

# EPSON

EPSON RC+ 6.0

Ver.6.2

## *User's Guide*

Project Management and Development

Rev.5

EM15XS3084F



EPSON RC+ 6.0 (Ver.6.2)

# *User's Guide*

Rev.5

Copyright © 2011-2015 SEIKO EPSON CORPORATION. All rights reserved.

## FOREWORD

Thank you for purchasing our robot products.

This manual contains the information necessary for the correct use of the Manipulator.

Please carefully read this manual and other related manuals before installing the robot system.

Keep this manual handy for easy access at all times.

## WARRANTY

The robot and its optional parts are shipped to our customers only after being subjected to the strictest quality controls, tests, and inspections to certify its compliance with our high performance standards.

Product malfunctions resulting from normal handling or operation will be repaired free of charge during the normal warranty period. (Please ask your Regional Sales Office for warranty period information.)

However, customers will be charged for repairs in the following cases (even if they occur during the warranty period):

1. Damage or malfunction caused by improper use which is not described in the manual, or careless use.
2. Malfunctions caused by customers' unauthorized disassembly.
3. Damage due to improper adjustments or unauthorized repair attempts.
4. Damage caused by natural disasters such as earthquake, flood, etc.

### Warnings, Cautions, Usage:

1. If the robot or associated equipment is used outside of the usage conditions and product specifications described in the manuals, this warranty is void.
2. If you do not follow the WARNINGS and CAUTIONS in this manual, we cannot be responsible for any malfunction or accident, even if the result is injury or death.
3. We cannot foresee all possible dangers and consequences. Therefore, this manual cannot warn the user of all possible hazards.

## TRADEMARKS

Microsoft, Windows, and Windows logo are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other brand and product names are trademarks or registered trademarks of the respective holders.

## TRADEMARK NOTATION IN THIS MANUAL

Microsoft® Windows® XP Operating system

Microsoft® Windows® Vista Operating system

Microsoft® Windows® 7 Operating system

Throughout this manual, Windows XP, Windows Vista, and Windows 7 refer to above respective operating systems. In some cases, Windows refers generically to Windows XP, Windows Vista, and Windows 7.

## NOTICE

No part of this manual may be copied or reproduced without authorization.

The contents of this manual are subject to change without notice.

Please notify us if you should find any errors in this manual or if you have any comments regarding its contents.

## INQUIRIES

Contact the following service center for robot repairs, inspections or adjustments.

If service center information is not indicated below, please contact the supplier office for your region.

Please prepare the following items before you contact us.

- Your controller model and its serial number
- Your manipulator model and its serial number
- Software and its version in your robot system
- A description of the problem

## SERVICE CENTER

--

## MANUFACTURER

### **Seiko Epson Corporation**

Toyoshina Plant

Robotics Solutions Operations Division

6925 Toyoshina Tazawa,

Azumino-shi, Nagano, 399-8285

Japan

TEL : +81-(0)263-72-1530

FAX : +81-(0)263-72-1495

## SUPPLIERS

### North & South America **Epson America, Inc.**

Factory Automation/Robotics

18300 Central Avenue

Carson, CA 90746

USA

TEL : +1-562-290-5900

FAX : +1-562-290-5999

E-MAIL : info@robots.epson.com

### Europe

#### **Epson Deutschland GmbH**

Factory Automation Division

Otto-Hahn-Str.4

D-40670 Meerbusch

Germany

TEL : +49-(0)-2159-538-1391

FAX : +49-(0)-2159-538-3170

E-MAIL : robot.infos@epson.de

### China

#### **Epson (China) Co., Ltd.**

Factory Automation Division

7F, Jinbao Building No. 89, Jinbao Street,

Dongcheng District, Beijing,

China, 100005

TEL : +86-(0)-10-8522-1199

FAX : +86-(0)-10-8522-1120

### Taiwan

#### **Epson Taiwan Technology & Trading Ltd.**

Factory Automation Division

14F, No.7, Song Ren Road, Taipei 110,

Taiwan, ROC

TEL : +886-(0)-2-8786-6688

FAX : +886-(0)-2-8786-6677

Korea	<b>Epson Korea Co., Ltd.</b> Marketing Team (Robot Business) 27F DaeSung D-Polis A, 606 Seobusaet-gil, Geumcheon-gu, Seoul, 153-803 Korea TEL : +82-(0)-2-3420-6692 FAX : +82-(0)-2-558-4271
Southeast Asia	<b>Epson Singapore Pte. Ltd.</b> Factory Automation System 1 HarbourFront Place, #03-02, HarbourFront Tower One, Singapore 098633 TEL : +65-(0)-6586-5696 FAX : +65-(0)-6271-3182
India	<b>Epson India Pvt. Ltd.</b> Sales & Marketing (Factory Automation) 12th Floor, The Millenia, Tower A, No. 1, Murphy Road, Ulsoor, Bangalore, India 560008 TEL : +91-80-3051-5000 FAX : +91-80-3051-5005
Japan	<b>Epson Sales Japan Corporation</b> Factory Automation Systems Department Nishi-Shinjuku Mitsui Bldg. 6-24-1 Nishishinjuku, Shinjuku-ku, Tokyo 160-8324 Japan TEL : +81-(0)3-5321-4161





<b>1. Introduction</b>	<b>1</b>
1.1 Welcome to EPSON RC+ 6.0.....	1
1.2 System Overview.....	1
1.2.1 RC620 Controller.....	2
1.2.2 Software.....	2
1.2.3 System Block Diagram.....	3
1.3 Options.....	4
1.4 EPSON RC+ 5.0 Ver.5.x Users.....	4
1.5 EPSON RC+ 3.x and 4.x Users.....	4
1.6 SPEL for Windows Users.....	4
1.7 Documentation.....	5
<b>2. Safety</b>	<b>6</b>
2.1 Overview.....	6
2.2 Definitions.....	6
2.2.1 Robot Power.....	6
2.2.2 Safeguard.....	7
2.2.3 Operation Modes.....	7
2.2.4 Start Mode.....	7
2.2.5 Changing Operation Mode.....	8
2.2.6 Emergency Stop.....	8
2.2.7 Teach Control Device.....	8
2.3 Safety-related Requirements.....	9
2.4 Installation and Design Precautions.....	10
2.4.1 Designing a Safe Robot System.....	10
2.4.2 Robot System Installation, Start-up, and Testing.....	13
2.5 Precautions regarding Robot Operation.....	15
2.5.1 General Precautions.....	15
2.5.2 Automatic Operation.....	15
2.5.3 Teaching Robot Points.....	15
2.5.4 Return to Automatic Operation.....	16
2.5.5 Program Verification.....	16
2.5.6 Troubleshooting.....	16
2.5.7 Maintenance.....	16
2.5.8 Backup of Projects and Controller.....	17
2.6 End User Instruction Manual.....	17
2.7 End User Training.....	17

<b>3. Getting Started</b>	<b>18</b>
3.1 Hardware Installation .....	18
3.2 Software Installation.....	18
3.3 Installation for Offline Development.....	18
3.4 Windows Security Administration.....	19
<b>4. Operation</b>	<b>20</b>
4.1 Simple Mode .....	20
4.2 System Power Up Procedure .....	20
4.2.1 Startup Sequence .....	20
4.2.2 Startup Configuration .....	23
4.2.3 Start Mode .....	23
4.2.4 Start Mode Dialog .....	24
4.2.5 Start Mode : Program .....	24
4.2.6 Start Mode : Auto .....	25
4.2.7 Auto Start .....	25
4.2.8 Using Monitor Mode .....	26
4.2.9 Windows Login .....	26
4.2.10 Command Line Options .....	27
4.2.11 Using Command Line Options .....	28
4.3 Writing your first Program .....	29
4.4 System Shutdown Procedure .....	33
<b>5. The EPSON RC+ 6.0 GUI</b>	<b>35</b>
5.1 GUI Overview.....	35
5.2 Project Explorer Pane .....	36
5.2.1 Context Menu .....	36
5.3 Status Window Pane.....	36
5.4 Status Bar .....	37
5.5 Online Help .....	37
5.6 File Menu.....	38
5.6.1 New Command .....	38
5.6.2 Open Command .....	39
5.6.3 Close Command .....	39
5.6.4 Save Command .....	40
5.6.5 Save As Command .....	40
5.6.6 Restore Command.....	40
5.6.7 Rename Command.....	40
5.6.8 Delete Command .....	41
5.6.9 Import Command .....	41

5.6.10	Print Command.....	42
5.6.11	Exit Command .....	43
5.7	Edit Menu.....	44
5.7.1	Undo Command .....	44
5.7.2	Redo Command .....	44
5.7.3	Cut Command.....	44
5.7.4	Copy Command.....	44
5.7.5	Paste Command.....	45
5.7.6	Find Command .....	45
5.7.7	Find Next Command.....	46
5.7.8	Replace Command.....	46
5.7.9	Select All Command .....	46
5.7.10	Indent Command .....	47
5.7.11	Outdent Command .....	47
5.7.12	Comment Block Command.....	47
5.7.13	Uncomment Block Command.....	47
5.7.14	Go To Definition Command .....	47
5.8	View Menu .....	48
5.8.1	Project Explorer Command .....	48
5.8.2	Status Window Command .....	48
5.8.3	System History Command.....	48
5.9	Project Menu .....	50
5.9.1	New Command.....	50
5.9.2	Open Command .....	51
5.9.3	Recent Projects Submenu.....	52
5.9.4	Close Command.....	52
5.9.5	Edit Command.....	52
5.9.6	Save Command.....	54
5.9.7	Save As Command.....	54
5.9.8	Rename Command .....	55
5.9.9	Import Command.....	56
5.9.10	Copy Command.....	60
5.9.11	Delete Command.....	61
5.9.12	Build Command .....	61
5.9.13	Rebuild Command.....	61
5.9.14	Properties Command.....	62
5.10	Run Menu .....	72
5.10.1	Run Window Command.....	72

5.10.2	Operator Window Command.....	72
5.10.3	Step Into Command .....	72
5.10.4	Step Over Command .....	72
5.10.5	Walk Command.....	73
5.10.6	Resume Command .....	73
5.10.7	Stop Command .....	73
5.10.8	Toggle Breakpoint Command.....	73
5.10.9	Clear All Breakpoints Command .....	74
5.10.10	Display Variables Command .....	74
5.10.11	Call Stack Command.....	75
5.11	Tools Menu.....	76
5.11.1	Robot Manager Command.....	76
5.11.2	Command Window Command .....	104
5.11.3	I/O Monitor Command.....	105
5.11.4	Task Manager Command .....	107
5.11.5	Macros Command.....	109
5.11.6	I/O Label Editor Command.....	110
5.11.7	User Error Editor Command.....	112
5.11.8	Controller Command .....	113
5.12	Setup Menu.....	117
5.12.1	System Configuration Command .....	117
5.12.2	Preferences Command .....	133
5.12.3	Options Command .....	140
5.13	Window Menu .....	141
5.13.1	Cascade Command .....	141
5.13.2	Tile Vertical Command .....	141
5.13.3	Tile Horizontal Command.....	142
5.13.4	Arrange Icons Command .....	142
5.13.5	Close All Command.....	142
5.13.6	1, 2, 3 Command.....	143
5.13.7	Windows Command .....	143
5.14	Help Menu.....	144
5.14.1	How Do I Command.....	144
5.14.2	Contents Command .....	144
5.14.3	Index Command.....	145
5.14.4	Search Command .....	145
5.14.5	Manuals Submenu .....	146
5.14.6	About EPSON RC+ 6.0 Command .....	146

<b>6 The SPEL<sup>+</sup> Language</b>	<b>147</b>
6.1 Overview .....	148
6.2 Program Structure .....	148
6.2.1 What is a SPEL <sup>+</sup> program?.....	148
6.2.2 Calling functions .....	148
6.3 Commands and Statements .....	149
6.4 Function and Variable Names .....	149
6.5 Data Types .....	150
6.6 Operators .....	150
6.7 Working with Variables .....	151
6.7.1 Variable scopes .....	151
6.7.2 Local variables .....	151
6.7.3 Module variables .....	151
6.7.4 Global variables .....	152
6.7.5 Global Preserve variables .....	152
6.7.6 Arrays .....	153
6.7.7 Initial values .....	153
6.7.8 Clearing variables .....	153
6.8 Working with Strings .....	154
6.9 Working with Files .....	155
6.10 Multi-statements .....	156
6.11 Labels .....	156
6.12 Comments .....	157
6.13 Error Handling .....	157
6.14 Multi-tasking .....	159
6.15 Using Multiple Robots .....	160
6.16 Robot Coordinate Systems .....	161
6.16.1 Overview.....	161
6.16.2 Robot Coordinate System .....	161
6.16.3 Local Coordinate Systems.....	165
6.16.4 Tool Coordinate Systems.....	165
6.16.5 ECP Coordinate Systems (Option).....	166
6.17 Robot Arm Orientations .....	167
6.17.1 SCARA robot arm orientations .....	167
6.17.2 6-axis robot arm orientations .....	168
6.17.3 RS series arm orientations .....	172
6.18 Robot Motion Commands.....	174
6.18.1 Homing the robot .....	174
6.18.2 Point to point motion.....	174
6.18.3 Linear motion.....	174

6.18.4	Curves .....	174
6.18.5	Joint motion .....	175
6.18.6	Controlling position accuracy .....	175
6.18.7	CP Motion Speed / Acceleration and Tool Orientation .....	176
6.18.8	PTP Speed / Acceleration for Small Distances .....	176
6.19	Working with Robot Points .....	177
6.19.1	Defining points .....	177
6.19.2	Referencing points by point label .....	177
6.19.3	Referencing points with variables .....	178
6.19.4	Using points in a program .....	178
6.19.5	Importing points into program .....	178
6.19.6	Saving and loading Points .....	178
6.19.7	Point attributes .....	179
6.19.8	Extracting and setting point coordinates .....	180
6.19.9	Alteration of points .....	180
6.20	Input and output control .....	181
6.20.1	Hardware I/O .....	181
6.20.2	Memory I/O .....	181
6.20.3	I/O Commands .....	181
6.21	Using Traps .....	183
6.21.1	Cautions of Trap when it triggers the system condition .....	183
6.22	Special Tasks .....	183
6.22.1	Precautions to Use the Special Task .....	183
6.22.2	NoPause/NoEmgAbort Task Specification .....	185
6.22.3	NoPause/NoEmgAbort Task Example .....	186
6.23	Background Task .....	187
6.23.1	Primary features of background task .....	187
6.23.2	Setup and start the background task .....	187
6.23.3	Holding background task (from being activated) .....	188
6.23.4	Commands that will cause error in the background task .....	190
6.23.5	Background task and Remote control .....	190
6.24	Predefined Constants .....	191
6.25	Calling Native Functions in Dynamic Link Libraries .....	195
<b>7</b>	<b>Building SPEL<sup>+</sup> Applications</b> .....	<b>200</b>
7.1	Designing Applications .....	200
7.1.1	Creating the simplest application .....	200
7.1.2	Application layout .....	200
7.1.3	Auto start at power up .....	201

7.2	Managing Projects .....	202
7.2.1	Overview .....	202
7.2.2	Creating a new project .....	203
7.2.3	Configuring a project .....	203
7.2.4	Building a project .....	204
7.2.5	Backing up a project .....	204
7.3	Editing Programs .....	205
7.3.1	Program rules .....	206
7.3.2	Typing in program code .....	205
7.3.3	Syntax Help .....	206
7.3.4	Syntax Errors .....	207
7.4	Editing Points .....	207
7.5	Running and Debugging Programs .....	210
7.5.1	The Run Window .....	210
7.5.2	Debugging .....	212
7.6	The Operator Window .....	216
7.6.1	Operator window configuration .....	217
7.6.2	Auto start configuration .....	217
7.7	Using Remote Control .....	217
7.8	Using Encrypt Files .....	218
<b>8</b>	<b>PG Motion System .....</b>	<b>219</b>
8.1	Standard Motion System .....	219
8.2	RC620 Drive Module Software Configuration .....	219
8.3	PG Motion System .....	219
<b>9</b>	<b>Robot Configuration .....</b>	<b>220</b>
9.1	Setting the Robot Model .....	220
9.1.1	Adding the Standard robot .....	220
9.1.2	Calibrating a standard robot .....	221
9.1.3	Changing robot system parameters .....	221
9.1.4	Deleting a standard robot .....	222
9.2	Configuration of Additional Axes .....	223
9.2.1	Adding the additional S axis .....	223
9.2.2	Adding the additional T axis .....	224
9.2.3	Calibrating the additional axes .....	224
9.2.4	Changing the parameters of robot with additional axes installed .....	224
9.2.5	Differences of the standard robot and robot with additional axes .....	224

9.2.6	Deleting the additional axes .....	226
<b>10</b>	<b>Inputs and Outputs .....</b>	<b>227</b>
10.1	Overview .....	227
10.2	I/O Commands .....	227
10.3	I/O Configuration .....	228
10.4	Monitoring I/O .....	228
10.5	Virtual I/O .....	228
10.6	Fieldbus Master I/O .....	228
10.7	Fieldbus Slave I/O .....	228
<b>11</b>	<b>Remote Control .....</b>	<b>229</b>
11.1	Overview .....	229
11.2	Remote Control Input Output Configuration .....	230
11.3	Control Device Configuration .....	230
11.4	Auto Mode with Remote Control .....	231
11.5	Teach Mode with Remote Control .....	231
11.6	Debugging Remote Control .....	231
11.7	Remote Inputs .....	232
11.8	Remote Outputs .....	235
11.9	Remote Input Handshake Timing .....	237
<b>12</b>	<b>RS-232C Communications .....</b>	<b>240</b>
12.1	RS-232C Software Configuration .....	240
12.2	RS-232C Commands .....	241
<b>13</b>	<b>TCP / IP Communications .....</b>	<b>242</b>
13.1	TCP/IP Setup .....	242
13.1.1	Ethernet Hardware .....	242
13.1.2	IP Addresses .....	242
13.1.3	IP Gateway .....	243
13.1.4	Testing Windows TCP/IP setup .....	243
13.2	TCP/IP Software Configuration .....	244
13.3	TCP/IP Commands .....	244
<b>14</b>	<b>Security .....</b>	<b>245</b>
14.1	Overview .....	245
14.2	Installation .....	245
14.3	Security Configuration .....	245
14.4	Security Log Check .....	249
14.5	SPEL <sup>+</sup> Security Command .....	249



<b>15 Conveyor Tracking</b>	<b>250</b>
15.1 Overview .....	250
15.2 Conveyor Tracking Processes .....	252
15.3 Hardware Installation .....	253
15.4 System Structure .....	261
15.5 Conveyor Encoder Configuration .....	264
15.6 Verifying New Encoder Operation .....	265
15.7 Conveyor Tracking Commands .....	266
15.8 Key Terms .....	267
15.9 Creating Conveyors in a Project .....	268
15.10 Configuring Conveyors .....	269
15.11 Vision Conveyors .....	271
15.12 Sensor Conveyors .....	292
15.13 Multiple Conveyors and Robots .....	308
15.14 Adjusting the Z value .....	322
15.15 Starting Area .....	324
15.16 Queue Sorting .....	332
15.17 Abort Tracking .....	332
15.18 Conveyor Tracking with 6-Axis Robot .....	333
15.19 Tracking Mode .....	333
15.20 Manipulator Posture.....	334
<b>16 ECP Motion</b>	<b>335</b>
16.1 Overview .....	335
16.1.1 How to move the arm with ECP motion.....	336
<b>17 Force Sensing</b>	<b>338</b>
17.1 Overview .....	338
17.2 Specifications .....	338
17.3 Installation .....	339
17.4 Force Sensing Commands .....	344
17.5 Using the Force Sensing Trigger .....	345
<b>18 Real-Time I/O</b>	<b>346</b>
18.1 Overview .....	346
18.2 Specifications .....	346
18.3 Usage .....	348
<b>19 Additional Axis</b>	<b>351</b>
19.1 Overview .....	351
19.2 Specification .....	351
19.3 Usage .....	353

20	Installing Controller Options	355
21	Software License Agreement	356
Appendix A: Automatic Processing of Project Import		A-1
	Project for EPSON RC+ 5.* .....	A-1
	Project for EPSON RC+ 3.* / 4.* .....	A-1
	Project for SPEL for Windows 2.* .....	A-3
Appendix B:EPSON RC+ 6.0 Software		B-1
	EPSON RC+ 6.0 Software Installation .....	B-1
	EPSON RC+ 6.0 Software Update .....	B-1

# 1. Introduction

## 1.1 Welcome to EPSON RC+ 6.0

Welcome to the EPSON RC+ 6.0 Project Management and Development Environment. EPSON RC+ 6.0 is used to develop application software for the EPSON RC620 Robot Controller.

