the mirrored position, but approach the target from the same direction as for the original target.
<b>Mirror Robot Configuration</b>	Select this option to also mirror the robot axis configuration for the targets. Selecting this option will mirror the robot's motions completely. For using this option, the following conditions must be met: <ul style="list-style-type: none"><li>• The Reference frame must be set to <i>Baseframe</i>.</li><li>• The Mirror plane must be set to X-Z.</li><li>• The tool of each move instruction must have its TCP in the X-Z plane of <i>tool0</i>.</li><li>• All targets in the path must have robot axis configuration set.</li><li>• The virtual controller must be running.</li></ul>
<b>More / Less</b>	Click this button to show or hide the commands for naming and location of generated targets and paths.
<b>New path name</b>	Specify the name of the path that will be generated by the mirroring here.
<b>Target prefix</b>	Specify a prefix for the targets that will be generated by the mirroring here.
<b>Receiving robot</b>	Specify the robot task in which the new targets and path shall be created.
<b>Receiving work object</b>	Specify the work object in which the new targets shall be created.

## 14.24 Mirror

### Mirroring a part

- 1 In the **Layout** browser, select the part to mirror and right-click.
- 2 Select **Mirror** and then click one of the following options from the context menu:

select...	to create a new part..
Mirror YZ	around YZ plane
Mirror ZX	around ZX plane
Mirror XY	around XY plane



#### Note

The mirror feature is applicable only to objects of type body and part that contains geometry. Parts and bodies that are imported without geometry cannot be mirrored. See [Libraries, geometries and CAD files on page 37](#).

For information on mirroring a path, see [Mirror Path on page 472](#).

## 14.25 Modify Curve

---

### Overview

The following table provides an overview of the generic content available in the Modify Curve dialog box.

<b>Extend</b>	Extend a curve at any vertex with a straight line in the direction of the curve's tangent.
<b>Join</b>	Join two or more curves into one. The original curves will be deleted when joining curves.
<b>Project</b>	Project a curve onto a surface or a body, creating a new curve on the target part.
<b>Reverse</b>	Reverse the direction of curves.
<b>Split</b>	Split a curve in two bodies. Only open curves can be split.
<b>Trim</b>	Cut a segment of a curve between intersection or end points.

---

### Extending a curve with a straight line in the direction of the curve's tangent

- 1 Click the **Modify curve** and then select **Extend Curve** from the drop down menu.  
The **Extend Curve** menu opens.
- 2 Click the curve you want to extend.
- 3 In the **From start vertex** and **From end vertex** boxes, type or select the required length you want to extend the curve. In the graphics window a yellow line displays a preview of the extension.
- 4 Click **Apply**.

---

### Joining curves

- 1 Click the **Modify curve** and then select **Join Curves** from the drop down menu.  
The **Join Curves** menu opens.
- 2 Click the curves to join in the graphics window. The curves may be either intersecting or adjacent to be joined.  
The **Selected curves** list displays the curves that will be joined. To remove a curve from the list, select the list entry and press the DEL key.
- 3 In the **Tolerance** box, enter a value. Adjacent curves whose end points lie within the tolerance will be valid for the operation.
- 4 Click **Apply**.

---

### Projecting curves on a surface

- 1 Click the **Modify curve** and then select **Project Curve** from the drop down menu.  
The **Project Curves** menu opens.
- 2 Click the curves to project in the graphics window.  
Note that when you rest the pointer over the curve, the projection direction is displayed. The project direction is always the negative Z direction of the

*Continues on next page*

User Coordinate System. To change the projection direction, create a new frame with the desired orientation and set it as user coordinate system.

The **Selected curves** list displays the curves that will be projected. To remove a curve from the list, select the list entry and press the DEL key.

- 3 Click in the **Target bodies** list and then click the bodies to project on in the graphics window. The bodies must be in the projection direction and be big enough to cover the projected curves.

To remove a bodies from the list, select the list entry and press the DEL key.

- 4 Click **Apply**. A new curve will now be created in a new part, wrapped around the surface of the selected bodies.

---

### Reversing curves

- 1 Click the **Modify Curve** and then select **Reverse Curve** from the drop down menu.

The **Project Curves** menu opens.

- 2 Click the curves to reverse in the graphics window.

Note that when you rest the pointer over a curve, the current direction of the curve is displayed by yellow arrows.

The **Selected curves** list displays the curves that will be reversed. To remove a curve from the list, select the list entry and press the DEL key.

- 3 Click **Apply**. The curves will now be reversed.

---

### Splitting a curve

- 1 Click the **Modify Curve** and then select **Split Curve** from the menu to open the tool.

- 2 Click the curve at the point to split at. Only open curves can be split.

Note that when you rest the pointer over the curve, point of the split is highlighted. This point is affected by the current snap mode setting.

- 3 Click **Apply**. The curve will now be split to two separate curves in the same part.

---

### Trimming a curve

- 1 Click the **Modify Curve** and then select **Trim Curve** in the menu to open the tool.

- 2 Click the curve segment to trim.

Trim only works on single curves with intersection points. If you want to trim a curve that intersects with another curve, first join the two curves.

- 3 Click **Apply**. The selected part of the curve will now be removed.

## 14.26 Modify External Axis

### Modifying external axis positions in targets

- 1 Select the targets (one or several) you want to modify, either in the **Layout** browser or in the graphics window. If you select several targets, the values you specify will be applied to all selected targets.
- 2 Click **Modify External Axis** to bring up a dialog box.
- 3 Edit the values of the axis by performing any of the following:

Action	Description
Type a new position value for an axis	In the <b>Eax</b> column, select the value of the external axis you want to edit, and enter the new value.
Jog the axis to the new position	Use the arrow buttons to the left of the Joint Values column for jogging the axis. Then click the right arrow button between the Joint values column and the Eax column to transfer the current joint value to the Eax value.

- 4 Click **Apply**.

### The Modify External Axis dialog box

<	Jog the joint of the external axis corresponding to each row by clicking the < button.
>	Jog the joint of the external axis corresponding to each row by clicking the > button.
value box	Enter the axis value for the corresponding joint of the external axis in the value box.
<-	With the arrow left button, you transfer the value from the <b>Eax</b> box to the corresponding value box.
->	With the arrow right button, you transfer the value from the value box to the corresponding <b>Eax</b> box.
<b>Eax</b>	Specify the value of the corresponding joint of the external axis.

## 14.27 Modify Instruction

### Modifying an instruction

- 1 In the **Paths&Targets** browser, select the instruction you want to modify. If you want to apply the same properties to several instructions, press the **CTRL** key and select them.
- 2 Click **Modify Instruction** to bring up a dialog box.
- 3 For move instructions, select joint or linear motion in the **Motion type** list.
- 4 In the **Instruction Argument** group, modify the values for the instruction.  
For details about each argument, see the selected instruction in the *RAPID Reference Manual*. For an overview of the arguments for move instructions, see below.
- 5 When you have finished modifying, click **Apply**.

### Arguments for move instructions

The table below is an overview of common arguments for move instructions. For detailed information about the arguments, see the selected instruction in the *RAPID Reference Manual*.

To set the	Use
subsequent instructions to be executed at once.	<b>\Conc</b>
destination target for the instruction target.	<b>ToPoint</b>
speed for the tool center point, the tool reorientation and external axes.	<b>Speed</b>
velocity of the TCP in mm/s directly in the instruction (it will be substituted for the corresponding speed data).	<b>\V</b>
total time in seconds during which the robot moves (it will be substituted for the corresponding speed data).	<b>\T</b>
size of the generated corner path.	<b>Zone</b>
position accuracy of the robot TCP directly in the instruction (the length of the path will be substituted for the corresponding zone specified in the zone data).	<b>\Z</b>
tool used for the movement (the TCP of this tool will be positioned at the destination target).	<b>\Tool</b>
workobject to which the robot position in the instruction belongs.	<b>\Wobj</b>

## 14 Context menus

---

### 14.28 Modify Mechanism

### 14.28 Modify Mechanism

---

#### The Modify Mechanism dialog box

See [The Modify Mechanism dialog box on page 322](#).

## 14.29 Modify Tooldata

---

### Modifying tooldata

- 1 In the **Layout** browser, select the tooldata you want to modify.
- 2 Click **Modify Tooldata** to bring up a dialog box.
- 3 In the **Misc Data** group:
  - Modify the **Name** of the tool.
  - Select if the tool is to be held by the robot in the **Robot holds tool** list.
- 4 In the **Tool Frame** group:
  - Modify the **Position x, y, z** of the tool.
  - Modify the **Rotation rx, ry, rz** of the tool.
- 5 In the **Load Data** group:
  - Enter a new **Weight** for the tool.
  - Modify the **Center of gravity** for the tool.
  - Modify the **Inertia** for the tool.
- 6 In the **Sync Properties** group:
  - In the **Storage type** list, select **PERS** or **TASK PERS**. Select **TASK PERS** if you intend to use the tooldata in MultiMove mode.
  - In the **Module** list, modify the module in which to declare the tooldata.
- 7 Click **Apply**.

### 14.30 Modify Workobject

---

#### Modifying a workobject

- 1 In the **Layout** browser, select the workobject you want to modify.
- 2 Click **Modify Workobject.** to bring up a dialog box.
- 3 In the **Misc Data** group, modify the values for the workobject:
  - Enter a **Name** for the workobject.
  - In the **Robot holds workobject** list, select **True** or **False**. If you select **True**, the robot will move the work piece instead of the tool.
  - In the **Moved by mechanical unit** list, select the mechanical unit with which the robot movements are coordinated. This setting is only valid when **Programmed** has been set to **False**.
  - In the **Programmed** list, select **True** or **False**. **True** means that the workobject will use a fixed coordinate system, and **False** that a movable coordinate system (that is, coordinated external axes) will be used.
- 4 In the **User Frame** group, do one of the following:
  - Modify the user frame by entering values for the **Position x, y, z** and the **Rotation rx, ry, rz** for the workobject. Click in one of these boxes, and then click the position in the graphics window to transfer the values.
  - Modify the user frame by using the **Frame by points** dialog box, see [Frame from Three Points on page 214](#) .
- 5 In the **Object Frame** group, do one of the following:
  - Modify the object frame by selecting values for **Position x, y, z** and **Rotation rx, ry, rz** for the workobject.
  - Modify the object frame by using the **Frame by points** dialog box.
- 6 In the **Sync Properties** group, modify the values for the workobject:
  - In the **Storage type** list, select **PERS** or **TASK PERS**. Select **TASK PERS** if you intend to use the workobject in MultiMove mode.
  - In the **Module** list, select the module in which to declare the workobject.
- 7 Click **Apply**.



#### Note

If you change the position of a workobject that is used in a program, you have to synchronize the affected paths to the virtual controller; otherwise, the program will not be updated.

---

## **14.31 Move Along Path**

---

### **Prerequisites**

At least one path must have been created in the station.

A virtual controller must be running for the robot to move along the path.

---

### **Moving along a path**

- 1 In the **Paths&Targets** browser, select the path to move along.
- 2 Click **Move along path**. In the graphics window, the robot will move along the path.

## 14 Context menus

---

### 14.32 Move to Pose

### 14.32 Move to Pose

---

#### Prerequisites

At least one joint position must be defined.

Only one mechanism may be selected at a time.

---

#### Moving to a pose

- 1 In the **Layout** browser, select one mechanism to move.
- 2 Click **Move to Pose** and then click one of the available poses. In the graphics window, the mechanism will move to the pose.

### 14.33 Offset Position

#### Offset the position an item

- 1 Right-click the item you want to move.
- 2 Click **Offset Position** to bring up the **Offset Position** dialog box.
- 3 In the dialog box, select the reference coordinate system you want to use:

If you want to move the item	Select
relative to its own coordinate system	<b>Local</b>
relative to the coordinate system of its parent	<b>Parent</b>
relative to the coordinate system of the station	<b>World</b>
relative to a user-defined coordinate system	<b>UCS</b>
relative to a target reference frame	<b>Target Reference Frame</b> This option is available only for targets.

- 4 In the **Translation X, Y, Z** boxes, either type the offset, or select it by first clicking in one of the value boxes and then clicking the point in the graphics window.
- 5 Specify the **Rotation** for the item.
- 6 Click **Apply**.

## 14 Context menus

### 14.34 Place

### 14.34 Place

#### Placing an item

- 1 Select the item you want move.
- 2 Click **Place** and then click one of the commands to bring up a dialog box.

If you want to move the item	Choose
from one position to another without affecting the orientation of the object. Select the axes to be affected.	<b>One Point</b>
according to the relationship between a start and a finish line. The object will move to match the first point, then it will rotate to match the second point.	<b>Two Points</b>
according to the relationship between a start plane and a finish plane. The object will move to match the first point, then it will rotate to match the third point.	<b>Three Points</b>
from one position to a target or frame position and simultaneously change the orientation of the object according to the frame orientation. The position of the object changes according the orientation of the to-point coordinate system.	<b>Frame</b>
from one frame of reference to another	<b>Two Frames</b>

- 3 Set the reference coordinate system you want to use.
- 4 Click the points in the graphics window to transfer values to the from-point boxes to the to-point boxes. For detailed information, see the tables below.
- 5 Click **Apply**.

#### The Place Object by One Point dialog box

<b>Reference</b>	Select the reference coordinate system to which all positions or points will be related.
<b>Primary Point - From</b>	Click in one of these boxes, and then click the primary point in the graphics window to transfer the values to the <b>Primary Point - From</b> boxes.
<b>Primary Point - To</b>	Click in one of these boxes, and then click the primary point in the graphics window to transfer the values to the <b>Primary Point - To</b> boxes.
<b>Translate along these axes</b>	Select whether the translation is to be performed along the X, Y or Z axis, or several of the axes.

#### The Place Object by Two Points dialog box

<b>Reference</b>	Select the reference coordinate system to which all positions or points will be related.
------------------	--

*Continues on next page*

<b>Primary Point - From</b>	Click in one of these boxes, and then click the primary point in the graphics window to transfer the values to the <b>Primary Point - From</b> boxes.
<b>Primary Point - To</b>	Click in one of these boxes, and then click the primary point in the graphics window to transfer the values to the <b>Primary Point - To</b> boxes.
<b>Point on X-Axis - From</b>	Click in one of these boxes, and then click the point on the x axis in the graphics window to transfer the values to the <b>Point on X-Axis - From</b> boxes.
<b>Point on X-Axis - To</b>	Click in one of these boxes, and then click the point on the x axis in the graphics window to transfer the values to the <b>Point on X-Axis - To</b> boxes.
<b>Translate along these axes</b>	Select whether the translation is to be performed along the X, Y or Z axis, or several of the axes.