### EPSON RC+ 6.0 features

- Integrated application development environment
- SPEL<sup>+</sup> programming language  
A powerful, easy to use BASIC-like programming language that supports multi-tasking, robot motion control, I/O control, and networking.
- I/O systems including Digital I/O boards and Fieldbus I/O
- TCP/IP and RS-232 communications
- Background task  
Controls entire system
- Database access
- Vision Guide option  
Integrated vision robot guidance
- VB Guide option  
Enables you to control the system using standard programming environments includes including Microsoft Visual Basic and Microsoft Visual C++.
- Security option  
Allows you to administrate all EPSON RC+ users on your system. It also includes usage auditing, so you can track how many hours are spent using the system, and if changes were made.
- Conveyor Tracking option  
Enables one or more robots to pick parts from moving conveyors using vision or sensors.
- PG Motion System option  
Allows you to use third party motors and drivers to control auxiliary equipment such as XY tables, slides, etc.
- ECP option  
Supports CP motion relative to a fixed point.
- GUI Builder option  
Integrated GUI development tool
- Force Sensing option  
Allows a robot to use torque/force sensing and measurement

### 1.2 System Overview

The EPSON RC+ 6.0 software contains several components that enable you to control an entire robotic work cell.

EPSON RC+ 6.0 supports the EPSON RC620 Controller

#### 1.2.1 RC620 Controller

The RC620 Controller is a powerful robotic work cell controller that controls our robot (G series, RS series, PS series, C3 series, and EZ module X5 series).

##### Controller features

- Integral with PC and sophisticated yet achieving reliability and stability
- Built in Motion System  
The motion drive system can control up to 8 axes simultaneously and two robots, and can add up to two drive units
- Includes standard I/O
- Wide variety of options

For detailed information on the Controller, refer to the controller manual.

#### 1.2.2 Software

The EPSON RC+ 6.0 software comes pre-installed in the controller.

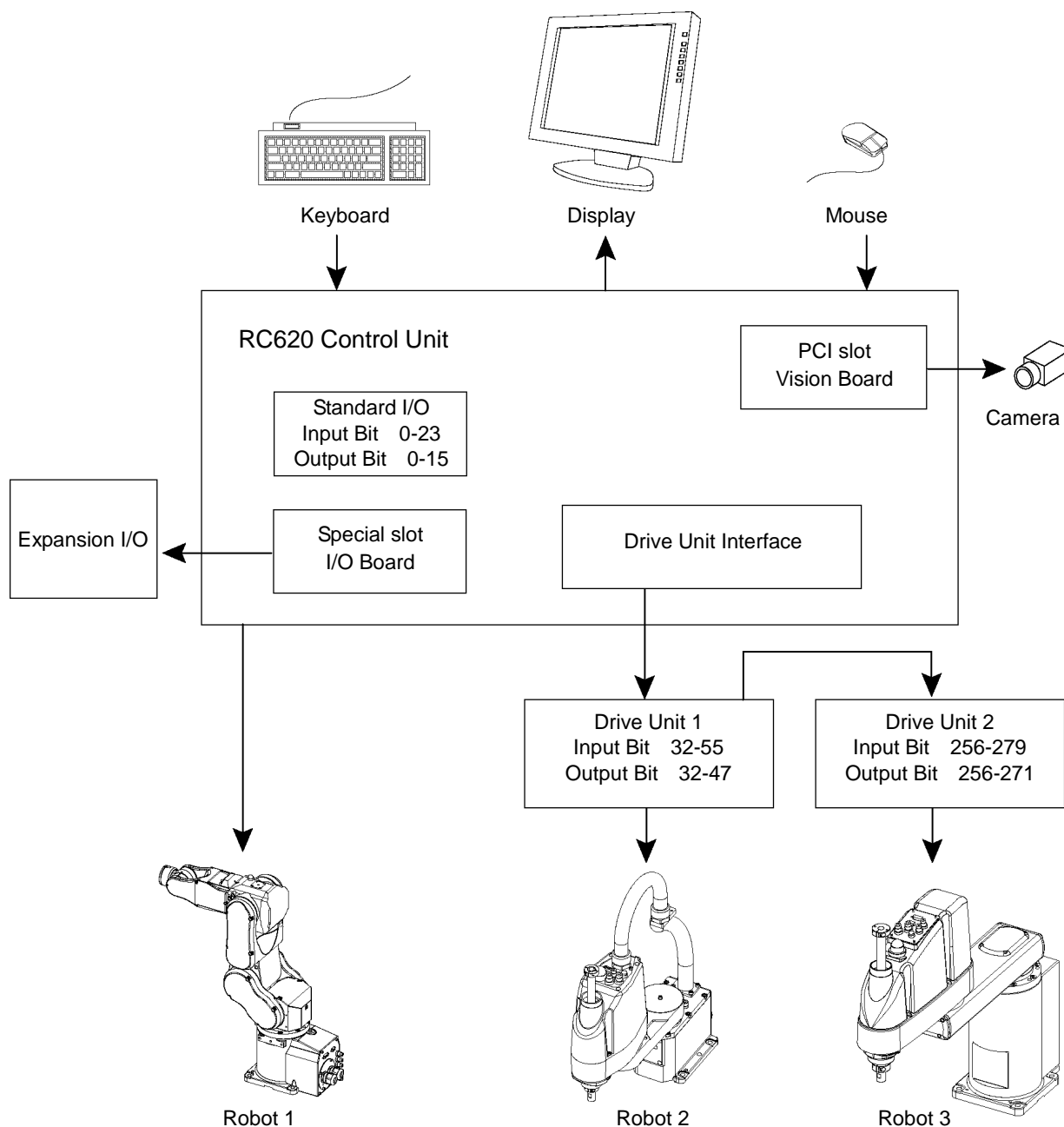
You can purchase options with the product or add them later.

The software can be installed on your PC using the installation disk supplied with the controller.

Using EPSON RC+ 6.0, you can develop application software for the SPEL+ language that runs in the RC620 controller.

### 1.2.3 System Block Diagram

The following system block diagram shows different methods for connecting a PC running EPSON RC+ 6.0 to one or more controllers.



### 1.3 Options

You can purchase options that must be enabled in the controller. EPSON RC+ 6.0 is used to enable these options. Refer to *18. Installing Controller Options* for details.

### 1.4 EPSON RC+ 5.0 Ver.5.x Users

EPSON RC+ 6.0 is compatible with EPSON RC+ 5.0 for the operation and language.

For EPSON RC+ 6.0, you can use all commands of EPSON RC+ 5.0.

You can use the current numbers for the I/O and communication port.

To enable the EPSON RC+ 5.x project in EPSON RC+ 6.0 environment, convert the project using Project menu | Import.

With above conversion, the entire project will be copied by EPSON RC+ 6.0.

`\EPSONRC50\Project directory` → `\EpsonRC60\Project directory`

Refer to Appendix A: Automatic Processing of Project Import for the details.

### 1.5 EPSON RC+ 3.x and 4.x Users

EPSON RC+ 6.0 is compatible with EPSON RC+ 3.x and 4.x for the operation.

For EPSON RC+ 6.0, there are new commands added to SPEL<sup>+</sup> language. Though there are also some commands deleted or amended, most commands are available.

To enable the project of EPSON RC+ 3.x or 4.x in EPSON RC+ 6.0 environment, convert the project using Project menu | Import.

With above conversion, the entire project will be copied by EPSON RC+ 6.0.

`\EPSONRC\Project directory` → `\EpsonRC60\Project directory`

Refer to Appendix A: Automatic Processing of Project Import for the details.

### 1.6 SPEL for Windows Users

EPSON RC+ 6.0 is compatible with SPEL for Windows 1.x and 2.x for the operation.

For EPSON RC+ 6.0, there are many new commands added to SPEL<sup>+</sup> language, which replaces SPEL. Also there are some commands deleted or amended.

To enable the project of SPEL for Windows 2.x in EPSON RC+ 6.0 environment, convert the project using Project menu | Import.

With above conversion, the file will be copied to a new directory or the program will optionally be converted by EPSON RC+ 6.0.

## 1.7 Documentation



All documentation is installed on the PC in PDF format.

To view manuals on the PC:

- Select Manuals from the Help Menu in EPSON RC+ 6.0
- From Windows desktop, click Start | Programs | EPSON RC+ 6.0

Available manuals are shown in the table below.

Title	Contents
EPSON RC+ 6.0 Users Guide	Information for the entire system
SPEL <sup>+</sup> Language Reference	Information for the SPEL <sup>+</sup> Language
Vision Guide 6.0	Information for options
Vision Guide 6.0 reference	
VB Guide 6.0	
GUI Builder 6.0	
Fieldbus IO	
PG Motion System	
Manipulator manual	Information for the purchased robot Each series has its own manual
Controller manual	Information for the purchased robot
Safety & Installation	Information for installing the robot system safely Paper manual will come with the product

<b>NOTE</b> 	The “NOTE” sections describe important information to be followed for operating the Robot system.
<b>TIP</b> 	The "TIP" sections describe hints for easier or alternative operations.




## 2. Safety

### 2.1 Overview

This chapter explains the important safety requirements for robotic systems using EPSON RC+ 6.0 and the RC620 Controller.

Installation of robots and robotic equipment should only be performed by qualified personnel in accordance with national and local codes. Please read and understand this entire chapter before using your EPSON RC+ 6.0 system.

Remember that safety is the most important consideration when designing and operating any robotic system.

 WARNING	This symbol indicates that a danger of possible serious injury or death exists if the associated instructions are not followed properly.
 WARNING	This symbol indicates that a danger of possible harm to people caused by electric shock exists if the associated instructions are not followed properly.
 CAUTION	This symbol indicates that a danger of possible harm to people or physical damage to equipment and facilities exists if the associated instructions are not followed properly.

### 2.2 Definitions

#### 2.2.1 Robot Power

The status of robot power is explained below in terms of restriction to operation:

**Operation-prohibited status:** Robot cannot be operated.

**Restricted (low power) status:** Robot can operate at low speed and low torque.

**Unrestricted (high power) status:** Robot can operate without restriction.

The robot will not operate regardless of the control actions taken by the operator when in the operation-prohibited state. During operation, when the safeguard circuit opens, the system will switch to operation-prohibited state.

The robot will operate at low speed and torque in the restricted state (low power). In the unrestricted state (high power), the robot will operate at the programmed speed and torque.

In the event that the robot should make an unexpected movement, the restricted state (low power) decreases operating speed allowing the operator to avoid danger. The torque is also decreased to minimize serious injury to the operator should one be struck by the robot. The maximum values of the decreased speed and torque are set according to the robot used and cannot be changed by the user.

As a safety precaution the initial power state of the robot will be set to either the restricted (low power) state or the operation-prohibited state. The system will not change to the unrestricted (high power) state if the appropriate procedures are not followed.

When the system is in restricted (low power) state or operation-prohibited state, a single failure will not cause a runaway action that surpasses the assigned speed or torque decrease. This is due to the multi-protect circuit and mutual monitoring circuit in the control system.



### 2.2.2 Safeguard

To ensure safe operation of the robotic work cell, you must install a safety system using safety doors, light curtains, safety floor mats, etc.



- There is a safeguard input circuit in the EMERGENCY connector on the controller that connects with the safety device interlock switch. In order to protect those working with the robot be sure that the interlock switch is connected and working properly.

If a closed safeguard is open during robot motion, the robot stops immediately and enters pause state. All robot motors are turned off. The descriptions below explain how the safeguard input works.

**Safeguard closed:** The safeguard input is turned ON. The robot can automatically operate in unrestricted (high power) state.

**Safeguard open:** The safeguard input is turned OFF, and the interlock function operates. The robot stops immediately, motors are turned off, and further operation is impossible until either the safeguard is closed or Teach mode is turned ON and the enable circuit is engaged.

For further details on the safeguard and interlock, refer to *2.4 Installation and Design Precautions* later in this chapter. For detailed wiring instructions, refer to the *RC620 Robot Controller manual, Setup & Operation: 8. EMERGENCY*.

### 2.2.3 Operation Modes

The operation mode is defined as the single control point for the controller, therefore you cannot use more than one operation mode at the same time.

There are three operation modes for the controller: AUTO, PROGRAM, and TEACH.

- AUTO operation modes allow you to execute programs in the controller when the safeguard is closed.
- PROGRAM operation mode allows you to execute and debug programs when the safeguard is closed.
- TEACH operation mode allows you to jog and teach the robot at slow speed while inside the safeguarded area.

### 2.2.4 Start Mode

The Start mode specifies the operation mode for EPSON RC+ 6.0 when it starts.

You can set the EPSON RC+ 6.0 to start in AUTO or PROGRAM mode.

For information on how to change the start mode, refer to *4. Operation*.

### 2.2.5 Changing Operation Mode

You can change from AUTO operation mode or PROGRAM operation mode to TEACH mode by setting the mode selector key switch on the Teach Pendant to the TEACH position.

When the mode selector key switch is changed back to AUTO, the operation mode is returned to previous operation mode (AUTO or PROGRAM).

The AUTO operation mode can be changed to PROGRAM mode during the EPSON RC+ 6.0 startup sequence. A password can be used to allow only certain personnel to change the startup operation mode.

When EPSON RC+ 6.0 starts in AUTO operation mode, the AUTO operation mode cannot be changed to PROGRAM operation mode after the system has started. To change the operation mode, restart the system and log into PROGRAM mode, then set the start mode again and restart EPSON RC+ 6.0.

For more information, refer to 4. *Operation*.

### 2.2.6 Emergency Stop

The controller is equipped with an emergency stop input terminal. If the normally closed emergency stop circuit is broken, the power supplied to all motors will be shut off (and enter servo-free status) and the robot will be stopped by dynamic braking.



- The path that the robot will follow from the time the emergency stop switch is pressed until the device stops, as well as the stop position itself, cannot be positively determined. In many cases, the stop position will not exceed the target position for the operation prior the emergency stop. Depending on the robot's loading condition and operation speed, overruns are inevitable. Taking this into consideration, be sure the layout for the peripheral equipment includes extra space.

For detailed wiring instructions, refer to the *RC620 Robot Controller manual, Setup & Operation: 8. EMERGENCY*.

### 2.2.7 Teach Control Device

Operators can use the TP1 teach pendant to operate the robot in the TEACH operation mode.

Refer to the *RC620 Robot Controller manual, Setup & Operation: 13. Teach pendant TP1* for operation instructions.

## 2.3 Safety-related Requirements

Specific tolerances and operating conditions for safety are contained in the manuals for the robot, controller and other devices. Be sure to read those manuals as well.

For the installation and operation of the robot system, be sure to comply with the applicable local and national regulations.

Robot systems safety standards and other examples are given in this chapter. Therefore, to ensure that safety measures are complete, please refer to the other standards listed as well.

(Note: The following is only a partial list of the necessary safety standards.)

ENISO10218-1	Robots for industrial environments -Safety requirements-Part 1: Robot
ENISO 12100-1	Safety of machinery - Basic concepts, general principles for design - Part 1: Basic terminology, methodology
ENISO 12100-2	Safety of machinery - Basic concepts, general principles for design - Part 2: Technical principles
ENISO 13849-1	Safety of machinery – Safety – related parts of control systems – Part 1: General principles for design
EN 60204-1	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
EN55011	Industrial, scientific and medical (ISM) radio-frequency equipment - Electromagnetic disturbance characteristics - Limits and methods of measurement
EN61000-6-2	Electromagnetic compatibility (EMC) -- Part 6-2: Generic standards - Immunity for industrial environments
ANSI/RIA R15.06	American National Standard for Industrial Robots and Robot Systems – Safety Requirements

### UL specification

Compatibility assessment of the UL-compliant model is performed according to the following standards.

UL1740 (Third Edition, Dated December 7, 2007)

ANSI/RIA R15.06-1999

NFPA 79 (2007 Edition)

CSA/CAN Z434-03 (February 2003)

CE Marking – Machinery Directive, Low Voltage Directive, EMC Directive

### 2.4 Installation and Design Precautions

#### 2.4.1 Designing a Safe Robot System

It is important to operate robots safely. It is also important for robot users to give careful consideration to the safety of the overall robot system design.

This section summarizes the minimum conditions that should be observed when using EPSON robots in your robot systems.

Please design and manufacture robot systems in accordance with the principles described in this and the following sections.

##### **Environmental Conditions**

Carefully observe the conditions for installing robots and robot systems that are listed in the “Environmental Conditions” tables included in the manuals for all equipment used in the system.

##### **System Layout**

When designing the layout for a robot system, carefully consider the possibility of error between robots and peripheral equipment. Emergency stops require particular attention, since a robot will stop after following a path that is different from its normal movement path. The layout design should provide enough margin for safety. Refer to the manuals for each robot, and ensure that the layout secures ample space for maintenance and inspection work.

When designing a robot system to restrict the area of motion of the robots, do so in accordance with the methods described in each robot manual. Utilize both software and mechanical stops as measures to restrict motion.

Install the emergency stop switch at a location near the operation unit for the robot system where the operator can easily press and hold it in an emergency.

Do not install the controller at a location where water or other liquids can leak inside the controller. In addition, never use liquids to clean the controller.

##### **Disabling Power to the System using lock out / tag out**

The power connection for the robot controller should be such that it can be locked and tagged in the off position to prevent anyone from turning on power while someone else is in the safeguarded area. For further details, refer to the section *Procedure of Lockout/Tagout* in the chapter *Safety Precautions* in the controller manual.

##### **End Effector Design**

Provide wiring and piping that will prevent the robot end effector from releasing the object held (the work piece) when the robot system power is shut off.

Design the robot end effector such that its weight and moment of inertia do not exceed the allowable limits. Use of values that exceed the allowable limits can subject the robot to excessive loads. This will not only shorten the service life of the robot but can lead to unexpectedly dangerous situations due to additional external forces applied to the end effector and the work piece.

Design the size of the end effector with care, since the robot body and robot end effector can interfere with each other.

### Peripheral Equipment Design

When designing equipment that removes and supplies parts and materials to the robot system, ensure that the design provides the operator with sufficient safety. If there is a need to remove and supply materials without stopping the robot, install a shuttle device or take other measures to ensure that the operator does not need to enter a potentially dangerous zone.