#### The Place an Object by Three Points dialog box

<b>Reference</b>	Select the reference coordinate system to which all positions or points will be related.
<b>Primary Point - From</b>	Click in one of these boxes, and then click the primary point in the graphics window to transfer the values to the <b>Primary Point - From</b> boxes.
<b>Primary Point - To</b>	Click in one of these boxes, and then click the primary point in the graphics window to transfer the values to the <b>Primary Point - To</b> boxes.
<b>Point on X-Axis - From</b>	Click in one of these boxes, and then click the point on the x axis in the graphics window to transfer the values to the <b>Point on X-Axis - From</b> boxes.
<b>Point on X-Axis - To</b>	Click in one of these boxes, and then click the point on the x axis in the graphics window to transfer the values to the <b>Point on X-Axis - To</b> boxes.
<b>Point on Y-Axis - From</b>	Click in one of these boxes, and then click the point on the y axis in the graphics window to transfer the values to the <b>Point on Y-Axis - From</b> boxes.
<b>Point on Y-Axis - To</b>	Click in one of these boxes, and then click the point on the y axis in the graphics window to transfer the values to the <b>Point on Y-Axis - To</b> boxes.
<b>Translate along these axes</b>	Select whether the translation is to be performed along the X, Y or Z axis, or several of the axes.

#### The Place Object with Frame dialog box

<b>Select Frame</b>	Specify the name of the frame with which you want to place the object.
---------------------	--

#### The Place by Two Frames dialog box

<b>From</b>	Select the frame object (For example, Target, Workobject, Tooldata or Frame) from this dropdown list to set the <b>From</b> point of moving the object.
<b>To</b>	Select any of the frame object (For example, Target, Workobject, Tooldata or Frame) from this dropdown list to set the <b>To</b> point of moving the object.

#### 14.35 Protected Smart Component

---

You can protect a Smart Component from being edited. To protect the smart component, right-click the smart component, and then click **Protected**. You can also optionally specify a password that will be required to unlock the component for edits.

For more information on protected smart components, see *Protecting a Smart Component from edits* in the section [Smart Component on page 264](#).



#### Note

Protecting a Smart Component in this manner is a way of hiding complexity, and is not for providing security or securely protecting it.

### 14.36 Remove Unused Targets

---

#### Removing unused targets

- 1 In the **Paths&Targets** browser, select either the *Controller* node or the *Task* node from which you wish to remove the unused targets, and then click **Remove Unused Targets**.
- 2 To the question **Are you sure you want to remove unused targets?**, answer **Yes**. All targets that not are used by any move instructions are now removed.

#### 14.37 Rename Targets

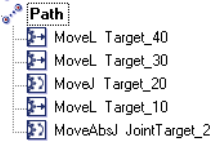
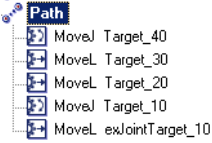
---

##### Renaming targets

- 1 In the **Paths & Targets** browser, select the targets to rename.  
To rename all targets in one or several paths, select the paths that contain the targets.
- 2 Click **Rename targets** to bring up a dialog box.
- 3 In the **Target Prefix** box, enter a text string to precede the target numbers.
- 4 Optionally, in the **Increment** box and the **Start with** box, change the numbering series for the target names.
- 5 Optionally, in the **Target Suffix** box, enter a text string to follow the target numbers.
- 6 Click **Apply**.

## 14.38 Reverse Path

### The commands

Simple	<p>Here you reverse only the target sequence. The new path will keep the move instruction for each path segment and just reverse the programed positions.</p>  <p>xx050046</p> <p>Note that move instructions are not changed, just the targets. Even the MoveAbsJ instruction to the joint target is preserved, but placed last.</p>
Advanced	<p>Both target sequence and move instructions are reversed in a way that corresponds to recording the robot movements and playing the movie backwards. For example, if the robot used a linear motion to move from a target, it will use a linear motion to move to the target after the reversal.</p>  <p>xx050047</p> <p>Note that the move instructions have changed together with the targets. For example, in the original path, a joint motion was used to reach target 20 and a linear motion to leave it. After the reversal there is a linear motion to the target and joint motion from it.</p> <p>Also, note that the jointtarget has been converted into an ordinary target; otherwise, it would not be possible to program a linear motion to that position.</p>

#### 14.39 Rotate

---

##### Rotating an item

- 1 Select the item you want to rotate.
- 2 Click **Rotate** to bring up a dialog box.
- 3 Select the reference coordinate system you want to use:

If you want to move the item	Select
absolute in the coordinate system of the station	<b>World</b>
relative to the coordinate system of its parent	<b>Parent</b>
relative to its own coordinate system	<b>Local</b>
relative to the user-defined system	<b>UCS</b>
relative to an axis defined by two points	<b>User defined axis</b>
relative to a target reference frame Note that this option is available only for targets.	<b>Target Reference Frame</b>

- 4 Specify the rotation of the item in the **Rotate around x, y, z** by first clicking in one of the boxes, and then click the center position in the graphics window to transfer the values.
- 5 If you have selected the coordinate system **User defined axis**, specify the **Axis start point x, y, z** and the **Axis end point x, y, z**.
- 6 Specify the **Rotation** of the item and the axis around which the rotation is to occur.
- 7 Click **Apply**.

## 14.40 Rotate Path

### Rotating a paths

- 1 In the **Layout** browser or the graphics window, select the paths to rotate.
- 2 Click **Rotate path** to bring up a dialog box.
- 3 In the **Reference frame** list, select the frame to rotate the paths around.

Select	To
World	rotate around the station's world coordinate system
Baseframe	rotate around the robot's baseframe
UCS	rotate around a frame or target that previously has been set to User Coordinate System.
Select Frame	rotate around an existing target or frame other than the listed ones. When using <b>Select Frame</b> , specify the frame to rotate around further down.

- 4 If **Selected frame** was selected in the **Reference frame** list, specify a frame or target in the text box by clicking in the box and then selecting the frame in the graphics window.
- 5 With the **Rotation axis** options, select the axis of the frame to rotate around.
- 6 In the **Rotation angle** box, enter the rotation.
- 7 Click **Apply**.

#### 14.41 Set Local Origin

---

##### Setting the origin of the local coordinate system

- 1 If the object you want modify is a library component, first disconnect it from the library.
- 2 In the **Layout** browser or the graphics window, select the part to modify.
- 3 Click **Set Local Origin** to bring up a dialog box.
- 4 In the **Set Local Origin** dialog box, select the reference coordinate system you want to use:

If you want to move	Select
relative to the part's current local coordinate system	<b>Local</b>
relative to the coordinate system of its parent	<b>Parent</b>
absolute in the coordinate system of the station	<b>World</b>
relative to a user-defined coordinate system	<b>UCS</b>

- 5 In the **Position X, Y, Z** boxes, either type the new position or, select it by first clicking in one of the value boxes and then clicking the point in the graphics window.
- 6 Type the **Orientation**.
- 7 Click **Apply**.

---

## 14.42 Set Normal to Surface

---

### Setting the target orientation normal to a surface

- 1 In the **Paths & Targets** browser, select the target to modify.
- 2 Click **Set Normal To Surface** to bring up a dialog box.
- 3 On the **Selection Level** toolbar, set the selection level.
  - To align the target to a specific surface, set the selection level to **surface**.
  - To align the target to a specific point at the surface, set the selection level to **part**.
- 4 In the graphics window, click the reference surface. This will transfer the name of the part or surface to the **Surface** box.
- 5 In the **Approach Direction**, click the button for the axis to be used as the approach direction.
- 6 To set the distance between the surface and the target in the approach direction, specify an **Offset** value.
- 7 Click **Apply**.

#### 14.43 Set Position

---

##### Positioning an item

- 1 Right-click the item you want to move.
- 2 Click **Set Position** to bring up the **Set Position** dialog box.
- 3 In the dialog box, select the reference coordinate system you want to use:

If you want to move the item	Select
relative to its own coordinate system	<b>Local</b>
relative to the coordinate system of its parent	<b>Parent</b>
absolute in the coordinate system of the station	<b>World</b>
relative to a user-defined coordinate system	<b>UCS</b>
relative to a target reference frame	<b>Target Reference Frame</b> This option is available only for targets.

- 4 In the **Position X, Y, Z** boxes, either type the new position, or select it by first clicking in one of the value boxes and then clicking the point in the graphics window.
- 5 Specify the **Orientation** for the item.
- 6 Click **Apply**.

## 14.44 Tool Compensation

### Offsetting a path to compensate for tool radius

- 1 In the **Paths&Targets** browser or the graphics window, select the path.
- 2 Click **Tool Compensation** to bring up a dialog box.
- 3 In the **Distance** box, enter the size of the compensation (normally, the tool radius).
- 4 Using the **Direction** options, select whether the new path shall be on the left or the right side of the current path.
- 5 Click **Apply**.



#### Note

Tool Compensation function supports only planar paths. In planar paths, the targets are in the same plane. User will be notified when a selected path is non-planar.

#### 14.45 Translate Path

##### Translating a path

- 1 In the **Paths&Targets** browser or the graphics window, select the paths to translate.
- 2 Click **Translate path** to bring up a dialog box.
- 3 In the **Reference frame** list, select the coordinate system to use as reference for moving the paths.

Select	To
World	move relative to the origin of the world coordinate system
Base Frame	move relative to the origin of the robot's baseframe
UCS	move relative to the origin of a frame or target that previously has been set to User Coordinate System.
Select Frame	move relative to the origin of an existing target or frame other than the listed ones. When using <b>Select Frame</b> , specify the frame to use further down.
Point to Point	move the path from one point to another without specifying any coordinate system.

- 4 If **Select frame** was selected in the **Reference frame** list, specify a frame or target in the text box by clicking in the box and then selecting the frame from the graphics window.
- 5 In the **Translation vector** box, specify the distance to move the path along the X, Y and Z axes of the reference frame.

Translation vector is applicable only if a reference frame is used. If **Point to Point** is used as reference, specify the start and end points for the translation, instead. To do this, click in one of the boxes for the point to specify and then select the point in the graphics window, or type the coordinates of the point.

- 6 Click **Apply**.

#### 14.46 View Robot at Target

---

##### Viewing a robot at a target

- 1 Click **View Robot at Target**.
- 2 Select a target, either in the **Paths&Targets** browser or in the graphics window.
- 3 The robot will be shown at each selected target whenever a target is selected. By stepping through the targets in the browser, it will be easy to see how the position of the robot changes.
- 4 To turn the function off, click the command again.

#### 14.47 View Tool at Target

---

##### Viewing a tool at a target

- 1 Click **View Tool at Target** and select the tool you want to view at the target.
- 2 Select a target, either in the **Paths&Targets** browser or in the graphics window. You can also multiselect targets to show several copies of the tool. A copy of the tool will be shown at the selected target. By stepping through the targets in the browser, it is easy to see how the tool orientation changes.
- 3 To turn the function off, click the command and clear the check box.

## 15 ScreenMaker tab

### 15.1 Introduction to ScreenMaker

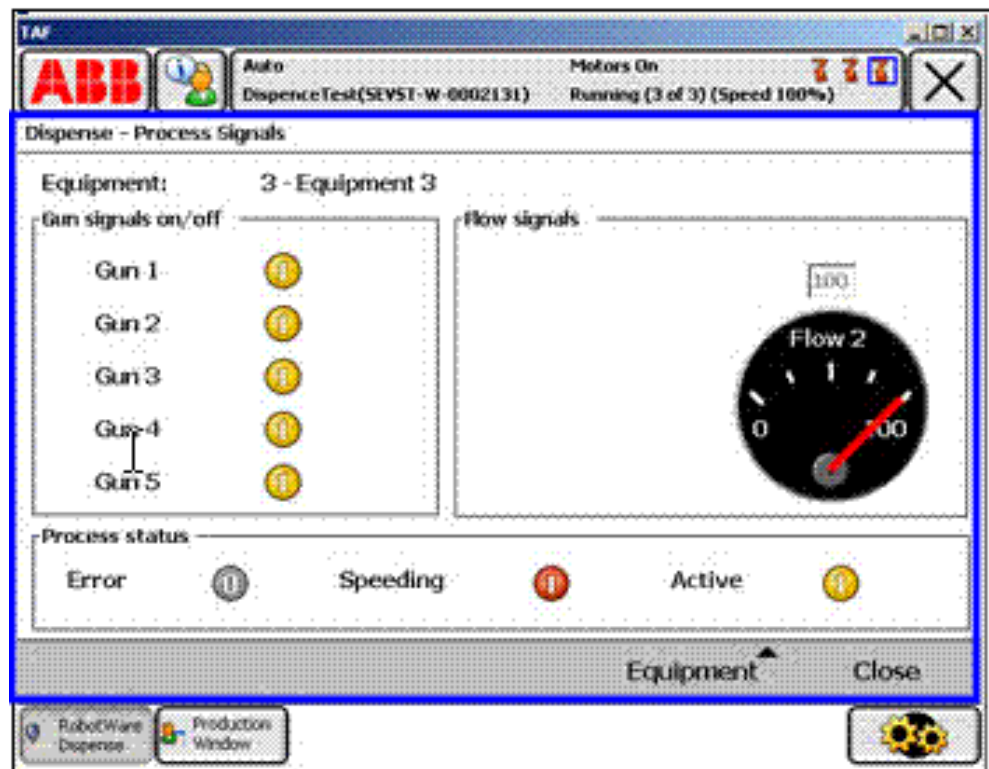
#### What is ScreenMaker?

ScreenMaker is a tool in RobotStudio for developing custom screens. It is used to create customized FlexPendant GUIs without the need to learn Visual Studio development environment and .NET programming.

#### Why use ScreenMaker?

A customized operator interface on the factory floor is the key to a simple robotic system. A well-designed custom operator interface presents the right amount of information at the right time and in the right format to the user.

#### GUI concepts



xx0800000226

A GUI makes it easier for people to work with industrial robots by presenting a visual front end to the internal workings of a robotic system. For FlexPendant GUI applications, the graphical interface consists of a number of screens, each occupying the user window area (the blue box in the figure above) of the FlexPendant touch screen. A FlexPendant screen is then composed of a number of smaller graphical components in a design layout. Typical controls (sometimes referred as widgets or graphic components) include buttons, menus, images, and text fields.

*Continues on next page*

## 15 ScreenMaker tab

---

### 15.1 Introduction to ScreenMaker

*Continued*

A user interacts with a GUI application by:

- Clicking a button
- Selecting from a menu
- Typing a text in a text box
- Scrolling

An action such as clicking a button is called an event. Whenever an action is performed, an event is sent to the GUI application. The exact content of an event is solely dependent on the graphic component itself. Different components trigger different types of events. The GUI application responds to the events in the order generated by the user. This is called event-driven programming, since the main flow of a GUI application is dictated by events rather than being sequential from start to finish. Due to the unpredictability of the user's actions, one major task in developing a robust GUI application is to ensure that it works correctly no matter what the user does. Of course, a GUI application can, and actually does, ignore events that are irrelevant.

The event handler holds sets of actions to be executed after an event occurs. Similar to trap routines in the RAPID program, the event handler allows the implementation of application-specific logic, such as running a RAPID program, opening a gripper, processing logic or calculating.

In summary, from a developer's point of view, a GUI consists of at least two parts:

- *the view part*: layout and configuration of controls
- *the process part*: event handlers that respond to events

Modern GUI development environments often provide a form designer, a (What You See Is What You Get ) WYSIWYG tool to allow the user to select, position and configure the widgets. As for event handlers, typically the developer must use a special programming language recommended by the development environment.

*Continues on next page*

## FlexPendant concepts



xx0800000228

Running Windows CE, the ABB FlexPendant has limited CPU power and memory compared to a PC. A custom GUI application must therefore be placed in the designated folders on the controller hard drive before being loaded. Once loaded, it can be found in the ABB menu as seen in the figure above. Clicking the menu item will launch the GUI application.

As the robot controller is the one actually controlling the robot and its peripheral equipment by executing a RAPID program, a GUI application needs to communicate with the RAPID program server in order to read and write RAPID variables and set or reset I/O signals.

It is essential for RAPID programmers to understand that there are two different levels controlling a work cell: an event-driven GUI application running on the FlexPendant, and a sequential RAPID program running in the controller. These reside on different CPUs and use different operating systems, so communication and coordination are important and must be carefully designed.

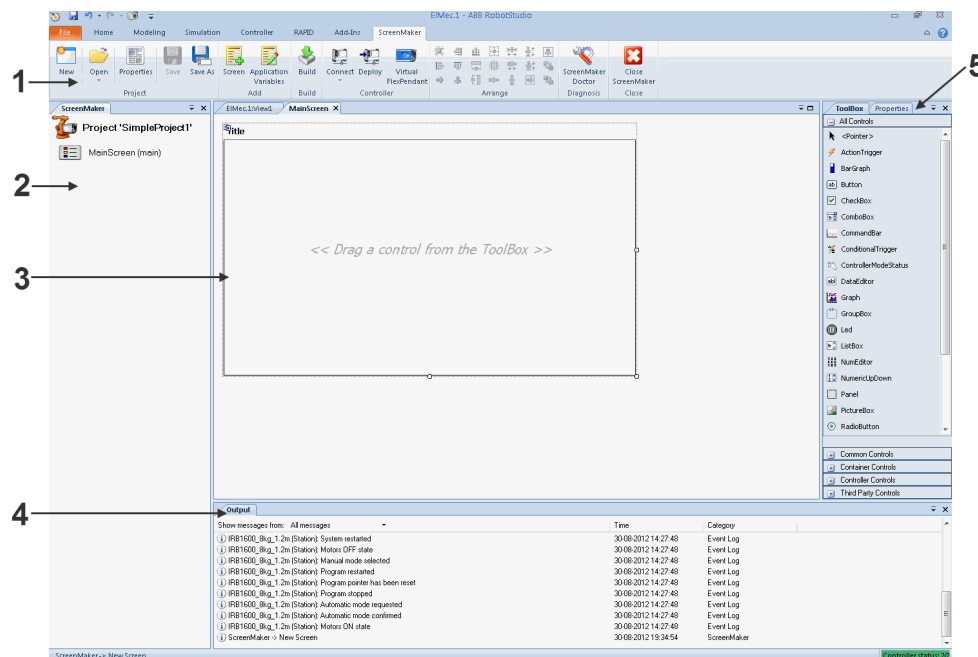
## 15 ScreenMaker tab

### 15.2 Development environment

### 15.2 Development environment

#### Overview

This section presents an overview of the ScreenMaker development environment for creating user screens.



en0900000584

	Parts	Description
1	Ribbon	Displays group of icons organized in a logical sequence of functions. See <a href="#">Ribbon on page 503</a> .
2	Project explorer	Shows the active screen project and lists the screens that are defined in the project. For more information, see <a href="#">Managing projects on page 507</a> .
3	Design area	Layout to design the screen with the available controls.
4	Output window	Displays information about the events that occur during ScreenMaker development.
5	ToolBox / Properties	Displays a list of available controls. For more information, see <a href="#">ToolBox on page 504</a> . Contains the available properties and events of the selected control(s). The value of the properties can either be a fixed value or a link to an IRC5 data or an Application Variable. For more information, see <a href="#">Properties window on page 506</a> .

Continues on next page

---

**Ribbon**

The ScreenMaker tab contains groups of commands organized in a logical sequence of functions that facilitates the user in managing ScreenMaker projects. The tab consists of the following groups:

Group	Functions used for
Project	Managing a ScreenMaker project. See <a href="#">Managing projects on page 507</a> .
Add	Adding screen and application variables. See <a href="#">Managing screens and Managing application variables on page 525</a> .
Build	Building a project. See <a href="#">Building a project on page 518</a> .
Controller	Connecting and deploying to the controller. See <a href="#">Connecting to controller on page 518</a> and <a href="#">Deploying to controller on page 519</a> . Also for opening the Virtual FlexPendant.
Arrange	Re-sizing and positioning the controls on the design area. See <a href="#">Arrange on page 503</a> .
Diagnosis	Detecting problems in the project and providing a diagnostic solution. See <a href="#">ScreenMaker Doctor on page 529</a> .
Close	Closing a project.

---

**Arrange**

This toolbar displays icons for resizing and positioning controls on the design area. The icons are enabled once you select a control or group of controls on the design area.



en0900000592

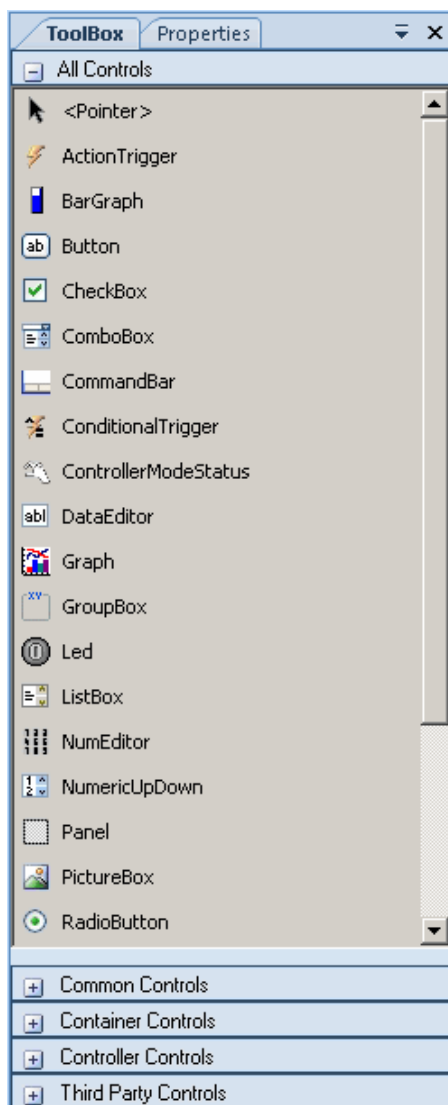
## 15 ScreenMaker tab

### 15.2 Development environment

*Continued*

#### ToolBox

ToolBox acts a container for holding all the available controls that can be placed on a screen.



en0900000407

The following table displays the GUI controls that can be dragged to the design area.

Control	Description
ActionTrigger	Allows to run a list of actions when either a signal or rapid data changes
BarGraph	Represents an analog value in a bar
Button	Represents a control that can be clicked. Provides a simple way to trigger an event, and is commonly used to execute commands. It is labeled either with text or an image.
CheckBox	Allows multiple selections from a number of options. They are displayed as a square box with white space (for unselected) or as a tick mark (for selected).

*Continues on next page*

Control	Description
ComboBox	Represents a control that enables to select items from a list Combination of a drop-down list and a textbox. It allows you to either type a value directly into the control or choose from the list of existing options.
CommandBar	Provides a menu system for a ScreenForm
ConditionalTrigger	Allows to define conditions while defining action triggers. An action is triggered, if there is any change in value of the data bound.
ControllerModeStatus	Displays the mode of the Controller (Auto - Manual)
DataEditor	Represents a text box control that can be used to edit the data.
Graph	Represents a control that plots data with lines or bars.
GroupBox	Represents a Windows control that displays a frame around a group of controls with an optional caption. Is a container used to group a set of graphic components. It usually has a title at the top.
LED	Displays a two states value, like a Digital Signal.
ListBox	Represents a control to display a list of items. Allows the user to select one or more items from a list contained within a static, multiple line text box.
NumEditor	Represents a text box control that can be used to edit a number. When the user clicks it, a Numpad is opened.
NumericUpDown	Represents a spin box that displays numeric values.
Panel	Used to group collection of controls.
PictureBox	Represents a picture box control that displays images.
RadioButton	Allows to select only one of a predefined set of options.
RapidExecutionStatus	Displays the execution status of the Controller Rapid Domain (Running - Auto)
RunRoutineButton	Represents a Windows button control that calls a RapidRoutine when clicked
Switch	Displays and lets change a two states value, like a Digital Output Signal.
TabControl	Manages a set of tab pages.
TpsLabel	Very commonly used widget that displays text, a label is usually static, that is, it has nointeractivity. A label generally identifies a nearby text box or other graphic component.
VariantButton	Used to change the values of RAPID variables or Application variables.

**Note**

For more information on using these controls and their properties, see the section [Development environment on page 502](#) and the chapter *Using the FlexPendant SDK* of the *Application manual - FlexPendant SDK*.

Continues on next page

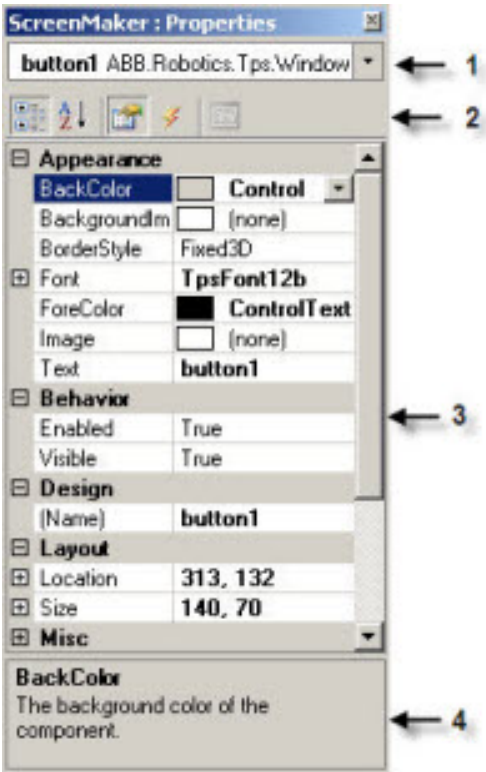
# 15 ScreenMaker tab

## 15.2 Development environment

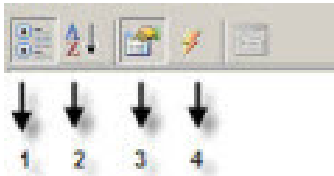
Continued

### Properties window

A control is characterized by its properties and events. Properties describe the appearance and behavior of the component, while events describe the ways in which a control notifies its internal state change to others. By changing the value of a property, the controls have a different look and feel, or exhibit different behavior.



en0900000408

	Element	Description
1	Graphical component name panel	Displays the selected component, and lists the available components of the active design screen.
2	Properties window toolbar	 en0900000409 <ul style="list-style-type: none"><li>1 Organizes table panel in categories</li><li>2 Organizes table panel alphabetically</li><li>3 Displays Properties in table panel</li><li>4 Displays Events in table panel</li></ul>
3	Table panel	Displays all the properties or events in two-columns. The first column shows the property or event name, the second shows the value of the property or name of the event handler.
4	Information panel	Display information about a property or event.

## 15.3 Working with ScreenMaker

### 15.3.1 Managing projects

#### Overview

This section describes how to manage projects in ScreenMaker. A complete cycle includes creating, saving, building, connecting, and deploying a ScreenMaker project.

You can manage a project (create, delete, load, or save) either from the ScreenMaker ribbon or the context menu.

#### Creating a new project

Use this procedure to create a new project:

- 1 Click **New** from the ScreenMaker ribbon or right-click **Project** context menu and select **New Project**.

The **New ScreenMaker Project** dialog box appears.



#### Note

You can create a new project either from *ScreenMaker installed templates* or *ScreenMaker custom templates*.

- 2 To create a new project from the *ScreenMaker installed templates*,
  - a Click *Simple Project*.
  - b Enter a name and specify the location for the new project. By default, the new project is saved on *C:\My Documents\RobotStudio\My ScreenMaker Projects*.
  - c Click **OK**.
  - d A screen *MainScreen(main)* is added in the tree view.
- 3 To create a new project from the **ScreenMaker custom templates**,
  - a Click **Basic**, *Standard*, or **Extended**.
  - b Enter a name and specify the location for the new project. By default, the new project is saved on *C:\My Documents\RobotStudio\My ScreenMaker Projects*.
  - c Click **OK**.



#### Note

- If you select the template **Basic**, a project with two screens are created.
- If you select the template **Standard**, a project with four screens are created.
- If you select the template **Extended**, a project with six screens are created.

*Continues on next page*

## 15 ScreenMaker tab

---

### 15.3.1 Managing projects

*Continued*

---

#### Loading a project or template

Use this procedure to load an existing project or an existing template:

- 1 Click **Open** from the ScreenMaker ribbon or right-click **Project** context menu and select **Open Project**.