Ensure that an interruption to the power supply (power shutoff) of peripheral equipment does not lead to a dangerous situation. Take measures that not only prevent a work piece held from being released as mentioned in “End effector Design” but that also ensure peripheral equipment other than the robots can stop safely. Verify equipment safety to ensure that, when the power shuts off, the area is safe.

### Remote Control

To prevent operation by remote control from being dangerous, start signals from the remote controller are allowed only when the control device is set to REMOTE, TEACH mode is OFF, and the system is configured to accept remote signals. Also when remote is valid, motion command execution and I/O output are available only from remote. For the safety of the overall system, however, safety measures are needed to eliminate the risks associated with the start-up and shutdown of peripheral equipment by remote control.

### Emergency Stop

Each robot system needs equipment that will allow the operator to immediately stop the system’s operation. Install an emergency stop device that utilizes emergency stop input from the controller and all other equipment.

During an emergency stop, the power that is supplied to the motor driving the robot is shut off, and the robot is stopped by dynamic braking.

The emergency stop circuit should also remove power from all external components that must be turned off during an emergency. Do not assume that the robot controller will turn off all outputs if configured to. For example, if an I/O card is faulty, the controller cannot turn off a component connected to an output. The emergency stop on the controller is hardwired to remove motor power from the robot, but not external power supplies.

Do not press the Emergency Stop switch unnecessarily while the Manipulator is operating. Pressing the switch during the operation makes the brakes work. This will shorten the life of the brakes due to the worn friction plates.

Normal brake life cycle: About 2 years (when the brakes are used 100 times/day)

Before using the Emergency Stop switch, be aware of the followings.

- The Emergency Stop (E-STOP) switch should be used to stop the Manipulator only in case of emergencies.
- To stop the Manipulator operating the program except in emergency, use Pause (halt) or STOP (program stop) commands.  
Pause and STOP commands do not turn OFF the motors. Therefore, the brake does not function.
- For the Safeguard system, do not use the circuit for E-STOP.

For details of the Safeguard system, refer to the following manuals.

*Safety and Installation*                      2.6 Connection to EMERGENCY Connector

To check brake problems, refer to the following manuals.

*Manipulator Manual Maintenance*                      2.2.2 Inspection While the Power is ON  
(Manipulator is operating)

*Safety and Installation*                      5.2 Inspection Point - Inspection While the  
Power is ON (Manipulator is operating)

### **Safeguard System**

To ensure safety, a safeguard system should be installed for the robot system.

When installing the safeguard system, strictly observe the following points:

Refer to each robot manual, and install the safeguard system outside the maximum space. Carefully consider the size of the end effector and the work pieces to be held so that there will be no error between the moving parts and the safeguard system.

Manufacture the safeguard system to withstand calculated external forces (forces that will be added during operation and forces from the surrounding environment).

When designing the safeguard system, make sure that it is free of sharp corners and projections, and that the safeguard system itself is not a hazard.

Make sure that the safeguard system can only be removed by using a tool.

There are several types of safeguard devices, including safety doors, safety barriers, light curtains, safety gates, and safety floor mats. Install the interlocking function in the safeguard device. The safeguard interlock must be installed so that the safeguard interlock is forced to work in case of a device failure or other unexpected accident. For example, when using a door with a switch as the interlock, do not rely on the switch's own spring force to open the contact. The contact mechanism must open immediately in case of an accident.

Connect the interlock switch to the safeguard input of the drive unit's EMERGENCY connector. The safeguard input informs the robot controller that an operator may be inside the safeguard area. When the safeguard input is activated, the robot stops immediately and enters pause status, as well as either operation-prohibited status or restricted status (low power status).

Make sure not to enter the safeguarded area except through the point where the safeguard interlock is installed.

The safeguard interlock must be installed so that it can maintain a safe condition until the interlock is released on purpose once it initiates. The latch-release input is provided for the EMERGENCY connector on the Controller to release the latch condition of the safeguard interlock. The latch release switch of the safeguard interlock must be installed outside of the safeguarded area and wired to the latch-release input.

It is dangerous to allow someone else to release the safeguard interlock by mistake while the operator is working inside the safeguarded area. To protect the operator working inside the safeguarded area, take measures to lock out and tag out the latch-release switch.

### **Presence Sensing Device**

The above mentioned safeguard interlock is a type of presence sensing device, since it indicates the possibility of somebody being inside the safeguard system. When separately installing a presence sensing device, however, perform a satisfactory risk assessment and pay thorough attention to its dependability.

Here are precautions that should be noted:

- Design the system so that when the presence sensing device is not activated or a dangerous situation still exists that no personnel can go inside the safeguard area or place their hands inside it.
- Design the presence sensing device so that regardless of the situation the system operates safely.
- If the robot stops operating when the presence sensing device is activated, it is necessary to ensure that it does not start again until the detected object has been removed. Make sure that the robot cannot automatically restart.

**Resetting the Safeguard**

Ensure that the robot system can only be restarted through careful operation from outside the safeguarded system. The robot will never restart simply by resetting the safeguard interlock switch. Apply this concept to the interlock gates and presence sensing devices for the entire system.

**Robot Operation Panel**

The robot operation panel must not be located inside of the robot work envelope / workcell. Ensure that the robot system can be operated from outside of the safeguard.

## 2.4.2 Robot System Installation, Start-up, and Testing

**Installation**

When installing the robot and robot system, follow the instructions contained in each of the robot and robot system manuals.

**Start-up and Functional Testing**

If the safeguard system is not ready at the time of start-up and functional testing, specify an area to install the safeguard system (as a temporary measure) and then begin.

During start-up and functional testing, do not allow workers inside the safeguarded area until the safeguard function is activated.

Before start-up and functional testing, carefully read the related manuals and obtain a good understanding of safety-related precautions.

Before supplying the robot and robot system with power for the first time, verify the items listed below.

**Items to check before supplying with power**

- Prescribed bolts are securely tightened to the robot.
- Electrical connections are set up correctly, and power supply conditions (including voltage, frequency, and error level) are within the specified range.
- Compressed air source (if applicable) is properly connected.
- Peripheral devices are properly connected.
- Safety device is equipped with an interlock switch, and it functions properly.
- Operating environment conditions conform to the conditions specified in the robot and controller manuals.

**Items to check after supplying with power**

- Start/stop, mode selection, and other functions work properly.
- Moving axes operate normally, and that the area of motion is limited as stipulated in the specifications.
- Emergency stop circuit functions correctly.
- Power supply can be shut off.
- Teach operation mode is functioning properly.
- Safety device and interlock switch function correctly.
- Other safeguards (if applicable) are installed correctly in their prescribed locations.
- Robot operates accurately in restricted status (low power status).
- Robot operates properly under rated loads and at maximum speed.

### **Restarting after a Change**

When restarting the robot system after its hardware or software has been corrected or serviced, strictly observe the following:

- Before supplying the system with power, check the locations where the hardware was modified.
- Test the functions of the robot system to make sure that it operates correctly.



## 2.5 Precautions regarding Robot Operation

### 2.5.1 General Precautions

Before operation, become familiar with the location of all emergency stop switches.

During an emergency, always press the nearest emergency stop switch. There should never be any emergency stop switches in the system that do not operate.

After an emergency, do not restore the emergency stop circuit until it has been determined that the entire system is safe to restart.

If your robot is a 6-axis type, record the pulse values of the reference points used for the calibration. For details, refer to the *6-axis Robot Manipulator manual, Setup & Operation: 3.6 Setting the Reference Points for Calibration*.

### 2.5.2 Automatic Operation

Ensure that system automatic operation is enabled only while the following requirements are being met:

- Emergency stop switches are installed in the prescribed location and operate correctly.
- No personnel are inside the safeguarded area of the system.
- Safety procedures that are established separately for the robot system (if applicable) are being followed.

### 2.5.3 Teaching Robot Points

If possible, teaching should be performed with no personnel inside the safeguarded area.

Teach mode can be used to allow the robot to be jogged or moved at slow speed when the safeguard is open. Before going inside the safeguarded area, robot operators that need to move the robot under servo control must switch Teach mode to ON by using the mode selector key switch of the teach pendant. Operators then carry the teach pendant while inside the safeguarded area. As a result, the operation mode cannot be changed from outside the safeguarded system while somebody is inside the safeguarded area.

#### **Auto Mode and Program Mode**

With the safeguard circuit open, the robot motors will be turned off and the robot cannot be jogged under power. However, the robot can be moved by hand to a position with the safeguard circuit open and the position can then be taught.

#### **Teach Mode**

The robot can be jogged or moved at slow speed as long as the three position enable (dead man) switch is engaged.

Please observe the following guidelines for teaching points:

- Robot operators must receive training that utilizes the same type of robot. Before teaching, the operator should be thoroughly familiar with teaching procedures.
- Before teaching, remove all errors and malfunctions.
- Before the robot operator goes inside the safeguard system, confirm that the robot motors go off when the safeguard is open and that emergency stop switches are functioning correctly.
- The robot operator should visually check the robot system and safeguard system interior to ensure that there are no potential hazards.
- Design the system in such a way that prevents the overall robot system from being started up from any location while the operator is inside the safeguarded area.

- If there is a possibility of a dangerous situation arising from the operation of a device other than the robot, such as an actuator, take steps to prevent such operation or ensure that these devices can only be controlled by the teaching operator.

### 2.5.4 Return to Automatic Operation

If there are safety devices that have been temporarily disabled for system inspection or other reasons, always return them to their original status before restarting automatic operation.

### 2.5.5 Program Verification

If it is necessary to verify a program while the system is in unrestricted (high power) status, first make sure that no personnel are inside the safeguarded area.

### 2.5.6 Troubleshooting

Troubleshoot from outside the safeguard system. If that is not possible, strictly observe the requirements below.

- Operators responsible for troubleshooting should be trained and qualified to perform such work.
- Establish work safety procedures to minimize the danger that operators inside the safeguard system will be exposed to.

### 2.5.7 Maintenance

In order to keep the robot and robot system operating safely, maintenance (and inspection) is important. Adequately trained personnel should perform the procedures required to do the maintenance work safely. Make sure that maintenance is performed according to the instructions in the robot and controller manuals (maintenance editions).

If maintenance is required inside the safeguarded area, take the following precautions:

- Shut off the power supply using lockout / tagout to prevent anyone from turning ON the robot power supply by mistake. For further details, refer to the section *Procedure of Lockout/Tagout* in the chapter *Safety Precautions* in the controller manual.
- If the robot system power supply cannot be shut off, strictly observe the following:
  - (1) Visually inspect the robot system to ensure that there are no conditions that could lead to a malfunction.
  - (2) If it is discovered that the robot system is damaged or malfunctioning, perform the required repairs and retest it before allowing the operator to go inside the safeguard system.
- Grant full control of the robot and robot system to those performing maintenance and/or repairs inside the safeguard system.
- Ensure that the robot system does not respond to any remote control devices.
- Ensure that all emergency stop devices are functioning correctly.
- Before starting the robot system in automatic operation, return all temporarily disabled safety devices to their original enabled status.
- Do not use tweezers or other metal tools to aid in battery replacement. This could cause a battery short. Replace a battery using only the specified type and be careful to observe the polarity of the battery.

### 2.5.8 Backup of Projects and Controller

After a project has been created or edited, or after system data including robot parameters has been edited, the project and controller files should be copied and stored in media other than the hard disk on the PC (e.g. USB memory key). Keep the backup media in a safe place in case of damaged data on the hard disk.

To backup, select Controller from the EPSON RC+ 6.0 Tools Menu and execute Backup Controller. Refer to the section *5.11.8 EPSON RC+ 6.0 GUI - Controller Command (Tools Menu)*.

Backup Controller is a function to backup both the project and the controller.

To backup only the project data, select Copy from the Project Menu. Refer to the section *5.9.10 EPSON RC+ 6.0 GUI - Copy Command (Project Menu)*.



CAUTION

- If your system cannot be restored by Restore Controller, you must restore robot calibration parameters (Hofs, CalPIs) before operating the robot. If you fail to do so, the robot will move to incorrect positions.

## 2.6 End User Instruction Manual

Be sure that the robot system instruction manual supplies a list of all the equipment included in the system (such as the robots, associated equipment, and safety devices) as well as a description of how to use each.

Be sure to provide the following in the manual:

- An easy to follow explanation of the robot system and how to install it, as well as a step by step summary of the system installation and external power supply connections.
- A description of all hazards and how to avoid them.
- A description (including interconnection diagrams) of the safety devices, interacting functions, and safeguard's interlocking function against hazardous conditions, in particular, the safeguard's interlocking function for devices installed to perform interactively.
- Precise instructions regarding usage of the system.

## 2.7 End User Training

Be sure those in charge of safety management confirm that the operators who program, operate, and maintain the robot and robot system obtain proper training and have the expertise to conduct the work safely.

Training should include at least the following:

- Study of regulation safety procedures, and safety-related recommendations by robot manufacturers and system designers.
- Clear explanation of the work involved.
- Description of all control equipment required for the work and their functions.
- Explanation of potential hazards involved in the work.
- Work safety procedures and specific methods of avoiding potential hazards.
- Safety device and interlock function testing and confirmation methods are working properly.

### 3. Getting Started

This chapter contains instructions for setting up and using EPSON RC+ 6.0. It is recommended that first time users first read the preceding Safety chapter, then read through this chapter to get more familiar with the system.

#### Contents

- Hardware Installation
- Software Installation
- Installation for Offline Development
- PC Configuration
- Windows Security Administration

### 3.1 Hardware Installation

EPSON RC+ 6.0 is used with the RC620 Controller. You need to install the controller and robot before you can use EPSON RC+ 6.0 to develop and run SPEL+ applications.

The Controller comes pre-configured at the factory. For instructions on installation, refer to the RC620 Controller manual.

### 3.2 Software Installation



EPSON RC+ 6.0 is used with the RC620 Controller.

For RC620 Controller, the EPSON RC+ 6.0 has been already installed before shipment.

To update the software version, refer to *Appendix B: EPSON RC+ 6.0 Software*.

EPSON RC+6.0 can be installed in the other PC such as laptop computer as well as the controller and edit programs and build projects.

For the details, refer to the next section *3.3 Installation for Offline Development*.

### 3.3 Installation for Offline Development



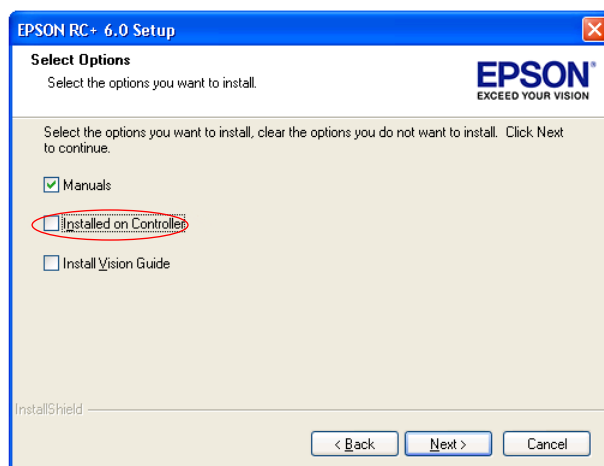
Offline development using EPSON RC+ 6.0 is supported on the following operating systems:

- Windows XP Professional Service Pack 3 or later
- Windows Vista Business or Windows Vista Business Service Pack 2 or later
- Windows 7 Professional

Insert the EPSON RC+ 6.0 setup DVD into the DVD drive of a PC (not the controller) such as laptop computer, and start the installation.

Follow the instructions in the setup wizard to install the software.

For offline development, make sure you uncheck the [Installed on Controller] preference in the following dialog before the installation.  
(The [Installed on Controller] preference is checked by default.)



When using EPSON RC+ 6.0 for offline development, you can edit programs and build projects. You cannot use Vision Guide 6.0 in offline development.

### 3.4 Windows Security Administration

Users need Administrator rights to use the EPSON RC+. Other users such as Power User, Limited User, Guest User cannot use EPSON RC+.

User Name and Password at shipment

The RC620 robot controller is configured at factory as shown below:

User name : EPSON RC User (with Administrator authority)

Password : epson

To provide security within the EPSON RC+ environment, a Security software option is available. This option allows you to manage EPSON RC+ users and audit development activity. Refer to *14. Security* for details.

## 4. Operation

This chapter contains instructions for operation of the EPSON RC+ 6.0 system. The main topics are:

- System Power Up Procedure
- Starting EPSON RC+ 6.0
- System Shutdown Procedure
- Writing your first program

### 4.1 System Power Up Procedure

Follow this procedure to power up the system:

1. Ensure that all safeguards are in place and that all personnel are clear of the equipment.
2. Apply power to the Controller, monitor, and I/O devices.
3. Start the EPSON RC+ 6.0 software.

### 4.2 Starting EPSON RC+ 6.0

There are three ways to start EPSON RC+ 6.0. You can also configure the mode that EPSON RC+ 6.0 starts in.

#### Start Method 1

1. Double click on the EPSON RC+ 6.0 robot icon located on the Windows desktop.

#### Start Method 2

1. Click the Windows Start button.
2. Select the EPSON RC+ 6.0 Program Group.
3. Select EPSON RC+ 6.0.

#### Start Method 3

Configure EPSON RC+ 6.0 to start automatically after Windows starts. The details are described later in *4.2.7 Auto Start*.



When using the VB Guide option, you do not need to start EPSON RC+ 6.0. The library provided with VB Guide will load EPSON RC+ 6.0 into your .NET application process automatically.

#### 4.2.1 Startup Sequence

When EPSON RC+ 6.0 starts, it reads initial settings for the current user and local system from the Windows registry.

The remainder of the startup sequence depends on the following:

- EPSON RC+ 6.0 session number
- SPEL Controller Control Device setting
- SPEL Controller Independent Mode setting

#### For session 1 with independent mode off and any control device

If there are no project files specified on the startup command line, then the last project that was opened will be opened at startup time. The program files that were last opened will be opened again.

If the Start Mode is Auto, the Start Mode dialog is opened (see *4.2.4 Start Mode Dialog*).

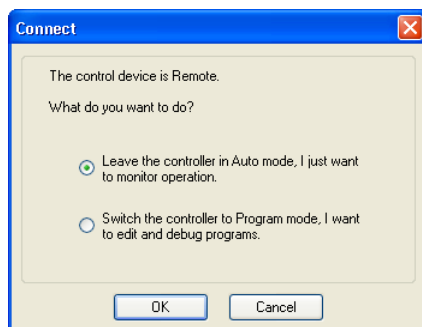
If the Start Mode is Program, the EPSON RC+ 6.0 GUI is opened.

**For session 1 with independent mode on and control device is remote**

If there are no project files specified on the startup command line, then the last project that was opened will be opened as read only at startup time. The program files that were last opened will be opened again.

If tasks are currently running, EPSON RC+ 6.0 will prompt to enter Monitor Mode.

If no tasks are currently running, a dialog below will be displayed.

**For sessions > 1**

EPSON RC+ should be started in Off-line mode. If there are no project files specified on the startup command line, then the last project that was opened will be opened as read only at startup time. A connection toolbar dropdown list is displayed allowing you to enter Monitor Mode to monitor session 1 operation.

For more information on Monitor Mode, refer to 4.2.8 *Using Monitor Mode*.

**Cooperative mode and Independent mode**

The RC620 robot controller has two CPUs.