The **Open Screen Project File** dialog box appears.



#### WARNING

A warning message appears when you open an existing ScreenMaker project where the FlexPendant SDK version is different from the version the project was created.

- 2 Browse to the location of the project file or template file to be loaded and click **Open**.



#### Note

You can also load an existing project using a quick access method.

- 1 Click **Recent** from the ScreenMaker ribbon or right-click **Project** context menu and select **Recent Projects**.
- 2 Select the project file from the list of most recently opened projects.

---

#### Saving a project

To save a project or template, follow this step:

- Click **Save** from the ScreenMaker ribbon or right-click **Project** context menu and select **Save**.

To save the existing project or template with a new name, follow this step:

- Click **SaveAs** from the ScreenMaker ribbon or right-click **Project** context menu and select **SaveAs**.



#### Note

- Project files are saved with the extension **\*.smk**.
- Template files are saved with the extension **\*.smt**.

#### SaveAs FlexPendant Project

To save the ScreenMaker project as a FlexPendant project, in the project context menu, click **SaveAs FlexPendant Project**.

The project is saved with the extension **\*.csproj** which can be opened using Microsoft Visual Studio 2008.

*Continues on next page*

## Designing screens

This section describes adding, copying, renaming, deleting, and editing a screen.

### Overview

The Form designer is a tool to edit or design a screen. It allows you to design the screen with the required controls and the design area resembles a FlexPendant screen.

### Editing a screen

To edit a screen, follow these steps:

- 1 Drag a control from the toolbox and drop it on the design area.  
The **Properties** window displays all the properties of the control.
- 2 Select the control and resize or reposition for configuration.



#### Note

You can either select a single control or multiple controls:

- Single control : Left-click the control on the design area or select the control from the list in the Properties window.
- Multiple controls: Left-click on the design area, drag the mouse and create a window selecting all the controls.

- 3 Click the smart tag on the upper right corner of the control to perform the basic tasks of configuration. See [Configuring data binding on page 526](#).



#### Note

You can perform additional configuration by editing the attributes in the Properties window. See [Properties window on page 506](#).

## Using ScreenMaker controls

This section describes building the GUIs using the following controls from the ToolBox.

### ActionTrigger

An action trigger initiates an event, such as making a hidden object visible when an action is performed using a control. It allows to run a list of actions when the property value changes. The property value can be bound to a signal, rapid data, or application variable.

ActionTrigger control can also be used to invoke the application from RAPID.

Use this procedure to add an ActionTrigger control::

	Action
1	Drag an <i>ActionTrigger</i> control from the ToolBox on to the design area.

*Continues on next page*

## 15 ScreenMaker tab

### 15.3.1 Managing projects

*Continued*

	Action
2	<p>You can modify the name, set the default value and configure data binding value for a ActionTrigger control.</p> <ul style="list-style-type: none"><li>• To set the values of a property, see <a href="#">Properties window on page 506</a>.</li><li>• You can set the trigger event for an ActionTrigger to any of the event handler created either from a control or from an Events Manager option.</li><li>• To configure the data binding values, see <a href="#">Configuring data binding on page 526</a>.</li><li>• To set the application variables, see <a href="#">Managing application variables on page 525</a>.</li></ul>



#### Note

An action is not triggered when the screen is launched for the first time, but is triggered when there is a difference in the bound value at any point of time. This functionality is supported only in RobotWare 5.12.02 or higher.

**Example:** Consider a signal being bound to the value property. The value of the signal changes at runtime on performing a specific action. The event handler configured for ActionTrigger control gets triggered based on this signal value change.

#### TpsLabel

TpsLabel is a standard Windows label that displays a descriptive text.

Use this procedure to add a TpsLabel control:

Step	Action
1	Drag a <b>TpsLabel</b> control from the <b>ToolBox</b> on to the design area.
2	<p>You can set the values, setup events, configure data binding values and set the application values for a TpsLabel control.</p> <ul style="list-style-type: none"><li>• To set the values of a property, see <a href="#">Properties window on page 506</a>.</li><li>• To set up the events, see <a href="#">Defining Events on page 514</a>.</li><li>• To configure the data binding values, see <a href="#">Configuring data binding on page 526</a>.</li><li>• To set the application variables, see <a href="#">Managing application variables on page 525</a>.</li></ul>
3	<p>You can set the option Allow Multiple States to true and change the property.</p> <ol style="list-style-type: none"><li>1 Click Allow Multiple States. The StatesEditor dialog box appears.</li><li>2 Click the check-box Allow Multi-States, select the properties to change from Properties For States and click OK.</li></ol>

The controls Button, PictureBox, and TpsLabel support AllowMultipleStates. For more information on how to use AllowMultipleStates, see [Picture object and changing images due to I/O on page 532](#).

#### Panel

Panel is used to group a collection of controls.

Use this procedure to add a Panel control:

Step	Action
1	Drag a Panel control from the ToolBox on to the design area.
2	You can add a group of controls to a panel.

*Continues on next page*

Step	Action
3	<p>You can modify the name, set the default value and binding value for a Panel control.</p> <ul style="list-style-type: none"> <li>To set the values of a property, see <a href="#">Properties window on page 506</a>.</li> <li>To set up the events, see <a href="#">Defining Events on page 514</a>.</li> <li>To configure the data binding values, see <a href="#">Configuring data binding on page 526</a>.</li> <li>To set the application variables, see <a href="#">Managing application variables on page 525</a>.</li> </ul>

**Note**

Currently only EventHandler, CancelEventHandlers, and MouseEventArgs are supported.

**ControllerModeStatus**

ControllerModeStatus displays the mode of the controller (Auto - Manual).

Use this procedure to add a ControllerModeStatus control:

Step	Action
1	Drag a ControllerModeStatuscontrol from the ToolBox on to the design area.
2	<p>You can set the values, setup events, configure data binding values, and set the application variables for a ControllerModeStatus control.</p> <ul style="list-style-type: none"> <li>To set the values of a property, see <a href="#">Properties window on page 506</a>.</li> <li>To set up the events, see <a href="#">Defining Events on page 514</a>.</li> <li>To configure the data binding values, see <a href="#">Configuring data binding on page 526</a>.</li> <li>To set the application variables, see <a href="#">Managing application variables on page 525</a>.</li> </ul>
3	<p>You can select the image to be displayed when the controller is in Auto mode and in Manual mode.</p> <ul style="list-style-type: none"> <li>Click AutoImage in the Properties window and browse to select the image to be displayed in Auto mode.</li> <li>Click ManualImage in the Properties window and browse to select the image to be displayed in Manual mode.</li> </ul>

**RapidExecutionStatus**

RapidExecutionStatus displays the execution status of the Controller Rapid Domain (Running - Auto). This control is used

Use this procedure to add a RapidExecutionStatus control:

Step	Action
1	Drag a RapidExecutionStatus control from the ToolBox on to the design area.
2	<p>You can set the values, setup events, configure data binding values, and set the application values for a RapidExecutionStatus control.</p> <ul style="list-style-type: none"> <li>To set the values of a property, see <a href="#">Properties window on page 506</a>.</li> <li>To set up the events, see Setup Events.</li> <li>To configure the data binding values, see <a href="#">Configuring data binding on page 526</a>.</li> <li>To set the application variables, see <a href="#">Managing application variables on page 525</a>.</li> </ul>

Continues on next page

## 15 ScreenMaker tab

### 15.3.1 Managing projects

*Continued*

Step	Action
3	You can select the image to be displayed when the Program is running and is stopped. <ul style="list-style-type: none"><li>Click RunningImage in the Properties window and browse to select the image to be displayed when the Program is running.</li><li>Click StoppedImage in the Properties window and browse to select the image to be displayed when the Program is stopped.</li></ul>

#### RunRoutineButton

RunRoutineButton represents a Windows button that calls a RapidRoutine when clicked.



#### Note

To call a routine containing movements, you are not recommended to use the RunRoutine Button control. Instead use a normal button control to call a Trap routine. In the Trap routine, use instructions such as StopMove, StorePath, RestorePath and StartMove to control the movements of the robot.

Use this procedure to add a RunRoutineButton control:

Step	Action
1	Drag a RunRoutineButton control from the ToolBox on to the design area.
2	Click the smart tag on the RunRoutineButton and select one of the following RunRoutineButtonTasks. <ul style="list-style-type: none"><li>Define Actions before calling Routine</li><li>Select Routine to call</li><li>Define Actions after calling Routine</li></ul>
3	Click Define Actions before calling Routine to define an action/event before calling the routine. The Events Panel dialog box appears.
4	Click Define Actions after calling Routine to define an action/event after calling the routine. The Events Panel dialog box appears.
5	Click Select Routine to call. The Controller Object Binding dialog box appears.
6	In the Properties window, set the value for the following properties: <ul style="list-style-type: none"><li>RoutineToCall - Set the routine to be called. Indicates the RAPID Routine that will be called when this button is pressed.</li><li>AllowInAuto - Set to True or False. Indicates if the routine could be called in the Auto mode.</li><li>TextAlign - Set to MiddleLeft and MiddleCenter. Indicates the text alignment.</li></ul> Note the following restrictions: <ul style="list-style-type: none"><li>You cannot bind RunRoutineButton to built-in Service routines.</li><li>Only user defined procedures with no arguments can be bound.</li><li>Set the PP to task before performing action through RunRoutineButton.</li></ul>

*Continues on next page*

## CommandBar

CommandBar allows you to add menu items in a controlled and organized order. Use this procedure to add menu items to the CommandBar control:

Step	Action
1	Drag a CommandBar control from the ToolBox on to the design area. The CommandBar appear at the bottom of the screen.
2	Click the smart tag on the CommandBar and select Add/Remove Items. The MenuItem Collection Editor window appears.
3	Click Add. A new menu item is added and its properties are displayed which can be edited. Note that while editing the menu item, ensure that the property Text is filled. If not, nothing appears on the CommandBar.
4	To remove the menu item, select the menu item and Click Remove.
5	Click Close to close the MenuItem Collection Editor window.

To add an event to a menu item, for example *menuItem1* on the command bar, use this procedure:

Step	Action
1	Go to the Properties window and select <i>menuItem1</i> from the drop-down list.
2	Click Events icon and then double-click the Click event. This opens the Events Panel dialog for the Click event.
3	Click Add Action from the Events Panel dialog. This opens a sub-list of actions.
4	Click an action from the sub-list of actions to add it to <i>menuItem1</i> 's Click event.

## VariantButton

The VariantButton control is a simple button control with additional features and properties. Using this control, you can change the values of RAPID or Application variables.

Use this procedure to add the VariantButton control:

Step	Action
1	Drag a VariantButton control from the ToolBox on to the design area.
2	You can perform the following VariantButton tasks from the SmartTag: <ul style="list-style-type: none"> <li>• Define Actions before value change</li> <li>• Define Actions after value change</li> </ul>
3	You can set the following VariantButton specific properties from the Properties window: <ul style="list-style-type: none"> <li>• Select Increment or Decrement from Behavior drop down. The default behavior of VariantButton is Increment.</li> <li>• Select StepRate and set the rate at which the value must be varied.</li> <li>• Select DataType to which the value should be bound and set the value property of the selected datatype.</li> </ul> Supports only the RAPID datatypes, <b>Num</b> and <b>Dnum</b> . For more information on data binding, see <a href="#">Configuring data binding on page 526</a> .

Continues on next page

## 15 ScreenMaker tab

### 15.3.1 Managing projects

*Continued*

Step	Action
4	You can also perform the following common tasks from the Properties windows: <ul style="list-style-type: none"><li>• Set BackColor, ForeColor, Location, and Size of the control.</li><li>• Select True or False from the Visible dropdown to hide or unhide the control.</li><li>• Select True or False from the Enabled drop down to enable or disable the control.</li></ul>

#### ConditionalTrigger

The ConditionalTrigger button defines the condition while defining action triggers. An action will be triggered if there is a change in the value of the data bound.

Use this procedure to add the ConditionalTrigger control:

Step	Action
1	Drag a ConditionalTrigger control from the ToolBox on to the design area.
2	You can set the following ConditionalTrigger properties from the Properties window: <ul style="list-style-type: none"><li>• Select the condition to execute from the Condition drop down. The following are the supported conditions AND, OR, XOR, NOT, and EQUAL.</li><li>• Select True or False from the Enabled drop down to enable or disable the control.</li><li>• Select LHS and RHS and bind the data value to Controller Object or Application Variable. For more information on data binding, see <a href="#">Data binding on page 526</a>.</li></ul>

#### Defining Events

Event handler is a set of actions to be executed after an event occurs.

To set up an event, follow these steps:

- 1 Select the control for which the event handler is to be defined.
- 2 Open the **Events Panel** dialog box in any one of the following ways:
  - Double-click the control.
  - Right-click the control, select **Events Manager**, click **Add** enter the name, and click **OK** and close.
  - Click smart tag and select the task from the list.
  - In the **Properties** window, click **Events** icon and select the desired event from the list.
- 3 Click **Add Action** to add an action from a predefined list of actions.

The following table lists the set of predefined actions:

Screens	<ul style="list-style-type: none"><li>• Open Screen</li><li>• Close Screen</li></ul>
Signals	<ul style="list-style-type: none"><li>• Set a Digital Signal</li><li>• Invert a Digital Signal</li><li>• Pulse a Digital Signal</li><li>• Read a Signal</li><li>• Write a Signal</li><li>• Reset a Digital Signal</li></ul>
RapidData	<ul style="list-style-type: none"><li>• Read a Rapid Data</li><li>• Write a Rapid Data</li></ul>
Application Variable	<ul style="list-style-type: none"><li>• Read and Write</li></ul>

*Continues on next page*

Advanced	<ul style="list-style-type: none"> <li>• Call another Action list</li> <li>• Call .NET method</li> <li>• Call Custom Action</li> <li>• Call FP Standard View</li> </ul>
----------	---

- 4 Select the action from the left window and perform the following:
  - Click **Delete** to delete the action.
  - Click **Move Up** or **Move Down** to change the order of execution of actions.
- 5 Click **OK**

#### Deleting an event handler

To delete a user created event handler, do the following:

- 1 Right-click the control, select **Events Manager**. The **Events Manager** dialog box appears.
- 2 Select the event handler to be deleted from the list and click **Delete**.

#### Advanced Actions

##### Call another Action List

Existing event handlers from Events Manager can be reused by other controls while defining actions for event. You can call another event handler from an existing event handler.