Real Part : Controls the SPEL<sup>+</sup> program (Specialized for the real time control)

Windows Part : Controls the Windows applications (GUI)

The main function of the robot can be run by Real Part and some functions of the controller uses the Windows Part (See below).

Function	RC+ Enabled	PC Enabled
Detail of available function	Vision Guide (Frame Grabber) VB Guide Fieldbus master	PC file PC RS-232C Database access DLL calling

Real Part and Windows Part are started up separately at the each timing.

To operate the robot system without problem, you should synchronize these two parts. At the shipment of RC620 robot controller, the **Cooperative mode** that synchronizes these parts is applied.



NOTE

According to the design of robot system, it may not need to synchronize Real Part and Windows Part. In this case, change to **Independent mode**.

For the instructions of this settings, see the section below *How to set the Independent mode*.

When the controller is in Cooperative mode, it has to wait until both of Real Part and Windows Part can start up without failure.

## 4. Operation

Meanwhile, the LCD on the controller face displays as below:

----  
Waiting for RC+

Then it also has to wait until Windows part is ready and RC+ can start up without failure.

This table shows the startup sequence when the controller is in Cooperative mode:

	LCD display	Console instruction	Background task
(1) Power ON	<div style="border: 1px solid black; padding: 5px;">EPSON Robot Controller</div>	Not available	Not started yet
(2) Real Part starts up	<div style="border: 1px solid black; padding: 5px;">---- Waiting for RC+</div>	Not available	Not started yet
(3) Windows part starts up	<div style="border: 1px solid black; padding: 5px;">---- Waiting for RC+</div>	Not available	Not started yet
(4) RC+ starts up	<div style="border: 1px solid black; padding: 5px;">----- Ready      V.6.0.0.1</div>	Available	Already started

(Includes the startup of the Operator Window and VBGuide application)

This table shows the startup sequence when the controller is in Independent mode:

	LCD display	Console instruction	Background task
(1) Power ON	<div style="border: 1px solid black; padding: 5px;">EPSON Robot Controller</div>	Not available	Not started yet
(2) Real Part starts up	<div style="border: 1px solid black; padding: 5px;">----- Ready      V.6.0.0.1</div>	Available    *1	Already started
(3) Windows part starts up	<div style="border: 1px solid black; padding: 5px;">----- Ready      V.6.0.0.1</div>	Available    *1	Running
(4) RC+ starts up	<div style="border: 1px solid black; padding: 5px;">----- Ready      V.6.0.0.1</div>	Available	Running

\*1 When the console is “SELF” :

It waits the command execution from the Operator Window or VBGuide application.

When the console is “REMOTE” :

(2) As Real Part starts up, Remote function becomes enable and starts operating.

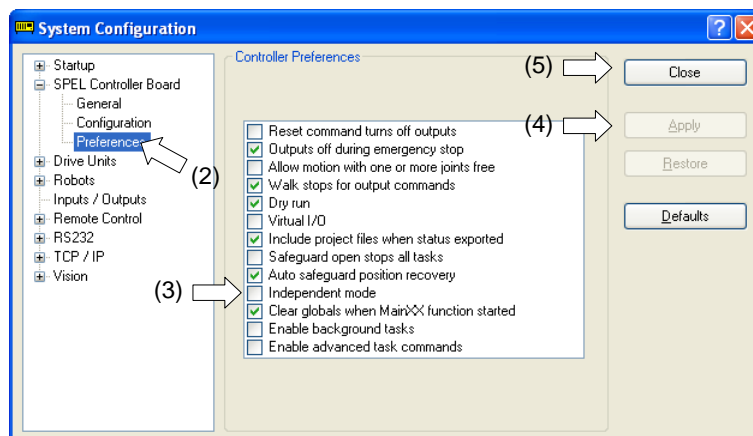


When the controller is in Cooperative mode, the state does not back to wait for the RC+ connection even after RC+ shutdown. Also when the console is Remote, you need to be careful during the RC+ shutdown because the remote command is still executable.



### How to set the Independent mode

- (1) Select the Setup | System Configuration from the main menu and displays the [System Configuration] dialog as shown below.
- (2) Select the SPEL Controller Board | Preferences.



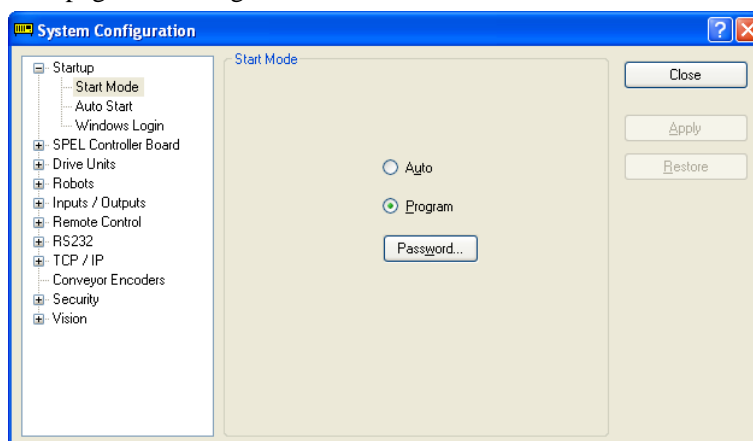
- (3) Set the [Independent mode] checkbox.
- (4) Click the **Apply** button.
- (5) Click the **Close** button.

#### 4.2.2 Startup Configuration

To configure startup, select the [System Configuration] from the Setup Menu. The Startup section has pages for Start Mode, Auto Start, and Windows Login.

#### 4.2.3 Start Mode

This page has settings for the EPSON RC+ 6.0 start mode.



There are two start modes:

**Program** This mode allows you to develop your projects. This is the default startup mode.

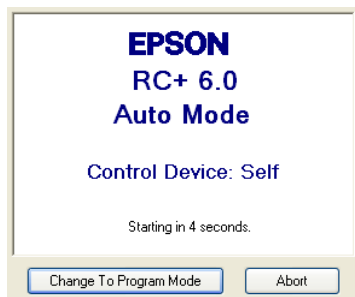
**Auto** This mode starts the system and displays the Operator Window.

Use the **Password** button to change the start mode password.

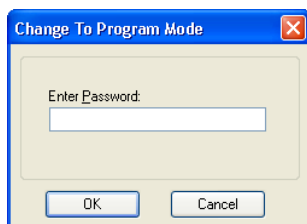
### 4.2.4 Start Mode Dialog

When the start mode is set for Auto, then a dialog is displayed at start up that allows you to change the startup mode using a password. After several seconds, if the **Change To Program Mode** button has not been clicked, the system will initialize and the Operator Window will be displayed.

You can disable this startup dialog using command line options described later in this section, *4.2.10 Command Line Options*.



If you click the **Change To Program Mode** button, another dialog will be displayed, as shown below:



To change to Program mode, you must supply the password and click **OK**, or you can abort startup all together by clicking **Cancel**.



NOTE

This allows authorized personnel to enter Program mode temporarily to make changes or adjustments.

When you change to PROGRAM mode from this dialog, it is only temporary. The next time EPSON RC+ 6.0 runs, the original start mode setting will be used.

### 4.2.5 Start Mode: Program

Program mode is the default start mode. This is the EPSON RC+ 6.0 development environment, from which you can:

- Create / edit projects.
- Configure the controller and set preferences.
- Run and debug programs.

### 4.2.6 Start Mode: Auto

Auto mode displays the Operator Window. The Operator Window is configured according to the settings in Project | Properties.

The current controller Control Device sets the mode of Auto operation.

Control Device	Description
Self	The Operator Window can be used as a simple operator interface for production.
Remote I/O	The Operator window is displayed with no operator buttons to allow any diagnostic messages to be viewed.

### 4.2.7 Auto Start

You can configure EPSON RC+ 6.0 to automatically start when Windows starts.

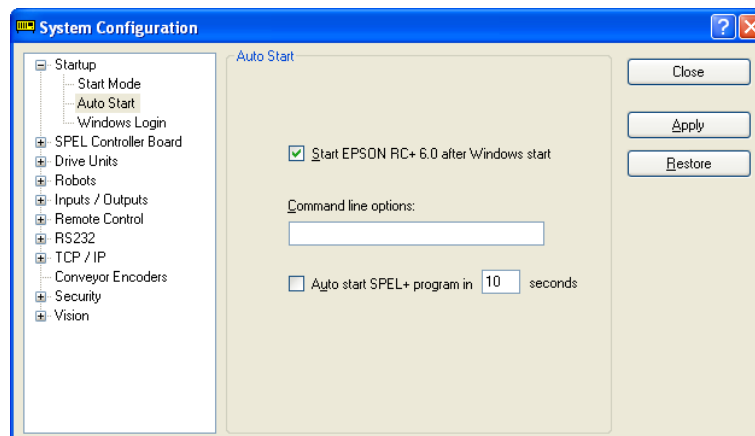
From the Setup | System Configuration | Auto Start page, set the **Start EPSON RC+ 6.0 after Windows start** check box.

In addition, if you set the checkbox above, you can specify EPSON RC+ 6.0 command line options (/auto, /nosplash, etc.) in the Command line options text box. Refer to the section 4.2.10 *Command Line Options*.

You can also specify if the main function should automatically be started after a delay. During the delay, the operator can click Stop to abort the automatic startup.



When using auto start, ensure that your application can automatically start safely and inform operators how to abort the startup.



### 4.2.8 Using Monitor Mode

Monitor Mode allows you to monitor operation of the controller. In Monitor Mode, you can do the following:

- View print output on the Run window
- Monitor I/O status using the I/O Monitor.
- Monitor task status using the Task Manager.
- Monitor variable values using Display Variables.

To enter monitor mode:

#### When independent mode is off

1. Start a second session of EPSON RC+ 6.0.
2. On the main tool bar, select Monitor from the connection dropdown list.

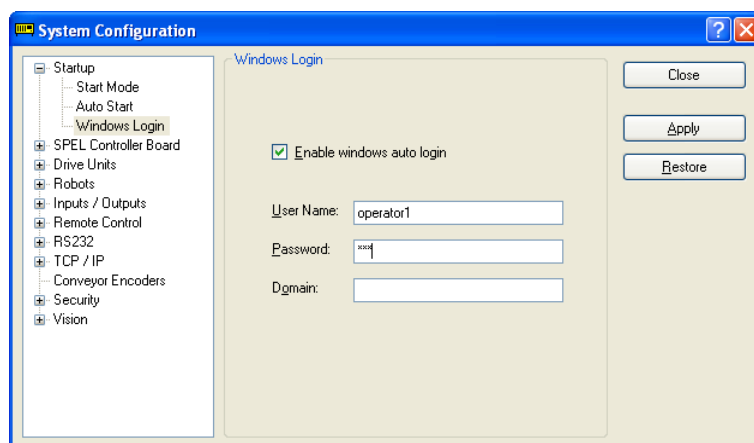
#### When control device is remote and independent mode is on

1. Start EPSON RC+ 6.0 first session.
2. If tasks are running, you will be prompted to connect and monitor operation. If tasks are not running, you will be prompted to connect in monitor mode, or switch to program mode.

### 4.2.9 Windows Login

You can configure automatic Windows login from EPSON RC+ 6.0. From the Setup | System Configuration | Startup | Windows Login page, set the **Enable windows auto login** check box. Then, enter the name and password of the user logging in. Optionally, you can supply a domain, if required.

However, you must have the authority of Windows Administrator to set login parameters. To configure automatic Windows login from EPSON RC+ 6.0, you must reboot the system the first time. After the reboot, Windows login will be automatic.



### 4.2.10 Command Line Options

Refer to 4.2.11 *Using Command Line Options* to see how to use the command line options.

There are command line options for EPSON RC+ 6.0 that provide the following functions:

#### Starting EPSON RC+ 6.0 for a specific project

When you start EPSON RC+ 6.0, you can optionally specify a project name in the command line.

```
ERC60.EXE [drive:project_name]
```

*drive:project\_name* The drive letter and name of a project. The name can include a subfolder of the \EpsonRC50\Projects directory.

#### Example

Open project *myapp* on drive C: at startup:

```
ERC60.EXE c:myapp
```

#### Change the EPSON RC+ 6.0 startup mode

You can select the startup mode and override the startup dialog using command line options.

#### To start in Program mode (no password required)

```
ERC60.EXE /PROG
```

#### To start in Auto mode

```
ERC60.EXE /AUTO
```

Use these command line options to override and hide the startup dialog and open the Operator Window directly.

If only the AUTO flag is supplied and the control device is PC, EPSON RC+ 6.0 will open the project from the last session and display the operator window. EPSON RC+ 6.0 will only be visible in the Windows Task Manager. When the operator window is closed, EPSON RC+ 6.0 will be terminated.



When the control device is PC, you cannot close the operator window while tasks are running.

#### Example

Open project *myapp* on drive C and display the operator window:

```
ERC60.EXE c:myapp /AUTO
```



The Controller should be ON before starting EPSON RC+ 6.0 with the /AUTO command line option. If EPSON RC+ 6.0 cannot communicate with the controller, then an error message will be displayed with a retry button.

For more details, see 7.6 *Operator Window*.

#### Login

You can automatically login from the command line if you are not using the Auto Login feature for the security Option:

```
ERC60.EXE /LOGIN "userID", "password"
```

This is especially useful when you are starting in operator mode.

If the user I/D or password is invalid, it will display an error dialog and exit the EPSON RC+ 6.0.

### Starting EPSON RC+ 6.0 specifying the language

You can specify the language to use in EPSON RC+ 6.0 GUI.

English	:	ERC60.EXE	/LANG_ENGLISH	
Japanese	:	ERC60.EXE	/LANG_JAPANESE	<sup>*1</sup>
German	:	ERC60.EXE	/LANG_GERMAN	<sup>*2</sup>
French	:	ERC60.EXE	/LANG_FRENCH	<sup>*2</sup>

<sup>\*1</sup> Available for the controller of Japanese version

<sup>\*2</sup> Available for the controller of English version

### Disabling the EPSON RC+ 6.0 splash window

You can suppress the splash window displayed at startup using the following syntax:

ERC60.EXE /NOSPLASH

#### 4.2.11 Using Command Line Options

Examples of command line options are:

#### Running from the Windows Run Box

You can specify a command from the Windows Start menu | Run | Open text box.

e.g. C:\EpsonRC60\exe\erc60.exe C:myapp

#### Making startup icons for your projects

You can create icons that automatically start EPSON RC+ 6.0 for different projects and start modes.

1. Create a new shortcut to ERC60.EXE on your desktop by selecting the EPSON RC+ 6.0 icon, then right clicking and selecting Create Shortcut.
2. Right click on the icon and select Properties.  
In the command line text box, type ERC60.EXE followed by the name of the project file you want to open as described above. Add any other options to the command line, such as /AUTO, /PROG, etc.

## 4.3 Writing your first Program

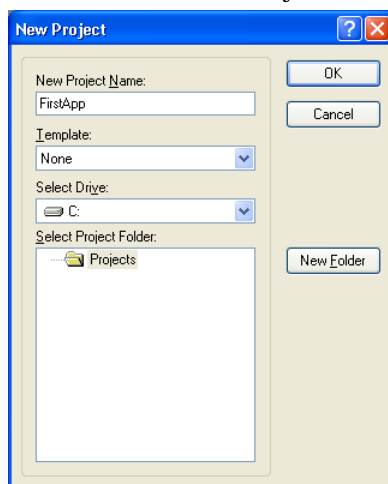
After installing the controller, robot, and EPSON RC+ 6.0 software on the RC620 Robot Controller, follow these instructions to create a simple application program so that you will become more familiar with the EPSON RC+ 6.0 development environment.

### 1. Start EPSON RC+ 6.0

Double-click the EPSON RC+ 6.0 icon on the desktop.

### 2. Create a new project

(1) Select New from the Project Menu.



(2) Type in a name for a project in the [New Project Name] box. e.g. FirstApp

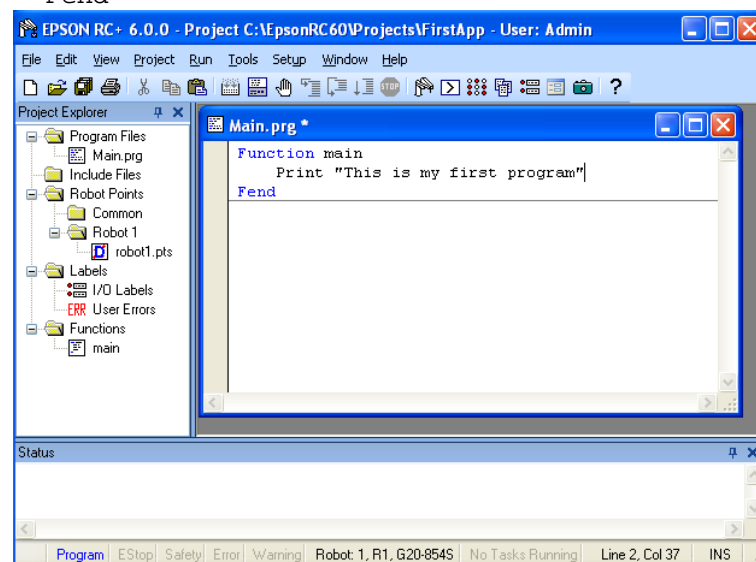
(3) Click **OK** to create the new project.

When the new project is created, a program called Main.prg is created. You will see a window open with the title Main.prg with a cursor flashing in the upper left corner. Now you are ready to start entering your first program.

### 3. Edit the program

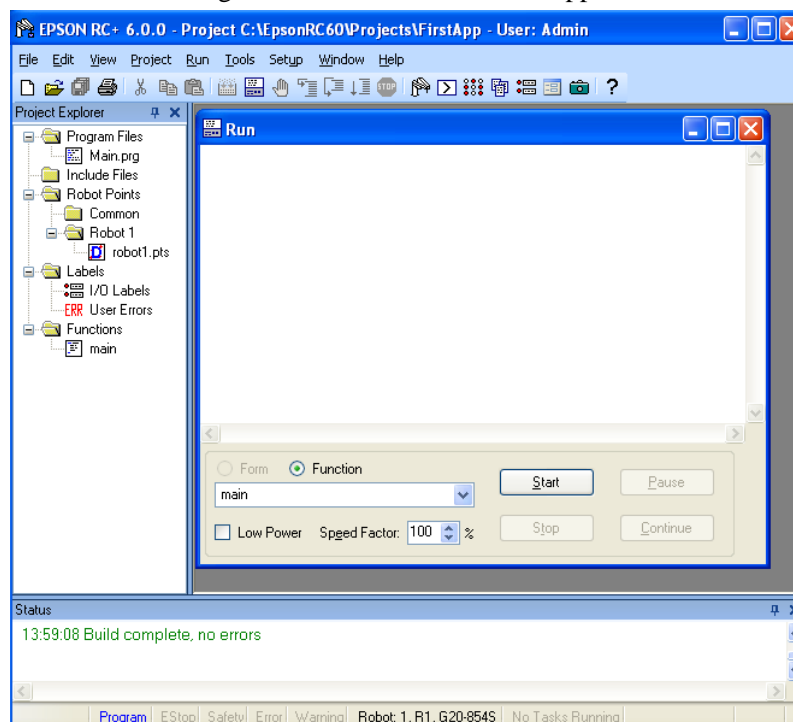
Type in the following program lines in the Main.prg edit window.

```
Function main
  Print "This is my first program."
Fend
```



### 4. Run the program

- (1) Press **F5** to run the program. (F5 is the hot key for the Run Window selection of the Run Menu). You will see the Status window located at the bottom of the main window showing the build operation status.
- (2) During project build, your program is compiled and linked. Then communications is established with the controller and project files are sent to the controller. If there are no errors during build, the Run window will appear.