In the following example, **listbox1\_SelectedIndexChanged** event handler is called from **comboBox1\_SelectionIndexChanged** event handler.

Select the *Show warning message before performing actions* check box to have a warning displayed before you can perform these actions.

##### Call .NET Method

You can import the dlls and add references to the *Advanced* tab of the **Project Properties** dialog box.

Once the references are defined, .NET methods appear in the *Project Properties* dialog box and can be included in the *Actions* list which will be executed on performing the desired action.

The .NET assembly supports only public static methods.

Double click the method and bind the return value to the application variable.

Binding can be done only to the application variable. For more information, see [Application variable data binding on page 528](#).



#### Note

ScreenMaker allows you to call static methods of the public classes defined in another DLL. This DLL is usually a class library or a control library. It has the following limitations and the user should be aware of them while using .Net DLLs.

- DLL's references must be in the same directory in order to load the DLL.
- ScreenMaker provides access only to the static methods which contain basic data types such as string, int, double, boolean, object.

Continues on next page

The following procedure provides information on creating a .NET assembly. This assembly can be added as a reference to ScreenMaker Project and for performing certain computations which are not directly possible using ScreenMaker or to call methods of FlexPendant or PCSDK.

Use Visual Studio 2010 or above to create a .NET assembly.

- 1 Create a new project with Class Library as your template.
- 2 Create public static methods like the following.

```
namespace SMDotNetMethods
{
    public class Methods
    {
        /// <summary>
        /// Inverts a boolean value
        /// </summary>
        /// <param name = "Value">input boolean value</param>
        /// <returns>inverted boolean value</returns>
        public static bool InvertBool(bool value)
        {
            return (value == false);
        }

        /// <summary>
        /// Increments a numerical value
        /// </summary>
        /// <param name="value">value to be incremented</param>
        /// <returns>incremented value</returns>
        public static double Increment(double value)
        {
            return (value + 1);
        }
    }
}
```

- 3 Build the project.
- 4 Use the assembly generated from this Class Library project.
- 5 Add it as a reference to the ScreenMaker project.

#### Call Custom Action

You can add an user control to the *ScreenMaker toolbox* and call a custom method for that control by defining it in the *ScreenMaker.dll.config* file.

Call Custom Action supports only the Graph control.

#### Call FP Standard View

Standard FlexPendant screens can be opened on any action performed on the control. The standard FlexPendant screens include Rapid Editor, Rapid Data, LogOff, Jogging, Backup and Restore.

For example, on button1\_click, Rapid Editor view is opened.

### Editing the property value

You can edit the property value of a control from the *Properties window* in three ways:

- 1 By typing the numerics, strings and text. For example, Location, Size, Name etc.
- 2 By selecting the predefined values from the list. For example, BackColor, Font etc.
- 3 By entering the values in the dialog box. For example, Enabled, States, BaseValue etc.

### Deleting an event handler

To delete a user created event handler, do the following:

- 1 Right-click the control, select **Events Manager**. The **Events Manager** dialog box appears.
- 2 Select the event handler to be deleted from the list and click **Delete**.

### Modifying Project properties

Project properties define the properties of the ScreenMaker project, including how the GUI is loaded and displayed in the FlexPendant.

Use this procedure to modify the project properties:

- 1 Right-click **Project** context menu and select **Properties**.  
The **Project Properties** dialog box appears.
- 2 In the **Display** tab under **Caption**, enter the text in the **Caption of the Application** field to edit the caption.  
The updated caption appears in the **ABB Menu** on the right side.
- 3 In the **Display** tab under **ABB Menu**, select the following options,

Option	Description
Left	application is visible to the left in the ABB Menu.
Right	application is visible to the right in the ABB Menu.
None	application is not visible at all in the ABB Menu.



#### Note

The applications that uses the option **None** cannot be run on RobotWare releases earlier than 5.11.01.

- 4 In the **Display** tab under **ABB Menu**, browse and select the **ABB menu image**.
- 5 In the **Display** tab under **TaskBar**, browse and select the **TaskBar image**.



#### Note

By default, the **Use Default Image** and **Use Menu Image** checkbox is enabled and the default image *tpu-Operator32.gif* is selected.

Continues on next page

## 15 ScreenMaker tab

---

### 15.3.1 Managing projects

*Continued*

- 6 In the **Display** tab under **Startup** , select **Automatic** to load the screen automatically at the Startup.



#### Note

By default, the start up type is **Manual**.

- 7 In the **Advanced** tab under **Run Settings**, select **Launch virtual FlexPendant after deploying** checkbox.

The virtual FlexPendant will be launched after deploying the ScreenMaker project to the virtual controller.



#### Note

This feature is not applicable if connected to a real controller.

- 8 In the **Project Properties** dialog, select the **General** tab to view the project properties which includes, **Name**, **Assembly**, **Version** and **Path**.

Version displays the specific versions of Controller and FlexPendant SDK that the ScreenMaker project uses.

---

### Connecting to controller

Use this procedure to connect to both real and virtual controllers:

- 1 Click **Connect** from the ScreenMaker ribbon or right-click **Project** context menu and select **Connect**.

The **Select a Robot Controller** dialog box appears.



#### Note

Click the **Connect** dropdown from the ScreenMaker ribbon to directly connect to the controller.

- 2 Click **Refresh** to find a list of all the available controllers.



#### Note

By default, the currently connected controller is highlighted and has a small icon before the row as an indicator.

- 3 Select the controller to be connected from the list and click **Connect**.

The connection status is displayed in the **Project tree view**.

To remove the connection with the controller, click **Disconnect** from the **Project** context menu.

---

### Building a project

The result from building the ScreenMaker project is a set of files including DLL file and images. The ScreenMaker project can be compiled into binary format (.dll) that can be deployed on a FlexPendant.

*Continues on next page*

Use this procedure to build a project:

- 1 Click **Build** from the ScreenMaker ribbon or right-click Project context menu and select **Build**

The result is displayed in the output window.

---

### Deploying to controller

Use this procedure to deploy a ScreenMaker project to a real controller or virtual controller:

- 1 Connect to the controller you want to deploy to. See [Connecting to controller on page 518](#).
- 2 Click **Deploy** from the ScreenMaker ribbon or right-click Project context menu and select **Deploy Screen to Controller**.

The **Download** dialog box appears displaying the progress of download. It disappears once the download is successful.

The **TpsViewxxxxx.dll** file is downloaded.

- 3 Restart the controller.



#### Note

- If a real controller is used, you can reboot the FlexPendant by moving its joystick three times to the right, once to the left, and once towards you.
- If a virtual controller is used, you can reboot the FlexPendant by closing the virtual FlexPendant window.

---

### Closing a project

To close a project, follow this step:

- Right-click **Project** context menu and select **Close Project**.

### Closing ScreenMaker

To close ScreenMaker, follow this step:

- Click **Close ScreenMaker** from the ScreenMaker ribbon.

---

### Managing ScreenMaker Widgets

#### What is a widget

A widget is a visual building block, containing an information arrangement, which represents an aspect of a robot application. It is a reusable and sharable user interface building block which can help speed up the development of screens.

A ScreenMaker widget is similar in function to the widgets used in computer programming. The widget is an element of a graphical user interface (GUI) that displays an information arrangement which is changeable by the user. The widgets, when combined in an application, hold data processed by the application and the available interactions on this data.

*Continues on next page*

## 15 ScreenMaker tab

---

### 15.3.1 Managing projects

*Continued*

#### Widget Workflow

Widget created from ScreenMaker can be used in ScreenMaker application and in Production Screen application.

The following are the steps required to create a Widget in ScreenMaker.

- 1 Start RobotStudio.
- 2 Launch ScreenMaker.
- 3 Create a new **Widget Project** or open an existing widget project.  
For information on how to create a new widget project, see [Creating a ScreenMaker widget project on page 520](#).
- 4 Connect to a real or a virtual controller, as required.
- 5 If required, change the widget properties, using the **Widget Properties** dialog box.  
For information on the Widget Properties dialog box, see [Specifying widget properties on page 523](#).
- 6 Drag-and-drop the necessary user interface components, as you would in a normal ScreenMaker project.
- 7 Link the user interface properties to the IRC5 data or to the application variables
- 8 Build the widget project. The widget component is created and saved in  
...\\Documents\\RobotStudio\\Widget Components folder.

#### Sample use case

Consider a case where you want to design a production screen which can do the following:

- Display a graph
- Show alarms
- Show status of the controller

To achieve this:

- 1 Create a new widget project in ScreenMaker and names it as, for example, GraphWidget.
- 2 Drags-and-drop the graph control and other necessary controls on the widget form.
- 3 Connect to a real or virtual controller, as required.
- 4 Bind the controls to the controller data.
- 5 Use the widget properties dialog box, to change the size of the widget.
- 6 Build the project
- 7 Download the output to the production screen.

You can then repeat the above steps to create widgets either in the same or in different projects based on your need to show the alarms and the status of the controller.

#### Creating a ScreenMaker widget project

- 1 On the ScreenMaker tab, click **New**. Alternatively, in the project context menu, click **New Project**.

*Continues on next page*

The **New ScreenMaker Project** dialog box appears.

- 2 Under **Widget Templates**, click **Widget**.
- 3 Specify a name for the widget project.

ScreenMaker widgets projects are by default stored in the  
... \Documents\RobotStudio\Widget Projects folder.

- 4 Click **Ok**.

The widget project along with a screen **MainScreen(main)**, appears in the tree view. The widget project has **.wzp** file name extension. Widgets also appear in the **Toolbox**.



#### Note

- You can have only one widget project open at any time. Close an open widget project before opening a new one.
- A widget project has only one screen, the main screen, on which the widgets are designed. All controls defined on a widget are considered as one widget.
- Widgets are loaded into the toolbox from a folder which contains the widget component DLLs, from Additional Options folder under MediaPool and from RobotApps Repository. If you delete the widget components from any of the above locations (... \Documents\RobotStudio\Widget Components), then the widgets will not be appear in the Toolbox.

#### Creating Production Screen Widget

ScreenMaker helps the user to create two types of widgets, Production Screen widget and Standard widget. Controls in a widget can be bound to rapid or signal data.

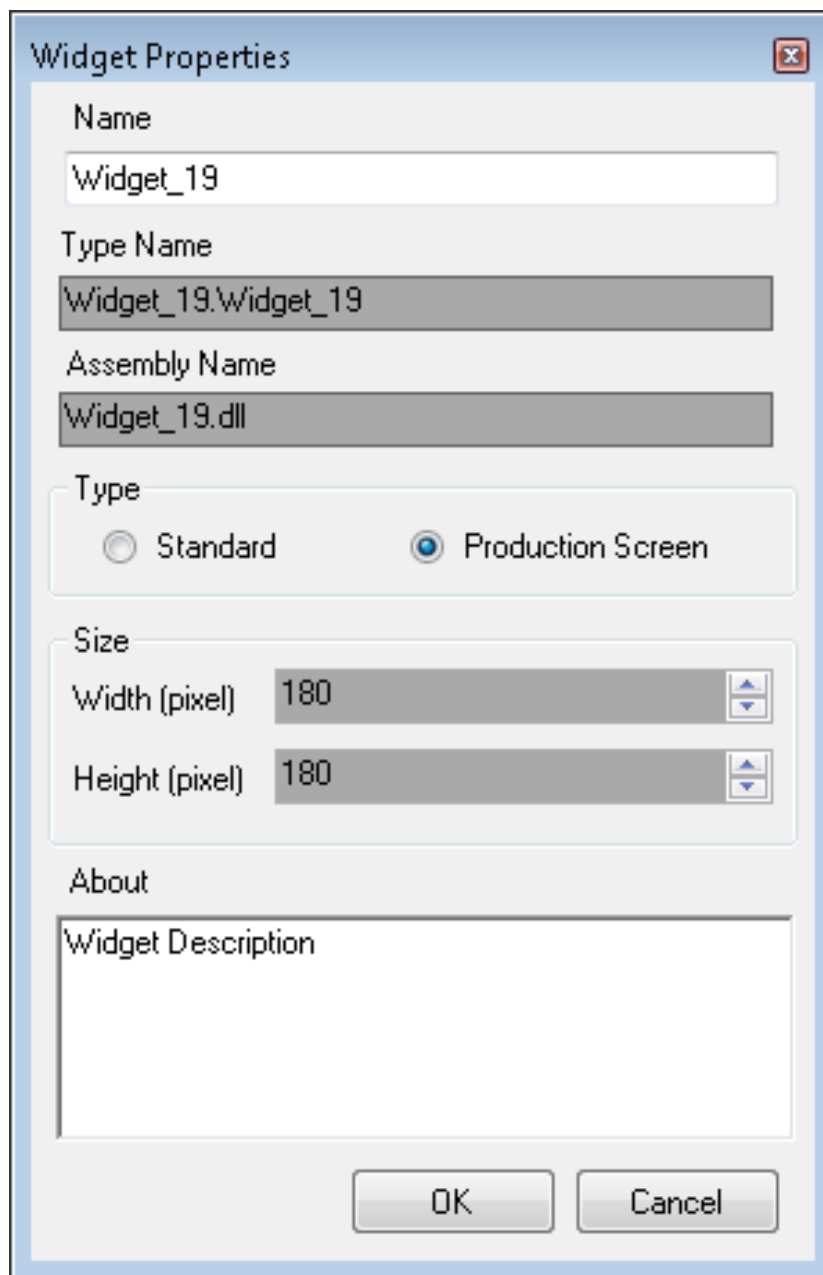
The Production Screen option is a framework for creating a customized GUI that can be used to present process data and status and execute FlexPendant applications.

To run widgets on the Production Screen, FlexPendant Interface option must be selected. For more information on Production Screen. Use the following procedure to create the Production Screen widget.

- 1 In the **Screenmaker** ribbon, select **New**. The **New Project** dialog opens
- 2 Select **Widget Template** to create a new widget project.
- 3 Drag and drop controls to the widget.
- 4 Select **Widget Properties**, **Widget Properties** dialog opens.
- 5 Under **Type**, click **Production Screen** and click **OK**.

*Continues on next page*

#### 6 Build the project.



1400000275

The *ProductionSetup.xml* file must be updated with widget details to view the widget that was created in the Production Screen. You can find *ProductionSetup.xml* under *\$System\HOME\ProdScr* and Widget components under *\$System\HOME\ProdScr\tps*.