- (3) Click the Start button on the Run window to run the program.
- (4) You should see text similar to the following displayed in the Status window:  
19:32:45 Task main started  
19:32:45 All tasks stopped

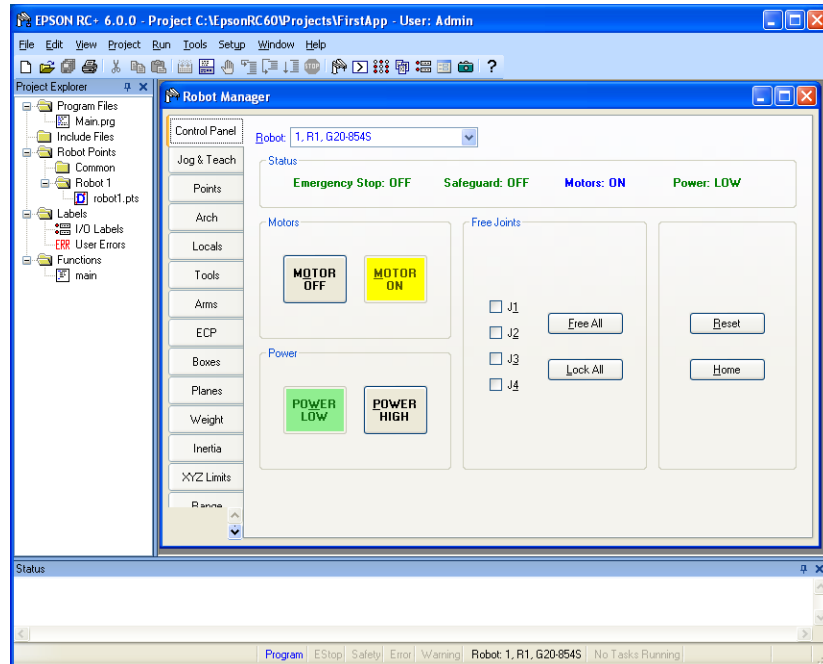
On the Run window, you will see the output of the print statement.



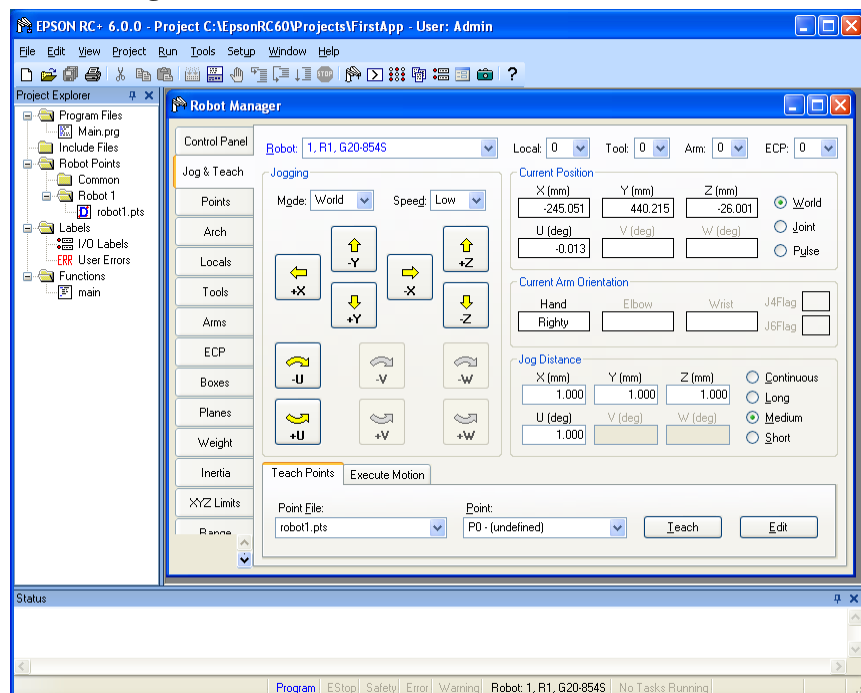
Now let's teach some robot points and modify the program to move the robot.

### 5. Teach robot points


- (1) Ensure that it is safe to operate the robot. Click the Robot Manager button  on the toolbar. You will see the Robot Manager window with the **Control Panel** page displayed.



- (2) Click on the **Motor On** button to turn on the robot motors. You will be prompted to confirm the operation.
- (3) Answer Yes to continue.
- (4) Click the **Jog & Teach** tab.



- (5) Click the **Teach** button in the lower right corner to teach point P0. You will be prompted for a point label and description.

- (6) Jog the robot by clicking the **+Y** jog button. Hold the button down to continue jogging. Let go when the robot is about half way out in the work envelope.
- (7) Jog the robot down by clicking the **-Z** button.
- (8) Now change the current point to P1 by selecting P1 in the Point dropdown list next to the Teach button.
- (9) Click the **Teach** button. You will see a confirmation message to teach the point.
- (10) Answer Yes.
- (11) Click the **+X** button to jog the robot in the +X direction.
- (12) Change the current point to P2 by selecting P2 in the Point dropdown list.
- (13) Click the **Teach** button. You will see a confirmation message to teach the point.
- (14) Answer Yes.
- (15) Click the Save Project  toolbar button to save the changes.

### 6. Modify the program to include robot motion commands

- (1) Insert three new Go statements into the Main.prg program as shown below:

```
Function main
    Print "This is my first program."
    Go P1
    Go P2
    Go P0
Fend
```

- (2) Run the program by pressing **F5** and then click on the **Start** button on the Run window.
- (3) The robot should move to each of the points you taught.

### 7. Modify the program to change speed of robot motion commands

- (1) Insert the Power, Speed, and Accel commands as shown in the program below:

```
Function main
    Print "This is my first program."
    Power High
    Speed 50
    Accel 50, 50
    Go P1
    Go P2
    Go P0
Fend
```

- (2) Run the program by pressing **F5**
- (3) Click on the **Start** button on the Run window.  
The robot should go to each of the points you taught at 50% speed, acceleration, and deceleration. The Power High statement enables your program to run the robot at high (normal) power, which in turn allows the robot speed and acceleration to be increased.

## 8. Backup the project and system configuration

Even though this is only a sample project, we will backup the project and controller configuration. This is easy to do with EPSON RC+ 6.0. It is important that you keep regular backups of your applications on an external media such as a USB memory key.

Follow these steps to backup the project and system configuration:

- (1) From the Project Menu, select Copy.
- (2) Change the Destination Drive to an arbitrary drive.
- (3) Click OK. The project will be copied to the external media.
- (4) From the Tools Menu, select Controller.
- (5) Click on the **Backup Controller** button.
- (6) Select the arbitrary drive.
- (7) Click OK. The system configuration will be backed up on the external media.

Now that you have written your first program, you should read *7.1.1 Creating the simplest application*.

## 4.4 System Shutdown Procedure

It is very important to shutdown the system properly. If the PC is turned off before shutdown, the controller UPS battery becomes overloaded and deteriorated, and also the files on the PC can be damaged.

There are three methods to shutdown the system:

- Shutdown from the Windows Start menu
- Shutdown from the **Shutdown** dialog with EPSON RC+6.0 shutdown
- Execute the Shutdown command from the SPEL<sup>+</sup> program
- Shutdown input from the Remote Console

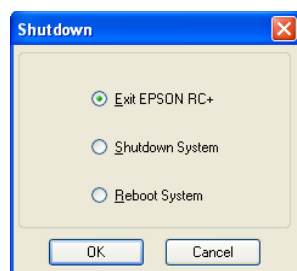
### Shutdown from Windows

Follow this procedure to shutdown the system:

1. Stop all tasks.  
You cannot shutdown EPSON RC+ 6.0 while tasks are running.
2. Shutdown EPSON RC+ 6.0 by executing File | Exit.
3. Shutdown Windows.

### Shutdown with EPSON RC+ 6.0 Shutdown

Shutdown the system with the dialog shown below that appears when you shutdown EPSON RC+ 6.0.



### **Shutdown from the SPEL<sup>+</sup> program**

You can shutdown the system using the Shutdown command.

For details refer to the *EPSON RC+ Online Help or SPEL<sup>+</sup> Language Reference manual*.

### **Shutdown input from the Remote Console**

With the Shutdown input from the Remote Console, the controller starts the shutdown processing.

For details refer to *11.7 Remote Inputs*.

## 5. The EPSON RC+ 6.0 GUI

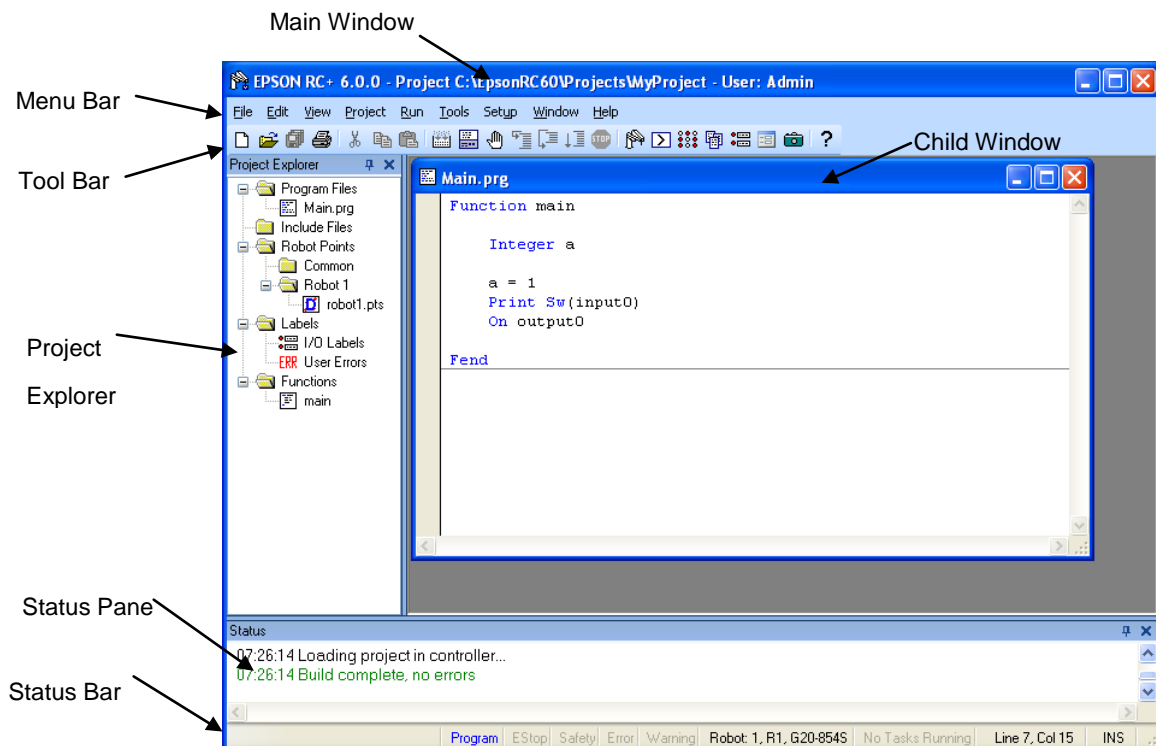
The chapter contains information on the EPSON RC+ 6.0 GUI. After an overview and discussing online help and toolbars, the remainder of the chapter is divided into sections that follow the menu system.

### Contents

- Overview
- Project Explorer Pane
- Status Window Pane
- Status Bar
- Online Help
- File Menu
- Edit Menu
- Display Menu
- Project
- Run
- Tools
- Setup
- Window
- Help

### 5.1 GUI Overview

EPSON RC+ 6.0 is a multiple document interface (MDI) application. There is one main parent window and several child windows which can be opened simultaneously. The main window has a menu bar, tool bar, and status bar, as shown below. In addition, there is a Project Explorer pane and Status Window pane.



## 5.2 Project Explorer Pane

The Project Explorer pane enables you to quickly open any file in the current project or jump to any function. The project files and functions are organized in a sorted tree structure.

Open a file or jump to a function : Double-click on the item.

Hide the Project Explorer : Click the X button on the bar above the pane.

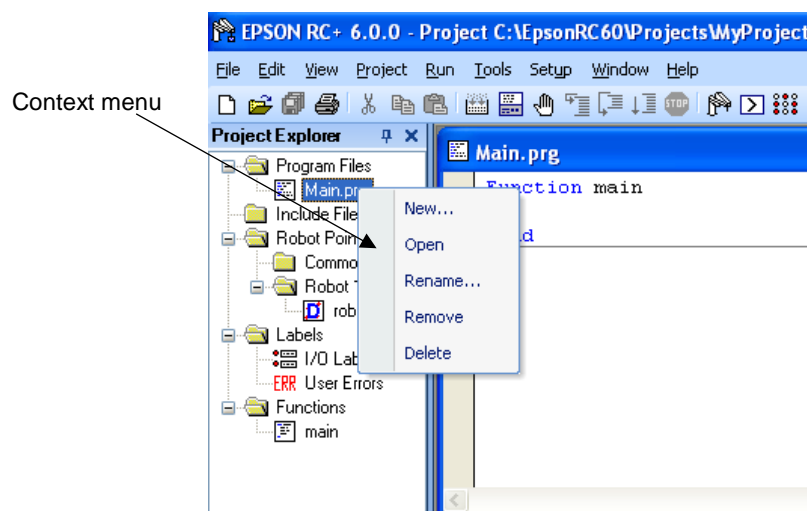
Show the Project Explorer : Select Project Explorer from the View Menu.

Resize the Project Explorer : Move the mouse cursor over the right side of the pane, then drag the pane right or left to the desired width.

You can move the Project Explorer pane to either the left or right side of the main window. To move the pane, click down on the bar above the pane, then drag to either the left or right side of the main window and release the mouse button.

### 5.2.1 Context Menu

The Project Explorer Pane has a context menu for various operations for elements in the project tree. To access the context menu, right click on an item in the project tree.



## 5.3 Status Window Pane

The status pane displays status messages, such as project build status, system errors and warnings, etc.

Hide the Status pane : Click the X button on the bar above the pane.

Show the Status pane : Select Status Window from the View Menu.

To resize the Status pane, move the mouse cursor over the top edge of the pane, then drag the top edge up or down.

The Status pane is always located at the bottom of the main window and cannot be moved.



NOTE

If the Status pane is closed and an error message is displayed on the status pane, such as during project build, the Status pane will automatically be opened so that the error message can be seen.

## 5.4 Status Bar

The status bar located at the bottom of the main window displays the following:

<b>Message area</b>	Displays the syntax error for the current line and system messages.
<b>Operation Mode status</b>	Indicates the controller operation mode.
<b>Emergency Stop status</b>	Indicates if emergency stop is active.
<b>Safeguard status</b>	Indicates if one or more safeguard circuits is open.
<b>Error status</b>	Indicates if the controller is in the error state. Put the mouse cursor over the Error status area to see the warning message.
<b>Warning status</b>	Indicates if there is a warning. Put the mouse cursor over the Warning status area to see the warning message.
<b>Current robot</b>	Displays the currently selected robot number, name, type number, and the dry run status.
<b>Tasks running status</b>	Indicates that one or more tasks are running.
<b>Current Line and Column</b>	When a program editor window is active, the current line and column are displayed.
<b>INS / OVR status</b>	Indicates insert or overtype mode.

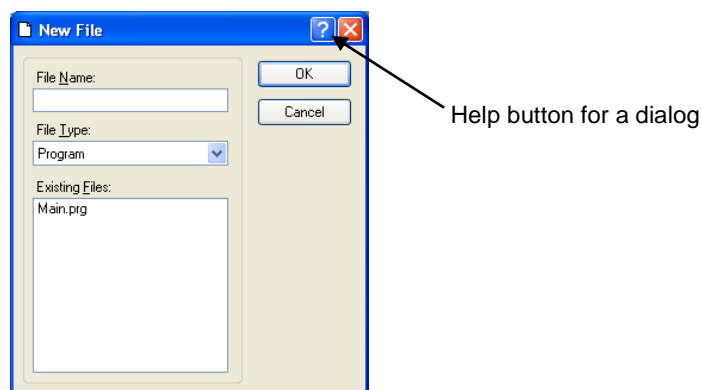
## 5.5 Online Help

EPSON RC+ 6.0 has an extensive context sensitive help system.

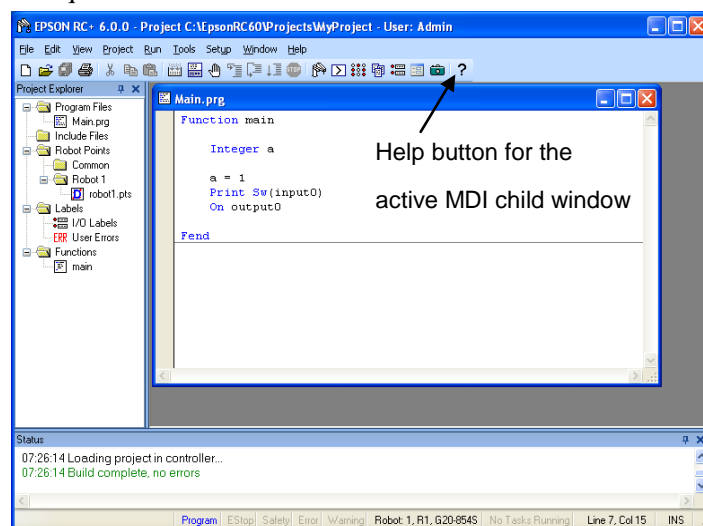
There are several methods to get help.

- Select **C**ontents from the Help Menu to browse help topics.
- Select **I**ndex from the Help Menu to enter the name of a specific topic.
- Select **S**earch from the Help Menu to search for a specific topic.
- When editing programs, press F1 with the caret in the keyword of interest.

When a dialog is open, press F1 or click the Help button. For dialogs, the Help button is located in the window title bar on the right side and is shown as a question mark icon as shown below.



For MDI child windows, the Help button is located on the main toolbar and is also shown as a question mark icon as shown below.




## 5.6 File Menu

The EPSON RC+ 6.0 File Menu includes commands for managing and printing files in the current project.

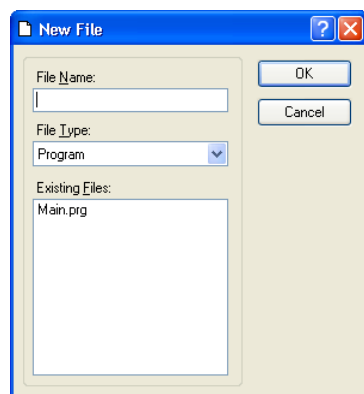
### 5.6.1 New Command (File Menu)

#### Shortcuts

Toolbar: 

Keys: Ctrl + N

The New command is used to add new files to the current project. When the New command is selected, the New File dialog is opened.




Item	Description
<b>File Name</b>	Enter a name for the new file in this box. If you supply a valid file extension, the File Type selection will change to match the extension. For a file name, two byte characters such as Japanese, Chinese characters are not allowed.
<b>File Type</b>	Use this dropdown list to select Program, Include or Point file.
<b>Existing Files</b>	Shows the files for the selected type currently in the Project folder.
<b>OK</b>	Click OK when you are ready to create the new file.
<b>Cancel</b>	Cancels the operation.



## 5.6.2 Open Command (File Menu)

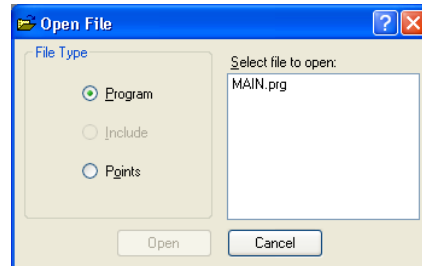
### Shortcuts

Toolbar: 

Keys: Ctrl + O

Open one or more files in the current project for editing. You can open program files, include files, or point files.

If there is a file in the current project folder (as shown in the Edit Project dialog box) and the file is not in the current project, you will not be able to open the file. You must add the file to the project before you can open it. This also applies to include files and point files.



Item	Description
<b>Program</b>	Select this radio button to show a list of program files in the current project.
<b>Include</b>	Select this radio button to show a list of include files in the current project.
<b>Points</b>	Select this radio button to show a list of point files in the current project.
<b>Select file to open</b>	Click on the file name you want to open. You can select more than one file by using the Ctrl key or Shift key. The Ctrl key allows you to select or deselect any file. The Shift key allows you to select a group of files.
<b>Open</b>	Opens the selected file(s).
<b>Cancel</b>	Cancels the open operation.



TIP

You can also double click on a file name in the **Select file to open** list box to open the file without having to choose the **OK** button.

## 5.6.3 Close Command (File Menu)

### Shortcuts

Keys: Ctrl + D

Close the currently active window.

Any of the windows can be closed with this command: Programs, Include files, Point files, Command Window, Run window, I/O Label Editor, user errors.



TIP

You can also close a window or dialog box by double clicking on the control box button located in the upper left corner of the window or dialog box.

### 5.6.4 Save Command (File Menu)

#### Shortcuts

Keys:           Ctrl + S

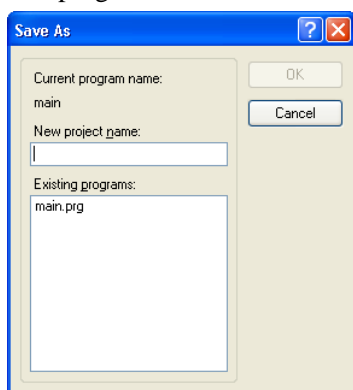
The Save command writes the current file to disk. The current file can be a program file, include file, points file, I/O label editor, etc. This command is disabled if the current file does not need to be saved.

### 5.6.5 Save As Command (File Menu)

Save the program, include file, or point file in the currently active window with a new file name. The original file will be removed from the project but will remain on the disk. The new name will be used throughout the current project in place of the old name.

If you use **Save As** on an include file, you must rename the file in each of your #include statements that refer to it. For a file name, two byte characters such as Japanese, Chinese characters are not allowed.

The program file name must be less than 24 characters (excluding the extension).



### 5.6.6 Restore Command (File Menu)

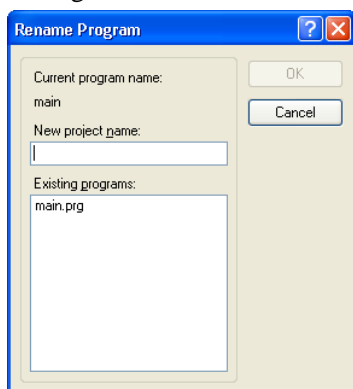
Restores the currently active program, include file, I/O labels, user errors, or point files from disk.

Use this function to change a document to the state it was in since last saved.

You will be prompted to confirm this operation.

### 5.6.7 Rename Command (File Menu)

Use Rename to change the name of the program, include file, or point file you are currently editing.



**To rename a file**

- Click anywhere on the program window
- Select the Open command from the File Menu
- Select the Window from the Window Menu
- Select from the Window Menu list

Select Rename from the File Menu. Type in a new name for the file and click **OK**.

The new file name cannot be the same as the existing files. You will get an error message if you enter a new name that is already being used.

If you use **Rename** on an include file, you must rename the file in each of your #include statements that refer to it.

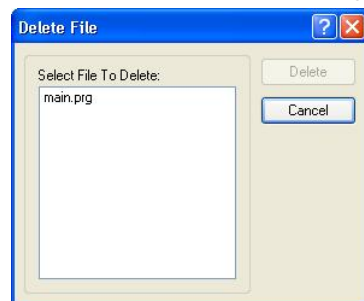
For a file name, two byte characters such as Japanese, Chinese characters are not allowed.

The program file name must be less than 24 characters (excluding the extension).

### 5.6.8 Delete Command (File Menu)

This command allows you to delete a file in the current project folder. You can delete program files, include files, and point files.

The file does not have to be registered in the project to delete.

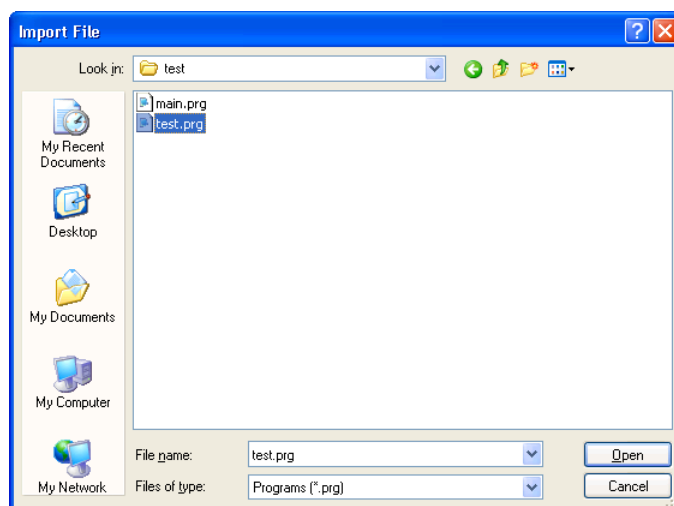


Item	Description
<b>Select file to delete</b>	Click on the file name you want to delete. This file list displays all .PRG, .INC, and .PTS files in the current project folder.
<b>Delete</b>	Deletes the selected file. You will be prompted with a confirmation message before the file is deleted. If the file is currently open, it will be closed and removed from the current project before it is deleted from disk.
<b>Cancel</b>	Cancels the delete operation.

### 5.6.9 Import Command (File Menu)

Import a file from other EPSON RC+ 6.0 projects. Use this command to import program files, include files, point files, I/O labels, user errors, and macros.

- Program file names for importing must have a .PRG extension.
- Include file names for importing must have a .INC extension.
- Point file names for importing must have a .PTS extension.
- I/O labels must have the file name IOLABEL.DAT
- User errors must have the file name USERERRORS.DAT.
- Macros must have the .MAC extension.



### To import a file

1. Select the file type from the **File Type** list box.
2. Navigate to the file you want to import.
3. Click **Open** to continue. If a file name is already used in the project folder, you will be prompted to confirm the overwriting. The file will then be copied to the current project's folder.



If you need to import files from previous versions of EPSON RC+ or from SPEL for Windows 2.0, you must first import the project using Project | Import, which converts the point files and label files into EPSON RC+ 6.0 formats. Then you can use File Import to import the desired files.

### 5.6.10 Print Command (File Menu)

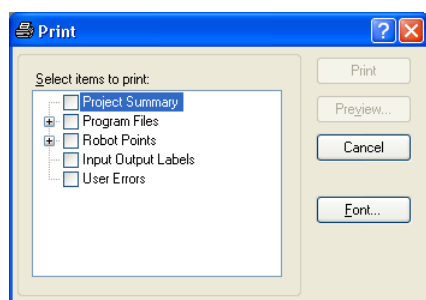
#### Shortcuts

Toolbar:

Keys: Ctrl + P

This command opens the Print dialog box. You can print programs, include files, point files, I/O labels, and user errors. Also you can print out a project summary.

Each document is printed with a header that includes the project name, product name, file name, date and time, and page number.



Item	Description
<b>Select items to print</b>	Check the items in the tree that you would like to print out.
<b>Project Summary</b>	Select this check box to print a summary of the programs and points used in the current project.

Item	Description
<b>Program Files</b>	Select this check box to print all program files, or click on the + button to view all program files and check the ones you want printed.
<b>Include Files</b>	Select this check box to print all include files, or click on the + button to view all include files and check the ones you want printed. This check box is not shown if there are no include files in the current project.
<b>Robot Points</b>	Select this check box to print all point files, or click on the + button to view all point files and check the ones you want printed.
<b>Input Output Labels</b>	Select this check box to print a listing of the all of the I/O labels used in the project.
<b>User Errors</b>	Prints a listing of all user errors for the current project. If either the label or message is non-blank, then the error definition will be printed.
<b>Print</b>	Prints the selected files. This button will be dimmed if nothing is selected to be printed.
<b>Preview</b>	Preview the selected files before printing. This button will be dimmed if nothing is selected to be printed.
<b>Font...</b>	Opens a dialog for selecting the printer font. The selected font is saved for subsequent printing.
<b>Cancel</b>	Closes the dialog box without printing anything.

#### 5.6.11 Exit Command (File Menu)

##### Shortcuts

Keys: Alt + F4

Exits from EPSON RC+ 6.0.

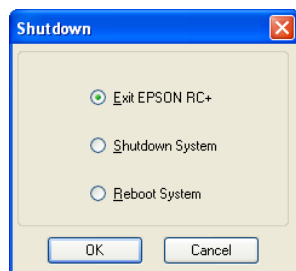
If you are running a program from the Run Window and the control device is PC, you will see a message that a program is running and you will not be allowed to exit. You must stop all tasks first before you can exit.

If there are any open program files, include files, point files, I/O labels, or user point files that have not been saved, for each file you will be prompted to save it with Yes, No, or Cancel.

If you select **Yes**, then the file will be saved and the shutdown dialog will be displayed.

If you select **No**, then the program will exit without saving the files and display the shutdown dialog.

If you select **Cancel**, the shutdown will not be displayed.



Item	Description
<b>Exit EPSON RC+</b>	Exits the EPSON RC+ 6.0.
<b>Shutdown System</b>	Exits the EPSON RC+ 6.0 and the system.
<b>Reboot System</b>	Exits the EPSON RC+ 6.0 and reboot the system.
<b>OK</b>	Executes the selected operation.
<b>Cancel</b>	Cancels the operation and close the dialog.

## 5.7 Edit Menu



TIP

The EPSON RC+ 6.0 Edit Menu includes commands for editing files.

You can also access the Edit Menu by right-clicking anywhere in a program editor window.

### 5.7.1 Undo Command (Edit Menu)

#### Shortcuts

Keys: Ctrl + Z

Undo the changes to the currently active program since it was open.

### 5.7.2 Redo Command (Edit Menu)


#### Shortcuts

Keys: Ctrl+Y

Redo the previous undo.

### 5.7.3 Cut Command (Edit Menu)

#### Shortcuts


Toolbar: 

Keys: Ctrl + X

Copies the current selection into the Clipboard and then deletes the selection.

### 5.7.4 Copy Command (Edit Menu)

#### Shortcuts


Toolbar: 

Keys: Ctrl + C

Copies the current selection into the Clipboard.

### 5.7.5 Paste Command (Edit Menu)

#### Shortcuts

Toolbar: 

Keys: Ctrl + V

Puts the contents of the Clipboard into the currently active document starting at the insertion point.

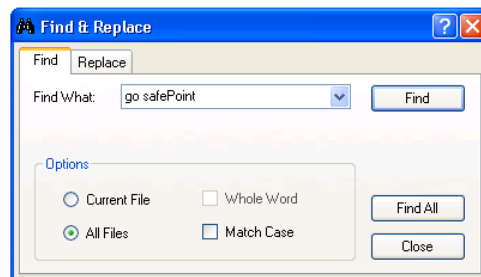
### 5.7.6 Find Command (Edit Menu)

#### Shortcuts

Keys: Ctrl + F

Find a text string in the current program or all programs in the project.

The first time you execute this function, the dialog box will be centered over the main window. If you reposition it, then the next time Find is executed, the dialog will appear where you last positioned it.



Item	Description
<b>Find What</b>	Type the text you want to search for. If any text was selected when you execute the Find command, it will be displayed here. When executing the Find with a text string selected, selected text will be displayed. If no text was selected, then the text from the last Find will be displayed. You are limited to one line of text. If selecting more than one line before executing Find, the search will not start.
<b>Current File</b>	Searches only in the current program file and include file.
<b>All Files</b>	Searches all files in the project.
<b>Whole Word</b>	Searches for the full word by itself and not as part of another word.
<b>Match Case</b>	Text must also match lower and upper case in order to be found.
<b>Find</b>	Starts the search. If the text is found in a file that is not open, then the file will be opened to display. This button will be dimmed if nothing is entered to be searched.
<b>Find All</b>	Search for all occurrences and list the results in the Status pane. Each result shows the file name, line number, and line where the text was found. You can then double click on a result to open the file where the text was found. The Find & Replace dialog will close after the results are displayed. This button will be dimmed if nothing is entered to be searched.
<b>Close</b>	Closes the dialog box.

### 5.7.7 Find Next Command (Edit Menu)

#### Shortcuts

Key: F3

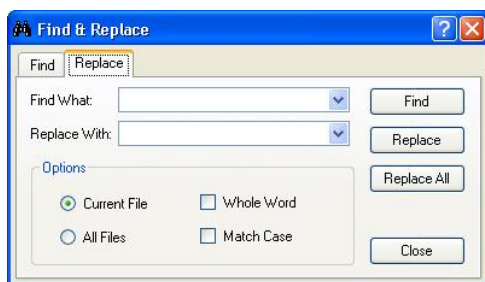
Find the next occurrence of the search text specified in the last Find command.

### 5.7.8 Replace Command (Edit Menu)

#### Shortcuts

Keys: Ctrl + R

Search for a text string and replace it with new text. The first time you execute this function, the dialog box will be centered over the main window. If you reposition it, then the next time Replace is executed, the dialog will appear where you last positioned it.



Item	Description
<b>Find What</b>	Type the text you want to search for. If any text was selected when you execute the Replace command, it will be displayed here. If no text was selected, then the text from the last Find will be displayed.
<b>Replace With</b>	Enter the replacement text here. The replacement text can be empty.
<b>Current File</b>	Searches only in the current program file and include file.
<b>All Files</b>	Searches all files in the project.
<b>Whole Word</b>	Searches for the full word by itself and not as part of another word.
<b>Match Case</b>	Text must also match lower and upper case in order to be found.
<b>Find</b>	Finds the next occurrence.
<b>Replace</b>	If already found, replaces the current find, otherwise searches for the next occurrence.
<b>Replace All</b>	Replaces all occurrences.
<b>Close</b>	Closes the dialog box.

### 5.7.9 Select All Command (Edit Menu)

#### Shortcuts

Keys: Ctrl + A

Selects the entire program file, include file, point file, I/O labels, or user errors. You can then execute Cut or Copy.



#### 5.7.10 Indent Command (Edit Menu)

##### Shortcuts

Key: Tab

Move the selected line one tab to the right.

#### 5.7.11 Outdent Command (Edit Menu)

##### Shortcuts

Keys: Shift + Tab

Move the selected line one tab to the left.

#### 5.7.12 Comment Block Command (Edit Menu)

Comments out the selected block of lines by adding the comment character to the beginning of each line.

To use, select one or more lines to be commented. Then :

- Select Comment Block from the Edit Menu.
- Right click and select Comment Block from the Context Menu.

A comment character will be added to the beginning of each of the selected lines.

#### 5.7.13 Uncomment Block Command (Edit Menu)

Removes leading comment character from the selected block of lines.

To use, select one or more lines to be uncommented. Then:

- Select Uncomment Block from the Edit Menu.
- Right click and select Uncomment Block from the Context Menu.

The first comment character from each of the selected lines will be removed.

#### 5.7.14 Go To Definition Command (Edit Menu)

Opens the window and sets the line where a function, variable, macro, point label, I/O label, or user error label is defined.

To use,

- Click on an identifier in a program window, and select Go To Definition from the Edit Menu.
- Right click on the identifier, and select Go To Definition from the Context Menu.

Identifier type	Display
<b>Function name or variable</b>	Program window where a function name or variable is declared.
<b>Pont label</b>	Point file which a label is defined.
<b>I/O label</b>	I/O label editor which a label is defined.
<b>User error label</b>	User error which a label is defined.

## 5.8 View Menu

The EPSON RC+ 6.0 View Menu includes commands for opening the Project Explorer and Status window. In addition, there is a command for viewing the system history.

### 5.8.1 Project Explorer Command (View Menu)

If you have closed the Project Explorer pane, you can open it by using this command.

For details, refer to *5.2 Project Explorer Pane*.

### 5.8.2 Status Window Command (View Menu)

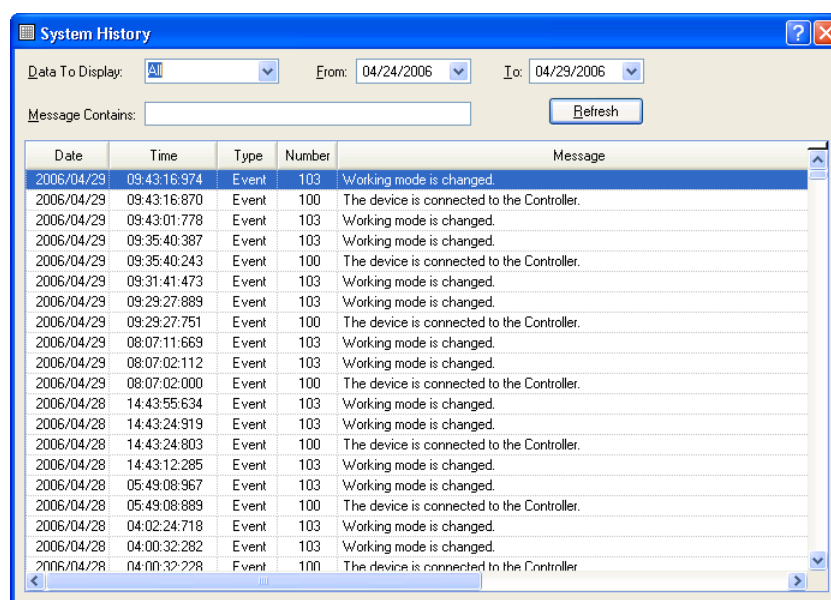
If you have closed the Status Window pane, you can open it by using this command.

For details, refer to *5.3 Status Window Pane*.

### 5.8.3 System History Command (View Menu)

This command opens the System History window. This window shows events, errors, and warnings that have been logged in the current controller's system history.

The data can be sorted by clicking on any column header. To sort multiple columns, hold down the shift key and click on multiple columns headers.



Item	Description
<b>Data To Display</b>	Select which data you would like to view. Choices are All, Events, Errors, and Warnings.
<b>From / To</b>	Select the dates you want to view data from. When the window is first opened, these are automatically set to the first and last dates in the history data.
<b>Message Contains</b>	Type in text to be found in the error message. After typing in the text, click the Refresh button.
<b>Refresh</b>	Click this button to reload the data from the controller.