An example of widget detail is provided here:

```
<Widget>
<Name>Widget_9</Name>
<Page>1</Page>
<Assembly>Widget_9.dll</Assembly>
<Type>Widget_9.Widget_9</Type>
```

Continues on next page

```

<Position>
<X>1</X>
<Y>2<Y>
</Position>
<ZIndex>1</ZIndex>
<Bindings>
<Binding PropertyName ="led1.Value" BindingType="SIGNAL"
      DataName="MOTLMP" />
<Binding PropertyName ="button1.Text" BindingType="RAPID"
      DataName="T_ROB1/BASE/wobj0" />
</Bindings>
</Widget>

```

The production screen provides the flexibility to modify bindings of the widget. This is provided under Bindings tag as shown here:

```

<Bindings>
<Binding PropertyName ="led1.Value" BindingType="SIGNAL"
      DataName="MOTLMP" />
<Binding PropertyName ="button1.Text" BindingType="RAPID"
      DataName="T_ROB1/BASE/wobj0" />
</Bindings>

```

#### Specifying widget properties

To specify the properties of a widget project, right-click a widget project, and then click **Properties**. The Widget Properties dialog box appears.

You set and modify the following in the properties for the widget project:

- Name of the project
- Size of the widget - x,y (in mm)
- Select the type of Widget
  - Production Screen: The Widget can be used with Production Screen environment
  - ScreenMaker: The Widget can be used with ScreenMaker applications

#### Modifying Binding Information of Widget

Use this option to modify the binding information of widget. When a widget is built from the Widget Project, an xml file is created. This xml contains widget details and binding information. This entry must be available in the *Production.xml* file to work with the Production Screen Environment .

```

<Bindings>
<Binding PropertyName ="meter1.Value" BindingType="IO"
      DataName="aoMeterSignal" />
<Binding PropertyName ="meter1.Title" BindingType="RAPID"
      DataName="Flow1Title" />
</Bindings>

```

It is possible to create, use and modify the bindings of a widget created from ScreenMaker and to view the results in Production Screen and in ScreenMaker application environment.

*Continues on next page*

## 15 ScreenMaker tab

---

### 15.3.1 Managing projects

*Continued*

#### Building and Deploying

The output of the Widget Project is a single Widget Component dll file, for example, *TpsViewMyWidget.dll*. The widgets built from the Widget Project are used in the ScreenMaker project. Widgets cannot be deployed to the controller from ScreenMaker. If Widgets are used in ScreenMaker projects, it gets deployed.

When the ScreenMaker project which uses a widget is built, the widget component is added as a reference to the project.

When the ScreenMaker project output is deployed to the controller, the referenced widget components are also copied to the system *HOME* folder.

## 15.3.2 Application variables

### Overview

Application Variables are variables defined inside a ScreenMaker application. An application variable is similar to a RAPID variable. It supports the data types supported by RAPID, such as num, dnum, string, tooldata, wobjdata, and so on. An application variable's definition includes its name, data type and initial value. During the execution of the ScreenMaker application, an application variable has a persistent value. It can store values coming from controller data or can be used to write values to controller data. Therefore, it is like an intermediate persistent variable which is used during RAPID execution along with other RAPID variables.

### Managing application variables

To create, delete, and rename an application variable, follow these steps:

- 1 On the **ScreenMaker** tab, in the **Add** group, click **Application Variables**.  
Alternatively, in the **ScreenMaker** browser, right-click the project, and then click **Application Variables**.  
The **Project Application Variables** dialog box appears.
- 2 Click **Add** and define the name, type and value of the new variable.
- 3 Select the variable, click **Delete** to delete a variable.
- 4 Select the variable, click **Rename**, enter the new name and click **OK** to rename a variable.
- 5 Click **Close**.

You can view the application variables related to a project listed in the **Project Application Variables** dialog box. To filter and view the variables according to their data types, use the **Type** list.



#### Note

For information on application variable data binding, see [Data binding on page 526](#).

### 15.3.3 Data binding

#### Overview

Data binding is the mechanism that links a GUI property with an external data source such that whenever the data source is updated the GUI property will be updated automatically and vice versa. Databinding has the following three aspects:

- A *unidirectional* connection means that an update of the data source is reflected by the GUI, or vice versa; a *bidirectional* connection means that updates to either are reflected by the other.
- A *temporal* connection can be suspended and resumed at any time.
- A *convertable* connection negotiates between the different data types or formats between the data source and the GUI property.

A screen has to be linked with data to be useful. There are two ways of linking the data with the GUI properties:

- [Controller object data binding on page 527](#)
- [Application variable data binding on page 528](#)

#### Configuring data binding

Data binding can be configured in the following two ways:

##### Using smart tag

Smart tags perform basic configuration tasks like binding default GUI property with controller data. The controls that either display or edit information normally have a value property to represent the information. Smart tag binds the value to the controller object.

- On the design area, select the control and click the smart tag. The tasks menu appears.

Click...	to...
Bind Value to a Controller Object	bind data to a Controller Object. For more information, see <a href="#">Controller object data binding on page 527</a> .
Bind Value to an Application Variable	binding data to a application variable. For more information, see <a href="#">Application variable data binding on page 528</a> .

##### Using Binding menu

- 1 On the design area, select the control.
- 2 In the Properties window, locate the row from the table for binding the value.
- 3 Select the property and click the list to display the Binding menu.

Click...	to...
Remove actual binding	removes the existing data binding.
Bind to a Controller object	select available data in the controller for binding. For more information, see <a href="#">Controller object data binding on page 527</a> .

*Continues on next page*

Click...	to...
Bind to an Application variable	select available data in project temporary data store for binding. For more information, see <a href="#">Application variable data binding on page 528</a> .

### Configuring data binding for different controls

Almost all the controls defined in the toolbox (except ComboBox and ListBox) have the following two options for binding values:

- Bind to a Controller Object
- Bind to an Application Variable

Binding to an array can be done with the following controls:

- DataEditor
- ComboBox
- ListBox

Control	Description
DataEditor	The default index value is 1. DataEditor is designed in such a way that the default value of the RAPID array starts with 1 and not 0.
ComboBox and ListBox	The default index value is -1. You can enter the appropriate index value but cannot bind to a controller object or an application variable. Note the following: <ul style="list-style-type: none"> <li>• You can limit the number of items to be displayed in the ComboBox and ListBox of an array.</li> <li>• While using a ComboBox, a RAPID index starts with 1 (1 specifies the first element) and the ComboBox index starts with 0 (0 specifies the first index).</li> </ul>

For more information on RAPID array, see [What is RAPID array on page 533](#).

### Controller object data binding

Controller object data binding lets you to select the data in the controller for binding.

Use this procedure to set up a binding with controller objects:

- 1 Select **Bind to a Controller Object** either using smart tag or binding menu.  
The **Controller Object Binding** dialog box appears.
- 2 In the **Type of Object** group, select either **Rapid data** or **Signal data**.
- 3 In the **Shared** group, select **Built-in data only** to access shared **Rapid data**.  
When you select **Built-in data only**, the option **Signal data** and the text box **Module** are disabled.
- 4 If you select **Rapid data**, from the **Scope** group, select a task and module from the list.  
When you select **Signal data**, the **Scope** group is disabled.
- 5 In the **See** list, select the desired data.

*Continues on next page*



#### Note

ScreenMaker supports binding to only constant and persistent variables. The variables must not be declared LOCAL. TASK PERS is supported

For example, the following binding is supported:

```
PERS num n1:=0;  
TASK PERS num n2:=0;  
CONST num n3:=0;
```

The following binding is not supported:

```
LOCAL PERS num n1:=0;  
VAR num n1:=0
```

---

### Application variable data binding

Application variables are used for data binding in the same way as controller data. See [Controller object data binding on page 527](#).

Use this procedure to set up a binding with application variables:

- 1 Select **Bind to an Application Variable** either using smart tag or binding menu.

The **Application Variables Bind Form** dialog box appears.

- 2 Select an application variable and the field to connect.
- 3 Click **Setup Variables** to manage the variables.

The **Project Application Variables** dialog box appears. See [Managing application variables on page 525](#).

- 4 Click **OK**.

### 15.3.4 ScreenMaker Doctor

#### Overview

ScreenMaker Doctor is a diagnostic solution to detect problems in the ScreenMaker project. It helps analyze the project and fix errors such as:

- Unused events
- Broken references, application variables, signals, modules, and Rapid data
- RunRoutine issue

#### Using ScreenMaker Doctor

Use this procedure to launch ScreenMaker Doctor, detect and report issues, and to view causes and solutions:

- 1 In the ScreenMaker ribbon, click **ScreenMaker Doctor**.

The **ScreenMaker Doctor Wizard** opens.

- 2 Click **Next**.

The wizard starts detecting issues and are reported as **Completed Checks**.

The detected issues are categorized as:

- Broken References
- Unused Events
- Broken ApplicationVariables
- Broken Signals
- Broken Modules
- Broken RapidData
- RunRoutine issue
- Broken Routine
- Other Dependencies

- 3 Click **View Causes and Solutions** to generate a report.

The left hand side of the report displays issues under each category and the right hand side of the report displays the Probable Causes and Solutions for the issues.

To check for issues again using the same instance, click **Re-Detect Issues**.



#### Note

In order to detect the signal data and RAPID, ScreenMaker project should be connected to the controller.

#### Errors fixed by ScreenMaker Doctor

The following sections show you how errors, which can be fixed by ScreenMaker Doctor, may manifest.

#### Unused Events

The following sequence of actions will result in creating unused events.

- 1 Create a ScreenMaker project.

*Continues on next page*

## 15 ScreenMaker tab

---

### 15.3.4 ScreenMaker Doctor

*Continued*

- 2 Define events for the controls.
- 3 Define the events *Button1\_Click* and *Button2\_Click* for the controls *Button1* and *Button2* respectively.
- 4 Delete the control *Button1*. The event *Button1\_Click* will still exist. An unused event is created.

You can run ScreenMaker Doctor to detect and fix this error.

#### Broken Reference

The following sequence of actions will result in creating broken references.

- 1 Create a ScreenMaker project.
- 2 Define events for the controls.
- 3 Define the events *Button1\_Click* and *Button2\_Click* for the controls *Button1* and *Button2* respectively.
- 4 Define action *ScreenOpen - Screen2* for the event *Button1\_Click*.
- 5 Delete or rename the screen. A broken reference is created.

You can run ScreenMaker Doctor to detect and fix this error.

#### Broken Application Variables

The following sequence of actions will result in creating broken application variables.

- 1 Create a ScreenMaker project.
- 2 Add an Application variable to the project.
- 3 Rename or delete the Application variable. No error is reported.

An error is reported during the run time due to the broken application variable.

You can run ScreenMaker Doctor to detect and fix this error.

#### Broken Rapid Data/Signals

If rapid data is bound but not found in the controller connected in the ScreenMaker project, then perform the following procedure:

- 1 Create a ScreenMaker project.
- 2 Connect to a controller.
- 3 Bind the properties of the controls with controller data.
- 4 Build the project and deploy it to the controller.  
The application works.
- 5 Connect the ScreenMaker project to another controller and deploy the same project.  
The application produces errors in the FlexPendant.
- 6 Run ScreenMaker Doctor. It detects that RapidData is not found in the controller, thereby suggesting to define the same.

#### Broken Modules

If modules are bound but not found in the controller connected in the ScreenMaker project, then perform the following procedure:

- 1 Create a ScreenMaker project.
- 2 Connect to a controller.
- 3 Bind the properties of the controls with controller data.

*Continues on next page*

- 4 Build the project and deploy to controller.  
The application works.
- 5 Connect the ScreenMaker project to another controller and deploy the same.  
The application produces errors in the FlexPendant.
- 6 Run ScreenMaker Doctor.  
It detects that the module in which the rapid data was defined is not found in the controller, thereby suggesting to define the same. ScreenMaker doctor also detects Hidden modules.

#### RunRoutine Issue

A check is made whether *ScreenMaker.sys* file is loaded on the controller or not.

An issue is detected if the system module is not loaded.

You can run ScreenMaker Doctor to detect and fix this error.

## 15.4 Frequently asked questions

---

### How to deploy manually to a Virtual Controller

If for any reason you wish to manually by-pass the Deploy button in RobotStudio and the virtual controller, the following information describes what files are to be moved.

#### Actions

##### Location of output files

The files that contain the FlexPendant application from ScreenMaker are found (for example) in the **bin** directory under the **My ScreenMaker Projects** located in the **My documents** directory of the user.

For example, **My Documents\My ScreenMaker Projects\SCM\_Example\bin** where **SCM\_Example** is the example ScreenMaker project.

The files in the **bin** directory are to be copied to a location where the Virtual FlexPendant can read them during the start of the FlexPendant.

##### Location where the Virtual FlexPendant reads the files

The recommended location for manually copying the ScreenMaker output files is the location of the virtual controller system.

If the system is created manually from **System Builder**, it is located in the **My Documents** directory.

For example, **My Documents\IRB4400\_60\_SCM\_Example\HOME** where **IRB4400\_60\_SCM\_Example** is the example controller system.

If the system is created by a Pack-and-Go and then restored, it is located in the **RobotStudio\System**s folder.

For example,

**MyDocuments\RobotStudio\System\IRB4400\_60\_SCM\_Example\HOME** where **IRB4400\_60\_SCM\_Example** is the example controller system.

##### Copy files

Copy the files from the ScreenMaker output to the Home directory of the virtual controller system.

Restart the Virtual FlexPendant and the new application will be loaded.

---

### Picture object and changing images due to I/O

The typical user objective is to have an image that changes when an I/O signal changes, this is common for a digital input to affect the state on the FlexPendant.

#### Actions

This is accomplished by adding an image and allowing the image to have multiple states.

Set **AllowMultipleState** to **TRUE** and set the Image state.

Create two states and add images for each state:

The **Value** property is extremely important. If binding to a digital input then there are two states for the input, 0 and 1. Set the **Value** property to the value of the

*Continues on next page*

bound variable. 0 and 1 for digital input. It is also possible to bind to RAPID variables and have multiple states and values for the values in the RAPID variable.

Set the SelectedStateValue property to bind to a controller object:

---

**How to get radio buttons to show state when entering**

The objective is to have two radio buttons that controls one digital output. When the screen is loaded, the buttons should show the current state of the output.

**Actions**

Create a group or a panel and place the two radio buttons on the group or panel.

For button1, set the property default value to **True** and bind the property to the value of the controller digital output signal.

For button2, do not do any changes.

When the screen is loaded, the state of the two radio buttons is established correctly.

---

**What is RAPID array**

A RAPID array is a variable that contains more than one value. An index is used to indicate one of the values.

**Sample RAPID array**

Consider the following RAPID code.

```
VAR string part{3} := ["Shaft", "Pipe", "Cylinder"];
```

Here, 'part' is a RAPID array which consists of three values. The index of the array in part ranges from 1 to 3.

The index of a RAPID array should not be negative and should start with 1.

---

**Screen navigation**

Screen navigation in ScreenMaker follows a tree structure.

Consider the following example,

- To open screen **A1**, you first have to open **Screen A**
- To navigate from screen **A1** to screen **B1**, you first have to close screen **A1** and then **Screen A** and navigate from **Main Screen** through **Screen B** to screen **B1**.
- Similarly, to navigate from screen **B1** to screen **C1**, you first have to close screen **B1** and **Screen B** and then navigate from **Main Screen** through **Screen C** to screen **C1**.

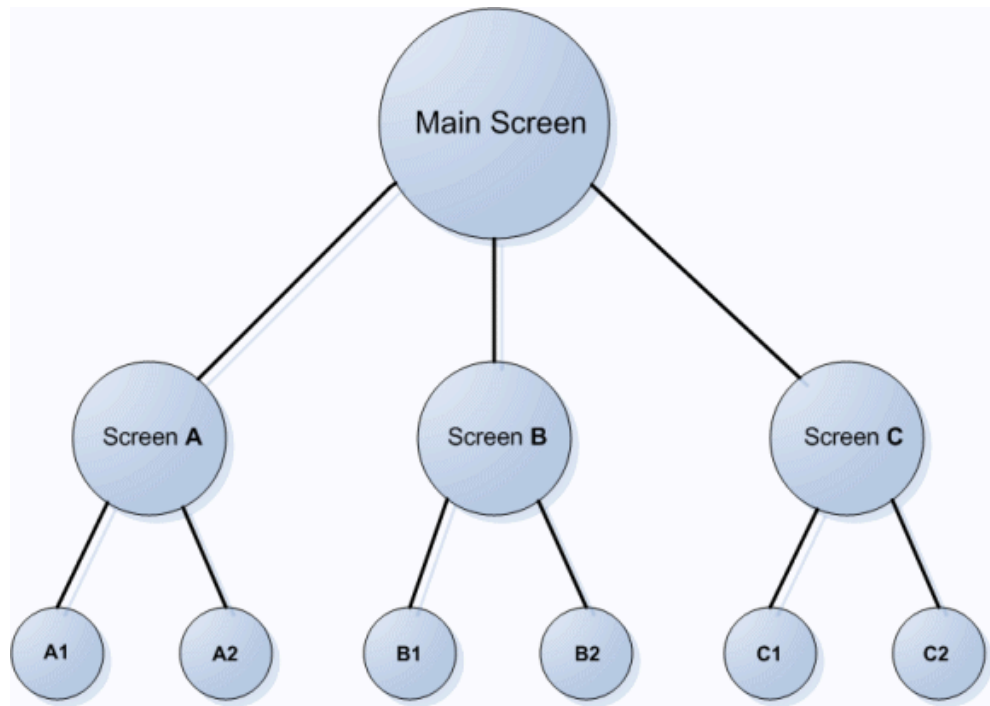
*Continues on next page*

## 15 ScreenMaker tab

---

### 15.4 Frequently asked questions

*Continued*



en0900000645

## 15.5 Tutorial

### 15.5.1 Overview

---

#### About this chapter

This chapter is designed as a tutorial to take you through the steps involved in designing a FlexArc Operator Panel.

The FlexArc Operator Panel is a simple arc welding cell, where the robots can perform the following three different jobs.

Job	Description
Produce	Welding the part
Service	Service the welding gun
Bull's Eye	Calibrate with bull's eye

The FlexArc Operator Panel displays the following graphic elements:

- Controller Status (controller mode auto or manual and the RAPID execution status)
- Part Status (number of produced parts, the average cycle time per part, and a Reset button)
- Robot jobs (Produce, Service, and Bull's Eye) and Robot locations (Robot at home position, service location, calibration location, and part location)
- Start and Stop buttons

#### 15.5.2 Designing the FlexArc operator panel

##### Procedure

Use this procedure to design the FlexArc operator panel:

	Action	Info
1	Create a system for the FlexArc operator panel.	Select the following options, <ul style="list-style-type: none"><li>• FlexPendant Interface</li><li>• PC Interface</li></ul> For more information about creating a system, see <a href="#">Creating a system from layout on page 206</a> .
2	Load EIO.cfg and MainModule.mod files.	For more information on loading these files, see <a href="#">Loading a configuration file on page 374</a> and <a href="#">Loading a RAPID module on page 423</a> . By default: <ul style="list-style-type: none"><li>• For <i>Windows XP</i>, the files can be found at <i>C:\Documents and Settings\&lt;user name&gt;\My Documents\RobotStudio\My ScreenMaker Projects\Tutorial</i></li><li>• For <i>Windows 7</i>, the files can be found at <i>C:\Users\&lt;user name&gt;\Documents\RobotStudio\My ScreenMaker Projects\Tutorial</i></li></ul>

*Continues on next page*


Action		Info	
3	The following signals are created after loading EIO.cfg file		
	<b>IO</b>	<b>Type</b>	<b>Description</b>
	<b>Conne-</b>	<b>tion</b>	
	DI_Ro- botAtHome	DI	Indicates robot at home position
	DI_RobotAt- Bullseye	DI	Indicates robot at bull's eye position
	DI_RobotAt- Service	DI	Indicates robot at service position
	DI_PRO- DUCE	DI	Indicates robot is producing part
	DO_SIM- HOME	DO	Simulate robot at home
	DO_SIMBU- LLS	DO	Simulate robot at bull's eye
	DO_SIM- SERVICE	DO	Simulate robot at service
	DO_PRO- DUCE	DO	Simulate robot is producing part
	GI_JOB	GI	The code of ordered job
	GO_JOB	GO	Simulate job order
4	Create an empty station in RobotStudio with the system created in the previous step.		For more information about creating a station, see <a href="#">New on page 190</a> .
5	Launch ScreenMaker from RobotStudio.		For more information, see <a href="#">Launching ScreenMaker on page 364</a> .
6	Create a new ScreenMaker project.		For more informaton, see <a href="#">Creating a new project on page 507</a> . <ol style="list-style-type: none"> <li>1 Enter the project name as <i>Flex-ArcGUI</i>, and save it in the default location, <i>C:\Users\&lt;user name&gt;\Documents\RobotStudio\My ScreenMaker Projects\Tutorial</i>.</li> <li>2 A new tab <i>MainScreen</i> is added to the Design Surface.</li> </ol>
7	Configure the Project properties.		To customize how the GUI should appear on the FlexPendant, modify the Project properties. For more information, see <a href="#">Modifying Project properties on page 517</a> .

Continues on next page

## 15 ScreenMaker tab

### 15.5.2 Designing the FlexArc operator panel

*Continued*

	Action	Info																		
8	Connect to the controller.	For more information, see <a href="#">Connecting to controller on page 518</a> . The result appears in the output window.																		
9	Create application variables (temporary variables) and configure them with the following data <table><tr><th>Name</th><th>Type</th><th>Value</th></tr><tr><td>MyResetValue</td><td>Num</td><td>0</td></tr><tr><td>JobProduce</td><td>Num</td><td>1</td></tr><tr><td>JobIdle</td><td>Num</td><td>0</td></tr><tr><td>JobBulls</td><td>Num</td><td>2</td></tr><tr><td>JobService</td><td>Num</td><td>3</td></tr></table> For more information, see <a href="#">Managing application variables on page 525</a> .	Name	Type	Value	MyResetValue	Num	0	JobProduce	Num	1	JobIdle	Num	0	JobBulls	Num	2	JobService	Num	3	
Name	Type	Value																		
MyResetValue	Num	0																		
JobProduce	Num	1																		
JobIdle	Num	0																		
JobBulls	Num	2																		
JobService	Num	3																		
10	Design the Main Screen.	For more information, see <a href="#">Designing the screen on page 539</a> .																		
11	Build and Deploy the project.	For more information, see <a href="#">Building and deploying the project on page 545</a> .																		
12	Open virtual FlexPendant and test the GUI	<ul style="list-style-type: none"><li>• In RobotStudio, press Ctrl+F5 to launch the virtual FlexPendant.</li><li>• Click FlexArc operator panel to launch the GUI.</li></ul> <div> <b>Note</b> Ensure that you switch the controller to Auto mode and start the RAPID execution.</div>																		

### 15.5.3 Designing the screen

#### Introduction to designing the screen

A major effort in the GUI project development is designing screens. The Form designer in the ScreenMaker allows you to drag controls from the toolbox to the design surface. Using the Properties window, you can resize, position, label, color, and configure the controls.

#### Designing FlexArc Operator Panel screen

Use this procedure to design the FlexArc Operator Panel screen:

- 1 Drag a GroupBox control from the General category; place it on the design surface and set the following values in the Properties window.

Property	Value
Location	14,45
Size	150,100
Title	Controller Status
BackColor	LightGray

- 2 Drag another GroupBox control from the General category; place it on the design surface and set the following values in the Properties window.

Property	Value
Location	14,170
Size	150,204
Title	Part Status
BackColor	LightGray

- 3 Drag a ControllerModeStatus control from the Controller Data category; place it in the *Controller Status* groupbox created and set the following values in the Properties window:

Property	Value
Location	19,40
Size	44,44
BackColor	LightGray

- 4 Drag a RapidExecutionStatus control from the ControllerData category; place it in the *Controller Status* groupbox created and set the following values in the Properties window:

Property	Value
Location	80,40
Size	44,44
BackColor	LightGray

*Continues on next page*

## 15 ScreenMaker tab

### 15.5.3 Designing the screen

*Continued*

- 5 Drag a TpsLabel control from the General category; place it in the *Part Status* groupbox created and set the following values in the Properties window:

Property	Value
Location	16,30
Size	131,20
Text	Parts Produced
BackColor	LightGray
Font	TpsFont10

- 6 Drag a NumEditor control from the ControllerData category; place it in the *Parts Status* groupbox created and set the following values in the Properties window:

Property	Value
Location	16,56
Size	116,23
Value	Link to RAPID variable <i>partsReady</i> defined in the module <i>MainModule</i> .

- 7 Drag another TpsLabel control from the General category; place it in the *Part Status* groupbox created and set the following values in the Properties window:

Property	Value
Location	16,89
Size	131,20
Text	Cycle time/part
BackColor	LightGray
Font	TpsFont10

- 8 Drag another NumEditor control from the General category; place it in the *Part Status* groupbox created and set the following values in the Properties window:

Property	Value
Location	16,115
Size	116,23
Value	Link to RAPID variable <i>cycleTime</i> defined in the module <i>MainModule</i> .

- 9 Drag a Button control from the General category; place it in the *Part Status* group box created and set the following values in the Properties window:

Property	Value
Location	33,154
Size	85,34
Text	Reset

*Continues on next page*

Perform the following for the **Reset** button in the *Part Status* group:

Step	Action
1	Double-click the button <b>Reset</b> . The <b>Events Panel</b> dialog box appears which is used to define the actions for Events.
2	<p>In the <b>Events Panel</b> dialog box, click <b>Add Action</b>; point to <b>Rapid Data</b> and select <b>Write a Rapid Data</b>.</p> <p>The <b>Action Parameters</b> dialog box appears; assign Rapid data to the following value and click <b>OK</b>.</p> <ul style="list-style-type: none"> <li>• <b>T_ROB1.MainModule.partsReady</b> to <b>MyResetValue.Value</b></li> </ul> <p>Similarly, assign Rapid data to the following value and click <b>OK</b>.</p> <ul style="list-style-type: none"> <li>• <b>T_ROB1.MainModule.cycleTime</b> to <b>MyResetValue.Value</b></li> </ul> <p>Two actions of similar type are needed to perform the <b>Reset</b> action. One is to reset Rapid variable <b>partsReady</b> to 0, the other is to reset Rapid variable <b>cycleTime</b> to 0.</p>

- 10 Drag a **PictureBox** control from the **General** category; place it on the design surface and set the following values in the **Properties** window:

Property	Value
Location	177,28
Size	284,359
SizeMode	StretchImage
Image	FlexArcCell.GIF



#### Note

You can find the graphic (.GIF) files at *C:\MyDocuments\RobotStudio\My ScreenMaker Projects\Tutorial\Images*.

- 11 Drag another **PictureBox** control from the **General** category; place it on the design surface and set the following values in the **Properties** window:

Property	Value
Location	237,31
Size	48,48
SizeMode	StretchImage
Image	RobotAtHome.GIF
AllowMultipleStates	True Select <b>Image</b> property from the <b>StatesEditor</b> dialog box.
SelectedStateValue	DI_RobotAtHome
States	Link State{0} to <i>RobotAtHome_gray.GIF</i> Link State{1} to <i>RobotAtHome.GIF</i>

Continues on next page

## 15 ScreenMaker tab

### 15.5.3 Designing the screen

Continued



#### Note

Add **AllowMultipleStates** option to the PictureBox control. The objective is to have an image that changes when an I/O signal changes.

For more information on how to use **AllowMultipleStates** for PictureBox control, see [Picture object and changing images due to I/O on page 532](#).

- 12 Drag a Button control from the General category; place it on the design surface and set the following values in the Properties window:

Property	Value
Location	486,66
Size	116,105
Text	Start
Font	TpsFont20b
BackColor	LimeGreen
Enabled	Link to DI_RobotAtHome

Perform the following for the **Start** button:

Step	Action
1	Double-click the button <b>Start</b> or click the <b>Smart tag</b> and select <i>Define Actions when clicked</i> . The <b>Events Panel</b> dialog box appears which is used to define the actions for Events.
2	In the <b>Events Panel</b> dialog box, click <b>Add Action</b> ; point to <b>Rapid Data</b> and select <b>Write a Rapid Data</b> . The <b>Action Parameters</b> dialog box appears.
3	In the <b>Action Parameters</b> dialog box, assign Rapid data to the following value and click OK. <ul style="list-style-type: none"><li>• T_ROB1.MainModule.JobProduce to JobProduce</li></ul>

- 13 Drag a Button control from the General category; place it on the design surface and set the following values in the Properties window:

Property	Value
Location	486,226
Size	116,105
Text	Stop
Font	TpsFont20b
BackColor	LimeGreen
Enabled	Link to DI_PRODUCE

Perform the following for the **Stop** button:

Step	Action
1	Double-click the button <b>Stop</b> or click the <b>Smart tag</b> and select <i>Define Actions when clicked</i> . The <b>Events Panel</b> dialog box appears which is used to define the actions for Events.
2	In the <b>Events Panel</b> dialog box, click <b>Add Action</b> ; point to <b>Rapid Data</b> and select <b>Write a Rapid Data</b> . The <b>Action Parameters</b> dialog box appears.