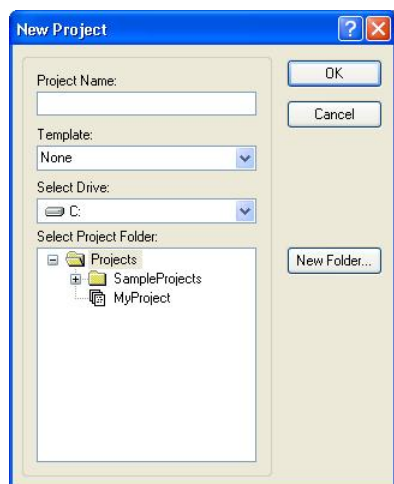
Item	Description
<b>Type Event</b>	Information for operation and mode change.
<b>Warning</b>	Program can be executed continuously, however, needs countermeasure.
<b>Error</b>	Error occurred in the program or the Robot.
<b>Number</b>	For details of the number, refer to <i>SPEL<sup>+</sup> Error Message</i> in the <i>SPEL<sup>+</sup> Language Reference</i> .
<b>Message</b>	
<b>Function and Line number</b>	Function name and the line number are displayed when error occurred while executing a program.
<b>Robot and axis number</b>	Robot and the axis number are displayed when Robot error occurred.
<b>Task number</b>	Task number of the task with error is displayed when error occurred while executing the program. "0" is displayed for others.
<b>Additional information 1 and 2</b>	More details are displayed for some errors. For details, refer to <i>SPEL<sup>+</sup> Error Message</i> in the <i>SPEL<sup>+</sup> Language Reference</i> .

## 5.9 Project Menu

The EPSON RC+ 6.0 Project Menu includes commands for managing and building projects.

### 5.9.1 New Command (Project Menu)

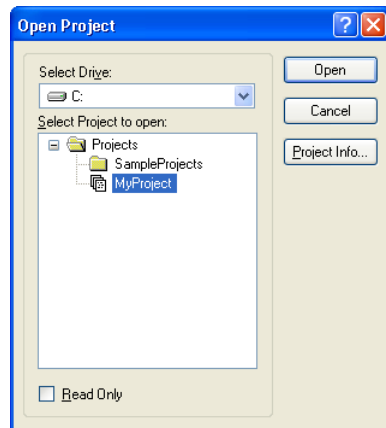
The New command is used to create a new EPSON RC+ 6.0 project. Projects can be on any disk drive on the system. They are stored in the \EpsonRC60\Projects folder on the selected drive. Subfolder can also be created.



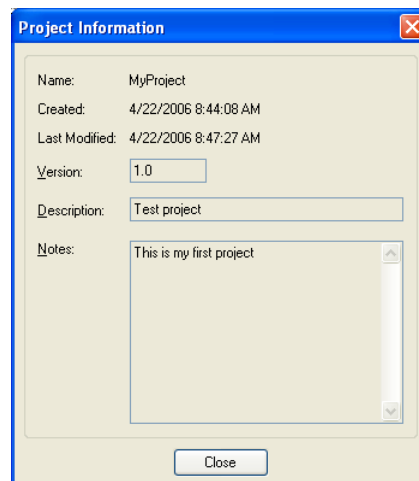
Item	Description
<b>Project Name</b>	Type in a new name for the project. The name can include alphanumeric characters along with underscores.  For a project name, two byte characters such as Japanese, Chinese characters are not allowed.
<b>Template</b>	Select a project template. The new project will be a copy of the template project.
<b>Select Drive</b>	Select the desired disk drive for the new project.
<b>Select Project Folder</b>	This is a list of folders and projects on the selected drive. If you click on a name in this list, it will be displayed in the New Project Name text box. You can then edit the name, or you can create a new project with the same name as one that has already been created. In the later case, you will be prompted to overwrite the old project if it is in the same folder.
<b>New Folder</b>	Creates a new folder in the currently selected folder.
<b>OK</b>	Creates the new project.
<b>Cancel</b>	Aborts creating a new project.

### 5.9.2 Open Command (Project Menu)

Use this command to open an EPSON RC+ 6.0 project. When the project is opened, the previous project is closed. You will be prompted to save changes.



Item	Description
<b>Select Drive</b>	Select the desired disk drive for the project you want to open.
<b>Select Project to Open</b>	Select a project name from the list box. To open a folder, double click on the folder or click the + box located to the left of the folder.
<b>Read Only</b>	If you set this check box and open a project, you can not edit the program file, include file, point file, I/O label, and user error.
<b>Open</b>	Opens the selected project.
<b>Cancel</b>	Cancels the operation.
<b>Project Info</b>	Displays general project properties for the selected project. To view project information, first select a project in the list, then click the <b>Project Info</b> button.

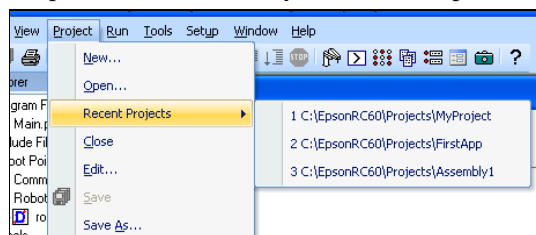


**NOTE** Project information for a project can be changed by selecting Properties from the Project Menu after opening the project.

### 5.9.3 Recent Projects Submenu (Project Menu)

The Recent Projects submenu contains up to eight of the most recently used projects.

When you select a project in the menu, the current project is closed and the selected project is opened the same as if you used the Open command from the Project Menu.



### 5.9.4 Close Command (Project Menu)

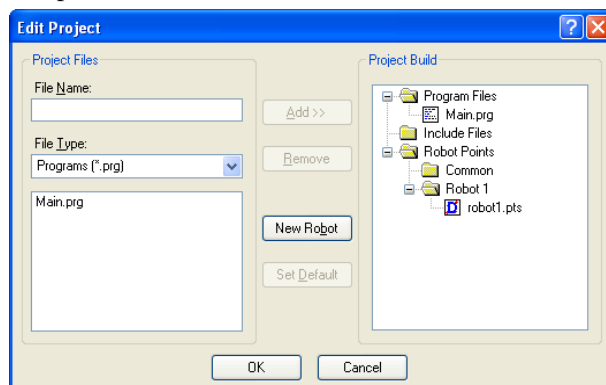
Use the Close command to close the current project. Several menu and toolbar commands will be disabled after the project is closed.

### 5.9.5 Edit Command (Project Menu)

The Edit command is used to define which program files, include files, and point files are to be used in the current project.

The [Project Files] contains a list of files in the current project folder. You can select which files to view from the **File Type** list box.

The [Project Make] contains a project make tree that includes program files, include files, and point files.



The files shown in the file list are in the current project disk directory. Before you can use a file in the project, you must put it into the project make tree using the Add button.

#### To create a new program

1. Type the name of program file in the **File Name** text box in the Program Files section. Add the PRG extension to the file name. For a file name, two byte characters such as Japanese, Chinese characters are not allowed. The program file name must be less than 24 characters (excluding the extension).
2. Click the **Add>>** button. You will be prompted to create a new file. Answer **Yes** to create the file and put it in Program Files folder in the project make tree.

#### To add an existing program file

1. Select the Program in the **File Type** list box.
2. Select the program file name you want to add to the project from the list box.
3. Click the **Add >>** button, or double click on the program file name in the file list box.

The file will be added to the Program Files folder in the project make tree.

**To create a new include file**

1. Type the name of the include file in the **File Name** text box.  
Add the INC extension to the file name. The name of the include file can also be the same name as a program. For a file name, two byte characters such as Japanese, Chinese characters are not allowed.
2. Click the **Add >>** button. You will get a message asking if it's okay to create the new file. Click **Yes** to create the file and put it in the Include Files folder in the project make tree.

**To add an existing include file to the project**

1. Select Include in the **File Type** list box.
2. Select the include file name you want to add to the project from the list box.
3. Click the **Add >>** button, or  
double click on the include file name in the file list box.

The file will be added to the list of include files for project make.

**To add a new point file**

1. Type the name of the point file you want to create into the **File Name** text box.  
Add the PTS extension. For a project name, two byte characters such as Japanese, Chinese characters are not allowed.
2. Select the robot folder you want to register from the Robot Points folder in the project make tree.
3. Click the **Add >>** button. You will be prompted to create a new file. Click **Yes** to create the file and put it in the selected robot of the Robot Points folder.

**To add an existing point file to the project**

1. Select Points from the **File Type** list box.
2. Select the robot folder you want to register from the Robot Points folder in the project make tree.
3. Select the point file name you want to add to the project from the list.
4. Click the **Add >>** button. The file will be put in the selected robot of the Robot Points folder.

**To remove a program file, include file, or point file**

1. Select the file you want to remove in the project make tree.
2. Click the **Remove** button to remove the file from project make. The file is not deleted from the project folder, so you will still see the file in the file list.

**To add a new robot**

Click the **New Robot** button. A robot will be added to the Robot Points folder in the Project Build tree.

**To set a default point file**

1. Select a point file to set as the default from each robot of Robot Points folder in the Project Build tree.
2. Click the **Set as default** button. The file will be set as the default of the registered robot.




The common point file is a point file that is available for all robots on the controller. To use this point file, you need to load it from the SPEL<sup>+</sup> program to the robot using LoadPoints command.

The default point file is a point file that is automatically loaded to a robot with the project load. Each robot can have one point file as the default.


### 5.9.6 Save Command (Project Menu)

#### Shortcuts

Toolbar: 

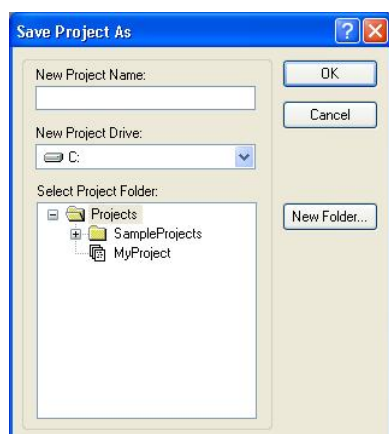
This command saves the active program file, include file, point file, I/O labels, or user errors. This menu selection will be dimmed if nothing needs to be saved.



It's a good idea to save files frequently while you are editing project files. Just click the disk button  on the toolbar to save all of your files.

### 5.9.7 Save As Command (Project Menu)

This command saves all files in the current project to a new drive and/or project name. The current project will be preserved.

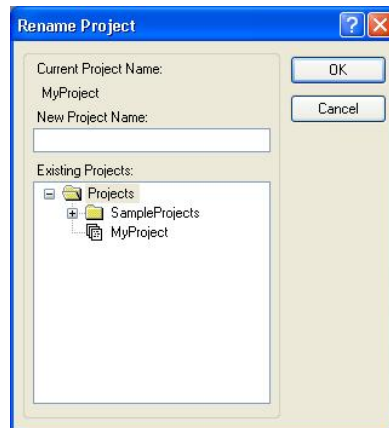


Item	Description
<b>New Project Name</b>	Type in a new name for the project. The name can include alphanumeric characters along with underscores but cannot include two byte characters such as Japanese, Chinese characters. The maximum number of characters is 24. You can use the same name as the current project if you select a drive and folder that is not the same as the current project folder and the folder drive.
<b>New Project Drive</b>	Drives for the new project location.
<b>Select Project Folder</b>	Click on the desired folder for the project.
<b>New Folder</b>	Click this button to create a new folder under the Projects folder.
<b>OK</b>	Saves the project using the new name and location.
<b>Cancel</b>	Cancels the operation.



### 5.9.8 Rename Command (Project Menu)

This command renames the current project. The project folder and all associated project files are also renamed.



Item	Description
<b>New Project Name</b>	Type in a new name for the project. The name can include alphanumeric characters along with underscores but cannot include two byte characters such as Japanese, Chinese characters.
<b>Existing Project</b>	This list box shows other projects on the selected drive. The new name you choose cannot be one of the names in this list.
<b>OK</b>	Renames the project.
<b>Cancel</b>	Cancels the operation.

### 5.9.9 Import Command (Project Menu)

The Project Menu Import Command uses a wizard to import projects from a PC, the current controller, or a controller status folder.

When a project is imported, the files are copied to a new project folder, so the original project is not changed.



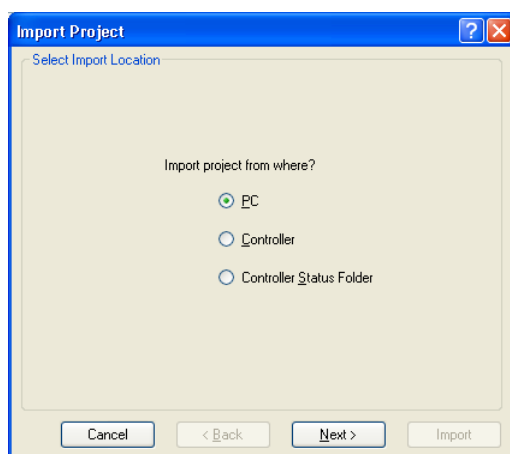
If the project to be imported is an EPSON RC+ 3.x / 4.x project or a SPEL for Windows 2.0 project, the files are converted to EPSON RC+ 6.0 format.

The sections below have instructions for importing a project from each type of source location.

#### Importing a PC project

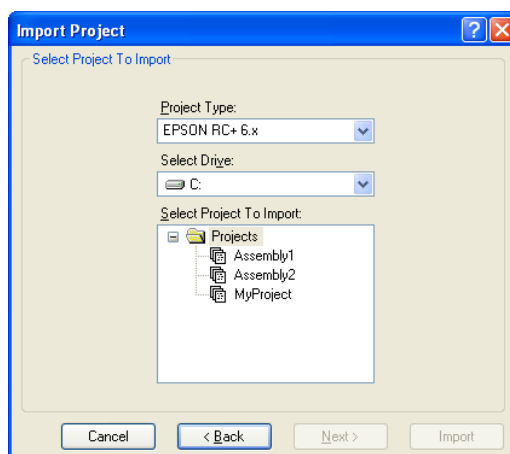
Follow these steps to import a project from a PC:

1. Select Import from the Project Menu to open the Import Project dialog.
2. Select **PC** and click **Next**.



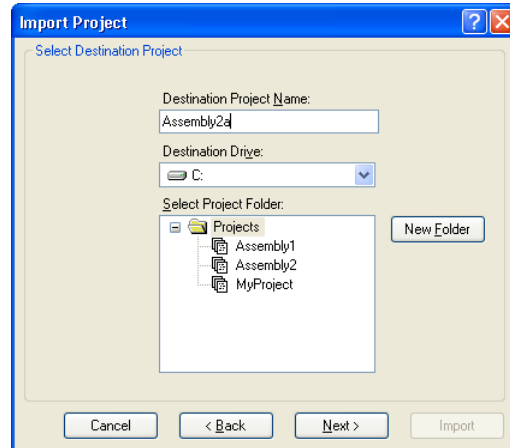
3. Select the project type. You can select from the following:

- EPSON RC+ 6.0
- EPSON RC+ 3.\* / 4.\* / 5.\*
- SPEL for Windows 2.0

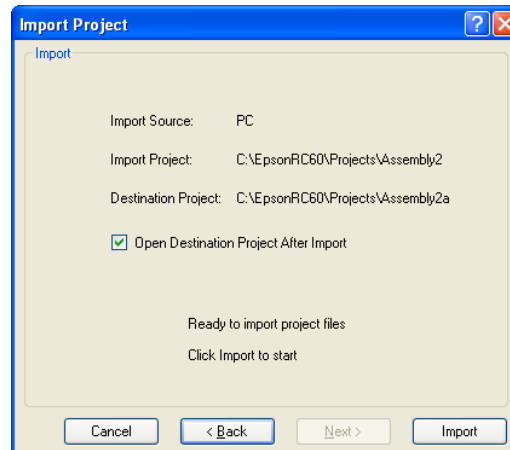


When project for EPSON RC+ 3.\* / 4.\* / 5.\* or SPEL for Windows 2.0 is imported, the project is converted to project for EPSON RC+ 6.0 by automatic processing. For details, refer to *Appendix A: Automatic Processing of Project Import*.

4. Select the drive. After you select the project type and drive, the project list will be updated to show the projects available for import. Select the project to import in the list and click **Next**.
5. The new project name is set to the name of the imported project. You can modify the destination project name if desired. Select the destination drive and project folder, then click **Next**.



6. Verify the import source, import project, and destination project. Check **Open Destination Project After Import** if you want the project to open after import.

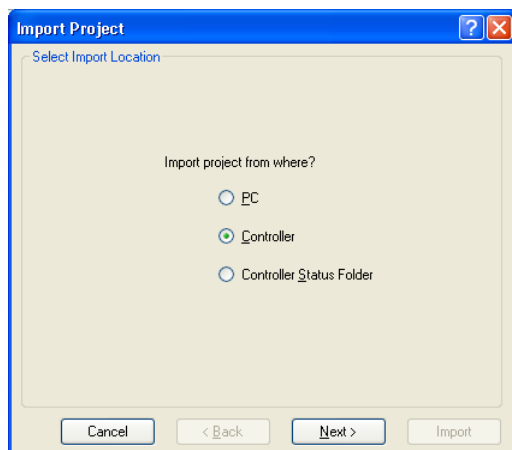


7. Click the **Import** button. If the destination project already exists, you will be asked if you want to overwrite it.

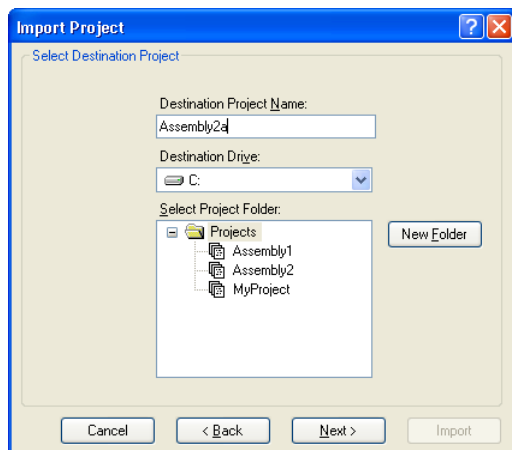
### Importing a Controller project

Follow these steps to import a project from a controller:

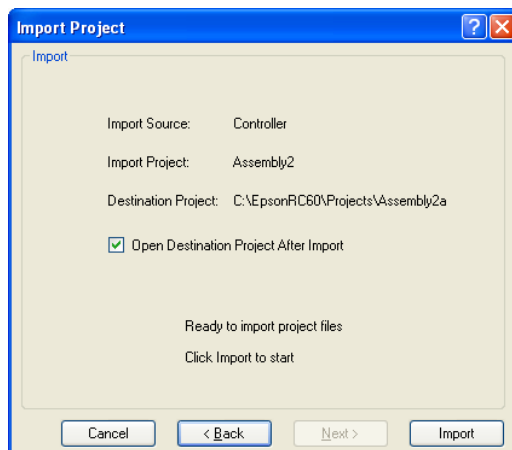
1. Select Import from the Project Menu to open the Import Project dialog.
2. Select **Controller** and click **Next**.



3. The new project name is set to the name of the current project in the controller. You can modify the new project name if desired. Select the destination drive and project folder, then click **Next**.



4. Verify the import source, import project, and destination project. Check **Open Destination Project After Import** if you want the project to open after import.



5. Click the **Import** button. If the destination project already exists, you will be asked if you want to overwrite it.

- The project in the destination project will be built.