Continues on next page

Step	Action
3	In the <b>Action Parameters</b> dialog box, assign Rapid data to the following value and click <b>OK</b> . <ul style="list-style-type: none"> <li>T_ROB1.MainModule.JobIdle to JobIdle</li> </ul>

- 14 Drag a Button control from the General category; place it on the design surface and set the following values in the Properties window:

Property	Value
Location	274,246
Size	111,47
Text	Bull's Eye
Font	TpsFont14b
Enabled	Link to DI_RobotAtHome
AllowMultipleStates	True Select <b>BackColor</b> property from the <b>StatesEditor</b> dialog box
SelectedStates	DI_RobotAtBull'sEye
States	Link State{0} to <i>Red</i> Link State{1} to <i>Green</i>

Perform the following for the **Bull's Eye** button:

Step	Action
1	Double-click the button <b>Bull's Eye</b> or click the <b>Smart tag</b> and select <i>Define Actions when clicked</i> . The <b>Events Panel</b> dialog box appears which is used to define the actions for Events.
2	In the <b>Events Panel</b> dialog box, click <b>Add Action</b> ; point to <b>Rapid Data</b> and select <b>Write a Rapid Data</b> . The <b>Action Parameters</b> dialog box appears.
3	In the <b>Action Parameters</b> dialog box, assign Rapid data to the following value and click <b>OK</b> . <ul style="list-style-type: none"> <li>T_ROB1.MainModule.JobBulls to JobBulls</li> </ul>

- 15 Drag a Button control from the General category; place it on the design surface and set the following values in the Properties window:

Property	Value
Location	274,324
Size	111,47
Text	Service
Font	TpsFont14b
Enabled	Link to DI_RobotAtHome
AllowMultipleStates	True Select <b>BackColor</b> property from the <b>StatesEditor</b> dialog box
SelectedStates	DI_RobotAtService
States	Link State{0} to <i>Red</i> Link State{1} to <i>Green</i>

Continues on next page

## 15 ScreenMaker tab

---

### 15.5.3 Designing the screen

*Continued*

Perform the following for the **Service** button:

Step	Action
1	Double-click the button <b>Service</b> or click the <b>Smart tag</b> and select <i>Define Actions when clicked</i> . The <b>Events Panel</b> dialog box appears which is used to define the actions for Events.
2	In the <b>Events Panel</b> dialog box, click <b>Add Action</b> ; point to <b>Rapid Data</b> and select <b>Write a Rapid Data</b> . The <b>Action Parameters</b> dialog box appears.
3	In the <b>Action Parameters</b> dialog box, assign Rapid data to the following value and click <b>OK</b> . <ul style="list-style-type: none"><li>• <b>T_ROB1.MainModule.JobService</b> to <b>JobService</b></li></ul>

## 15.5.4 Building and deploying the project

### Procedure

- 1 From the ScreenMaker ribbon, click Build.  
For more information on building the project, see [Building a project on page 518](#).
- 2 From the ScreenMaker ribbon, click Deploy.  
For more information on deploying the project, see [Deploying to controller on page 519](#).
- 3 In RobotStudio, press **Ctrl+F5** to launch the Virtual Flexpendant and click FlexArc Operator Panel to open the GUI.



#### Note

Ensure that you start the RAPID execution and switch the controller into Auto mode.

**This page is intentionally left blank**

# Index

## A

- ABB library, 204
- action instruction
  - about, 28
- Activate RobotStudio
  - Automatic activation, 42
  - Manual activation, 43
- Add Controller, 358
- add to path, 447
- Adjust Robtargets, 432
- alerts
  - activate, 142
- align frame orientation, 448
- align target orientation, 449
- Application grants, 403
- Application Variables, 525
  - create, delete, rename application variables, 525
- attach to object, 450
- Authenticate, 384
  - Edit User Accounts, 384
  - Login as a Different User, 384
  - Log in as Default User, 384
  - Log off, 384
  - Log off all controllers, 384
  - UAS Grant Viewer, 384
- auto configuration, 451
- AutoPath, 225

## B

- Backup, 367
  - create backup, 367
- Back up
  - restore backup, 369
- browser
  - Layout, 49
  - Modeling, 52
  - Paths & Targets, 50

## C

- CAD file
  - troubleshoot and optimize, 90
- CAD formats
  - convert, 89
- check reachability, 453
- collision
  - detection, 137
  - sets, 137
- configuration
  - robot axis, 35
- configuration editor, 372
  - instance editor, 373
- configuration file, 182
- configuration monitoring
  - about, 36
- Configure data binding, 526
  - Using Binding menu, 526
  - Using Smart tag, 526
- confJ
  - about, 36
- ConfL
  - about, 36
- Connecting a controller, 518
- Controller grants, 401
  - Backup and save, 401

- Calibration, 402
- Delete log, 402
- Edit RAPID code, 401
- Execute program, 401
- Full access, 401
- I/O write access, 401
- Manage UAS settings, 401
- Modify configuration, 401
- Modify controller properties, 402
- Modify current value, 401
- Program debug, 402
- Read access to controller disks, 402
- Safety Controller, 402
- Write access to controller disks, 402

- controller menu, 182
- Controller Shutdown, 407
- controller status window, 58
  - access, 59
  - controller name, 58
  - controller state, 58
  - logged on as, 59
  - operating mode, 58
  - program execution state, 58
  - system name, 58
- controller system
  - create, 158
- controller world coordinate system, 31
- control panel, 406
  - enable device, 406
  - manual full speed, 406
  - motors on, 406
  - operation mode, 406
  - release device, 406
  - reset emergency stop, 406
- convert frame to workobject, 455
- coordinate systems, 29
- cycle time
  - measure, 143

## D

- data binding
  - Controller object data binding, 527
- Data Binding, 526
  - Application variable data binding, 528
- data declaration, 26
- detach, 458
- detecting collision, 138
- device browser, 390

## E

- element
  - select, 70
- event
  - create, 140
- events, 360
- external axis
  - program, 131

## F

- File transfer, 385
  - Controller explorer, 386
  - PC explorer, 386
- FlexPendant Viewer, 387
- frame
  - converting to workobject, 104
  - creating by points, 104
- Frame

- create, 213
- create from three points, 214
- frames, 29
- function, 26

## G

- geometry
  - troubleshoot and optimize, 90
- Go Offline, 392
- Grants, about, 157
- Grants, give to groups, 399
- graphics window, 69
- Group, about, 156
- group, add, 398
- group, add user, 397
- group, remove, 399
- group, rename, 398
- Groups, give grants, 399

## H

- handle events, 185
  - date and time, 186
  - event category, 186
  - event code, 186
  - event description, 187
  - event log list, 185
  - event title, 186
  - event type, 185
  - manage events, 187
  - retrieve controller events, 187
  - sequential number, 186

## I

- I/O
  - set, 141
- I/O system, 361
  - I/O signals, 178
  - input signals, 178
  - output signals, 178
  - simulated signals, 178
  - virtual signals, 178
- import, 87
- import geometry, 212
- import library, 205
- instruction, 26
  - about, 28
- item
  - select, 70

## J

- jog
  - mechanism, 105
  - robot, 105
  - several mechanisms, 105
- jog reorient, 248
- jointtarget
  - creating, 106

## K

- keyboard shortcuts, 72
  - General commands, 72

## L

- Layout browser, 49
- LED, 505
- library
  - troubleshoot and optimize, 90

- Load Parameters, 374
- local coordinate system
  - set, 96
- local origin
  - set, 96

## M

- Manage ScreenMaker project
  - Close ScreenMaker, 519
  - ScreenMaker Doctor, 14, 503, 529
- Manage ScreenMaker Project, 507
  - Close project, 519
  - Create project, 507
  - Load project, 508
  - Save project, 508
- Manage Screens, 509
- Managing ScreenMaker Projects
  - Build project, 518
- MediaPool, 25
- Mirror, 473
- Modeling browser, 52
- Modify project properties, 517
- module, 26
- move instruction
  - about, 28
- Move instruction
  - teach, 235
- MoveJ
  - teach, 235
- MoveL
  - teach, 235
- MultiMove
  - programming workflow, 124

## N

- near-miss detection, 138
- network settings, 153
  - firewall settings, 153
  - local network connection, 153
  - remote network connection, 153
  - service port connection, 153

## O

- object
  - select, 70
  - set local origin, 96
  - troubleshoot and optimize, 90
- Offline and Online browser, 53
- Online Monitor, 393
- operator window, 60
  - enabling operator window, 60
  - show virtual operator window, 60
- orientations, 111
  - align target, 113
  - copy and apply, 114
  - target normal to surface, 112
  - unordered, 111
- output window, 57
  - event types, 57

## P

- pack, unpack, 146
- part
  - set local origin, 96
- Password, change for user, 397
- path, 108
  - about, 28

- compensating, 109
- creating, 108
- creating from curve, 108
- reversing, 108
- rotating, 109
- setting axis configuration, 108
- translating, 109
- Paths & Targets browser, 50
- Placing an item, 484
  - Frame, 484
  - One Point, 484
  - Three Points, 484
  - Two Frames, 484
  - Two Points, 484
- positioner.program, 131
- procedure, 26
- process time
  - measure, 143
- program
  - copy, 145
- programming
  - overview, 103
- Program Pointer, 436
- Properties, 389
  - Device Browser, 390
  - Renaming the controller, 389
  - Save System Diagnostics, 391
  - Set controller ID, 389
  - Set date and time, 389
  - View controller and system properties, 390
- Properties Window
  - Event Help panel, 506
  - Graphical Component Name panel, 506
  - Properties window toolbar, 506
  - Table panel, 506
- Property editor, 293

## R

- RAPID
  - concepts, 26
  - copy program, 145
- RAPID Data Editor, 425
- RAPID Editor, 416
- RAPID instructions, 115
- RAPID Profiler, 438
- RAPID task, 428
- RAPID Watch window, 440
- reachability
  - test, 122
- Relation, 376
- Release Write Access, 383
- remote subnet, 154
- Request Write Access, 382
- Restore, 369
- robot
  - programming overview, 103
- Robot system button, 206
  - adding an existing system, 207
  - adding a template system, 207
  - conveyor setup, 207
  - create system from layout, 206
  - remove objects from conveyor, 208
- RobotWare, 24
  - license key, 24
- RobotWare system, 24
- routine, 26
- Run mode, 431

- continuous, 431
- single, 431

## S

- safety, 19
- safety configuration, 381
- set task frame, 408
- Signal Analyzer, 345
- signals
  - set, 141
- simulate
  - alerts, 142
  - create event, 140
  - measure process time, 143
  - set signals, 141
  - TCP trace, 142
- simulation, 135, 137
- simulation control, 340
- Simulation watch, 294
  - Break condition, 295
- station
  - build workflow, 75
  - pan, 69
  - rotate, 69
  - zoom, 69
- station world coordiante system, 29
- Stopwatch, 344
- switch, 505
- synchronization, 134
  - station to VC, 134
  - VC to station, 134
- system
  - create, 158
  - create with positioner, 175
  - RobotWare, 24
- System Builder, 158, 371
  - about virtual and real systems, 158
  - building new system, 161
  - copying system, 169
  - create boot media, 172
  - create system from backup, 170
  - download a system to controller, 171
  - modify controller system, 165
  - viewing system properties, 160
- System Configuration, 409
  - controller values, 409
  - stored station values, 410
  - used current station values, 410
- system parameters, 179
  - editing parameters, 180
  - load parameters, 183
  - save system parameters, 182

## T

- target, 106
  - about, 28
  - creating, 106
  - modifying, 106
  - modifying with ModPos, 106
  - removing unused, 107
  - renaming, 106
  - teaching, 106
- Targets on Edge, 222
- TCP, 29
- TCP trace
  - activate, 142
- ToolBox

- ActionTrigger, 504
- BarGraph, 504
- CheckBox, 504
- ComboBox, 505
- CommandBar, 505
- ConditionalTrigger, 505
- ControllerModeStatus, 505
- DataEditor, 505
- Graph, 505
- GroupBox, 505
- ListBox, 505
- NumEditor, 505
- NumericUpDown, 505
- Panel, 505
- PictureBox, 505
- RapidExecutionStatus, 505
- RunRoutineButton, 505
- TabControl, 505
- Tool Center Point coordinate system, 29
- tooldata, 95
- tools, 95
- TpsLabel, 505
- track
  - program, 131
- Transfer, 376
- trap, 26

### U

- UAS Grant Viewer, 400

- UCS, 34
- unpack, 192
- User, about, 156
- User, add, 396
- User, add to group, 397
- User, change password, 397
- User, change user name, 397
- User, enable and disable, 397
- User, remove, 397
- User account, 395
  - User tab, 395
- user coordinate system, 34

### V

- VariantButton, 505
- viewpoint, 253
  - create, 253
  - move to viewpoint, 254
  - viewpoint functions, 253
- Virtual FlexPendant, 405
- Virtual FlexPendant operator window, 60

### W

- workobject
  - creating, 104
  - modifying, 104
- WorkObject, 34
- workobjects, 104
- world coordinate system, 29



# Contact us

## **ABB AB**

**Discrete Automation and Motion  
Robotics**

S-721 68 VÄSTERÅS, Sweden

Telephone +46 (0) 21 344 400

## **ABB AS, Robotics**

**Discrete Automation and Motion**

Nordlysvegen 7, N-4340 BRYNE, Norway

Box 265, N-4349 BRYNE, Norway

Telephone: +47 51489000

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

5 Lane 369, ChuangYe Road

KangQiao Town, PuDong District

SHANGHAI 201319, China

Telephone: +86 21 6105 6666

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