## NOTE

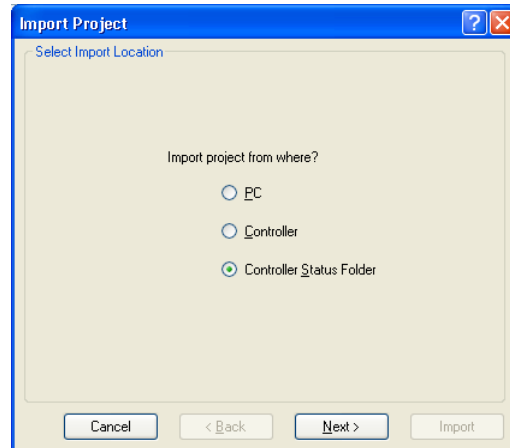


### Importing a Controller Status project

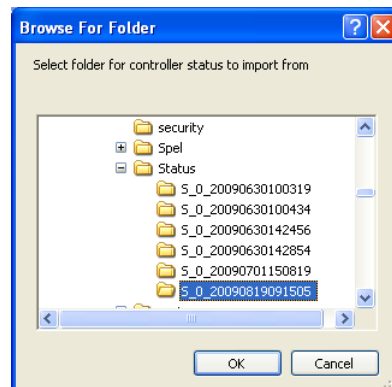
The projects using Vision Guide cannot be imported from the Controller Status Folder.

Follow these steps to import a project from a Controller Status Folder:

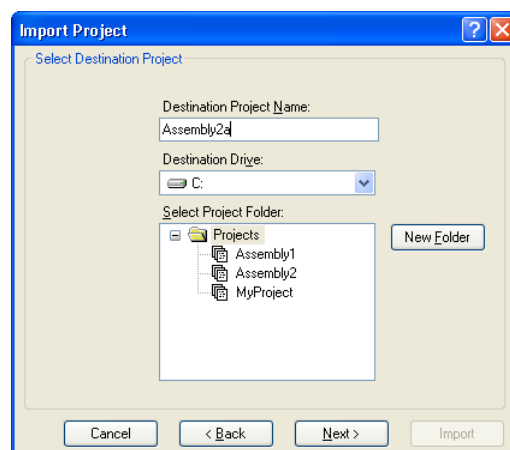
- Select Import from the Project Menu to open the Import Project dialog.
- Select **Controller Status Folder** and click **Next**.



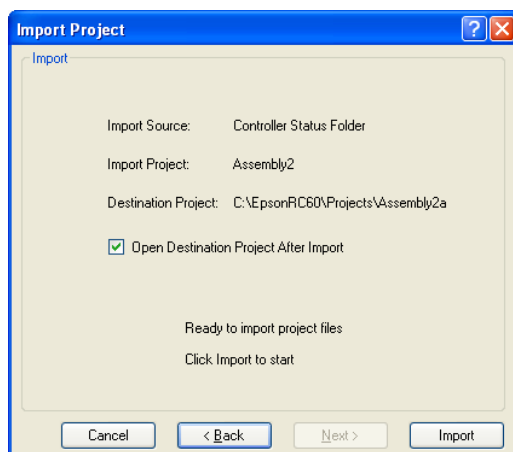
- Select a controller status folder and click **OK**.



- The new project name is set to the project found in the controller status folder. You can modify the new project name if desired. Select the destination drive and folder, then click **Next**.



5. Verify the import source, import project, and destination project. Check **Open Destination Project After Import** if you want the project to open after import.

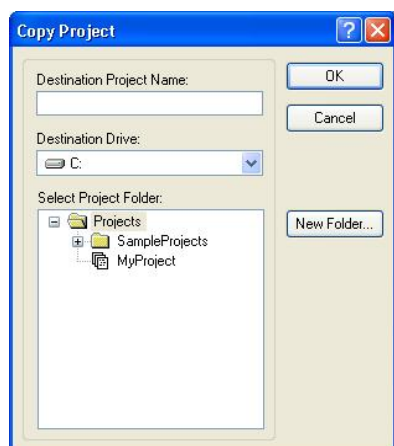


6. Click the **Import** button. If the destination project already exists, you will be asked if you want to overwrite it.

#### 5.9.10 Copy Command (Project Menu)

The Copy command copies all files in the current project to a specified drive, folder, and project name. You can use the current project name for the destination name if you select a new drive or folder. You can also specify a new name for the destination project.

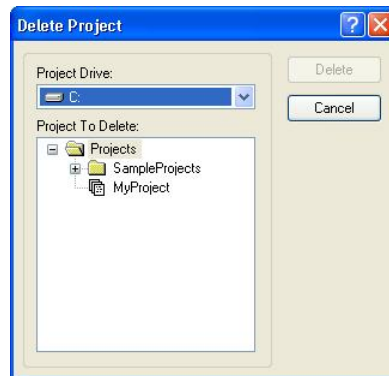
You should use the Copy command to make backup copies of your project on a regular basis.



Item	Description
<b>Destination Project Name</b>	Type in a name for the new copy of the project. The name can include alphanumeric characters along with underscores but cannot include two byte characters such as Japanese, Chinese characters. The maximum number of characters is 24. You can use the same name as the current project if you select a drive and folder that is not the same as the current project's drive and folder.
<b>Destination Drive</b>	Drives for the project copy.
<b>OK</b>	Performs the copy process.
<b>Cancel</b>	Cancels the operation.

### 5.9.11 Delete Command (Project Menu)


This command deletes an entire project from a PC disk. All files in the project folder will be destroyed.



Item	Description
<b>Project Drive</b>	Select drive for the project to delete.
<b>Project To Delete</b>	Select a project to delete from the list.
<b>Delete</b>	Delete the project. You will be prompted to confirm the operation.
<b>Cancel</b>	Cancel the operation.

### 5.9.12 Build Command (Project Menu)

#### Shortcuts

Toolbar:  Keys: Ctrl + B

This command builds the current project so that it can be executed. The Build command does the minimum amount of work required to bring the project up to date in the robot controller. For example, if a change was made to one program file in the project, then Build will compile the changed file, link it with the remaining object files (if they exist), and send the new files to the controller.

During the build process, the status window displays each step of the build. If there are any errors, they will be displayed on the status window.

### 5.9.13 Rebuild Command (Project Menu)

#### Shortcuts

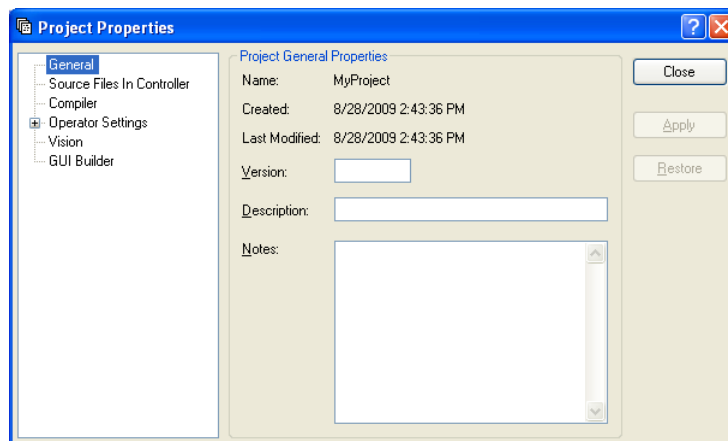
Keys: Ctrl + Shift + B

Rebuilds the entire current project. All program files are re-compiled, linked, and sent to the controller. All point files in the project are sent to the controller.

## 5.9.14 Properties Command (Project Menu)

**Project: Properties: General Page**

Use this page to view and edit general properties for the current project. All project property settings are stored in the project file, which is also stored in the controller during project build.



Item	Description
<b>Name</b>	The name of the current project.
<b>Created</b>	Date and time when the project was created.
<b>Last Modified</b>	Date and time when the project was last modified.
<b>Version</b>	User version number of the project. You can type any text here.
<b>Description</b>	A description of the project. You can type any text here.
<b>Notes</b>	Any project notes can be entered into this section.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Revert back to previous values.
<b>Close</b>	Close the Project Properties dialog.



TIP

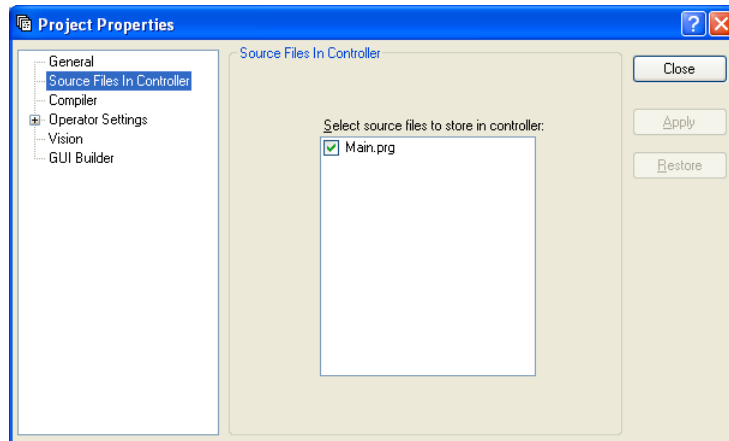
When the Open Project dialog is used, clicking the **Project Info** button will open a dialog that contains the general project properties entered on this page.



### Project: Properties: Source Files In Controller Page

This page allows you to select which source files will be stored in the controller during project build.

After changes are applied, the next project build will clear the project in the controller and perform a rebuild.



Item	Description
<b>Select Source Files To Store in Controller</b>	This is a list of the source files in the project. Select which source files you want to have stored in the controller.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

**Project: Properties: Encrypted Files Page**

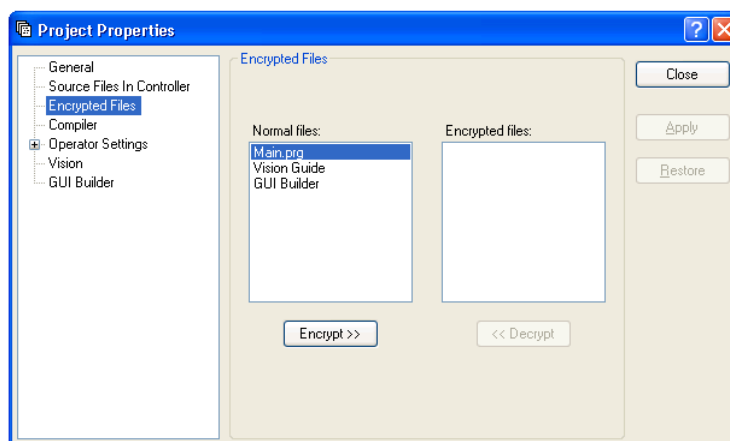
This page allows you to encrypt files in the current project.

For details on using encrypted files, refer to section 7.8 *Using Encrypted Files*.



■ **USE EXTREME CAUTION!**

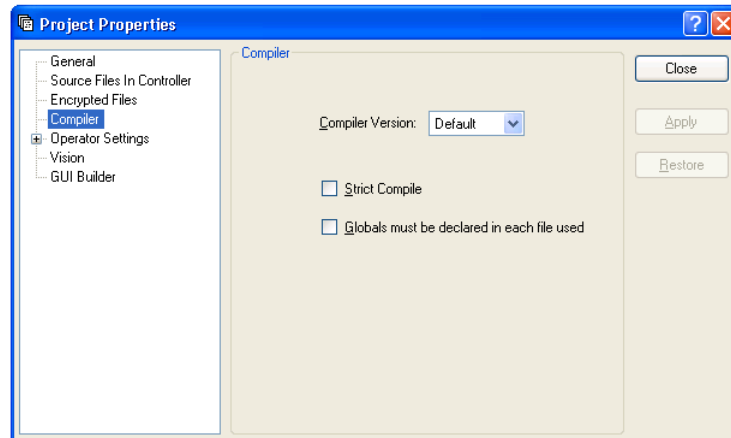
Keep a record of the password(s) used for encryption in a safe place. Once a file is encrypted, it can only be opened with the password you enter. If you forget the password, the file contents **CANNOT BE RECOVERED**



Item	Description
<b>Normal Files</b>	This is a list of the source files in the project that are not encrypted. Select which source files you want to encrypt.
<b>Encrypted Files</b>	This is a list of the source files in the project that are encrypted. Select which source files you want to decrypt.
<b>Encrypt &gt;&gt;</b>	Encrypts the files selected in the Normal files list. When this button is clicked, you will be prompted for a password that will be used to access these encrypted files.
<b>&lt;&lt;Decrypt</b>	Decrypts the files selected in the Encrypted files list. When this button is clicked, you will be prompted for the password that was used to encrypt the files.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

## Project: Properties: Compiler Page

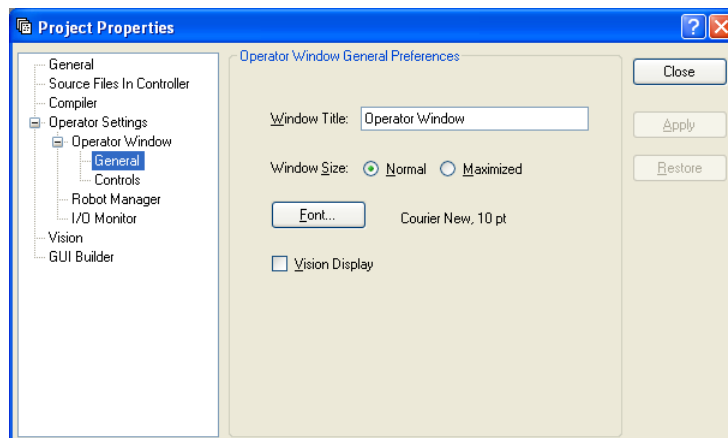
This page allows you to configure the compiler settings.



Item	Description
<b>Compiler Version</b>	<p>[Default] is the normal setting.</p> <p>When the projects cannot be built because new SPEL+ language keywords have been added that conflict with your variable names, you can select a previous version to build the projects. Specify the controller version that compiles the project.</p>
<b>Strict Compile</b>	<p>Check the Boolean type strictly.</p> <p>When Strict Compile is checked, you cannot assign integer values to Boolean variables.</p>
<b>Globals must be declared in each file used</b>	<p>To improve link times for large projects that use many global variables.</p> <p>When this item is checked, you must declare global variables in each file that they are used in, otherwise an error will occur.</p>
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the [Project Properties] dialog.

**Project: Properties: Operator Settings: Operator Window: General Page**

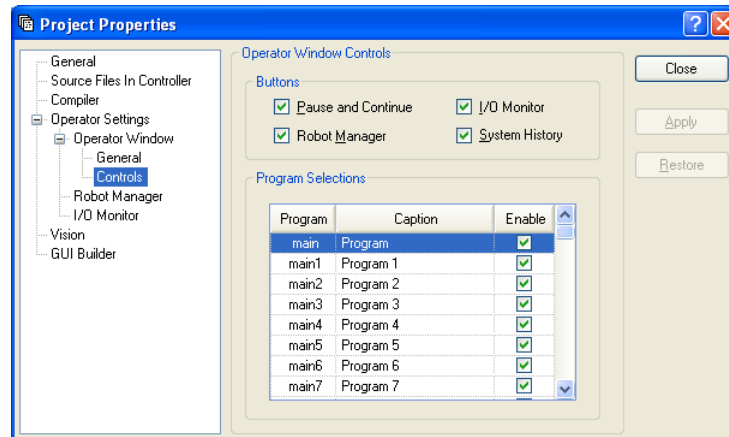
This page allows you to configure the general settings for the Operator Window.



Item	Description
<b>Window Title</b>	Type in the title that you want to appear at the top of the operator window.
<b>Window Size</b>	Choose Normal or Maximized.
<b>Font</b>	Click on the Font button to open the fonts dialog. Choose the font you desire for the operator window. The current font name and size is displayed next to the Font button.
<b>Vision Display</b>	If this check box is set, the Vision Guide image will be displayed in the operator window.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

**Project: Properties: Operator Settings: Operator Window: Controls Page**

This page allows you to configure the controls for the Operator Window.



Item	Description
<b>Pause and Continue</b>	Check this box if you want the Pause and Continue buttons to be displayed. This will allow the operator to pause and continue from the operator window.
<b>I/O Monitor</b>	Check this box if you want the I/O Monitor button to be displayed. This will allow the operator to view input and output status.
<b>Robot Manager</b>	Check this box if you want the Robot Manager button to be displayed. This will allow the operator to open the Robot Manager from the operator window.
<b>System History</b>	If this check box is set, the <System History> button will appear. You can check the system history.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

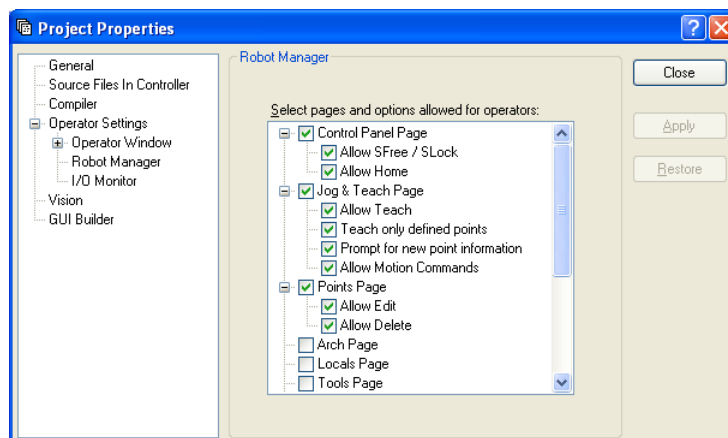
**Program Selections Details**

Each project can have up to 64 programs that can be started from the Operator Window. The programs are named main, main1, main2, ... main63. Each program has an associated startup function using the same name as the program (main, main1, main2...main63).

In the program selections grid, you can define a friendly name for each of the 64 programs. You can also define which selections will be displayed in the Operator Window program list by checking the Enable checkbox.

**Project: Properties: Operator Settings: Robot Manager Page**

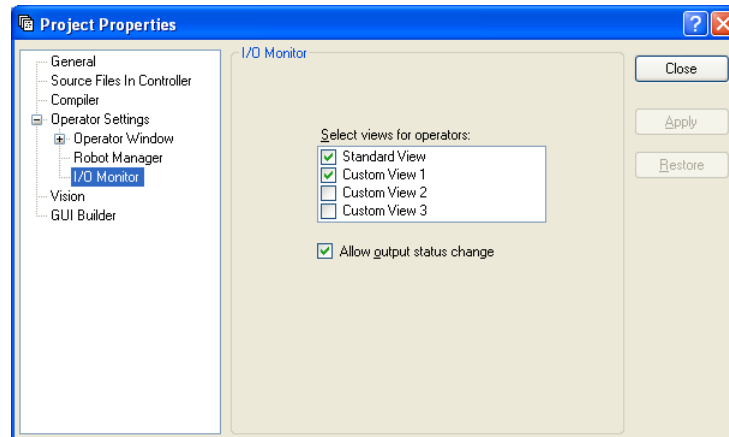
Use this page to configure the Robot Manager for operators.



Item	Description
<b>Page and options enabled for operators</b>	Check the pages that you want the operator to have access to when the Robot Manager is displayed from the operator window. In some pages, there are additional options.
<b>Allow SFree / SLock</b>	Allows the operator to free or lock joints from the Control Panel page.
<b>Allow Home</b>	Allows the operator to home the robot from the Control Panel page.
<b>Allow Teach</b>	Allows the operator to teach points from the Jog & Teach page.
<b>Teach only defined points</b>	Only defined points are shown in the point list on the Jog & Teach page.
<b>Prompt for new point information</b>	When the operator teaches a new point, a dialog will be displayed for entering the point label and description.
<b>Allow Motion Commands</b>	Allows the operator to execute motion commands from the Jog & Teach page.
<b>Allow Edit</b>	Allows the operator to edit point data on the Points page.
<b>Allow Delete</b>	Allows the operator to delete points on the Points page.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

**Project: Properties: Operator Settings: I/O Monitor Page**

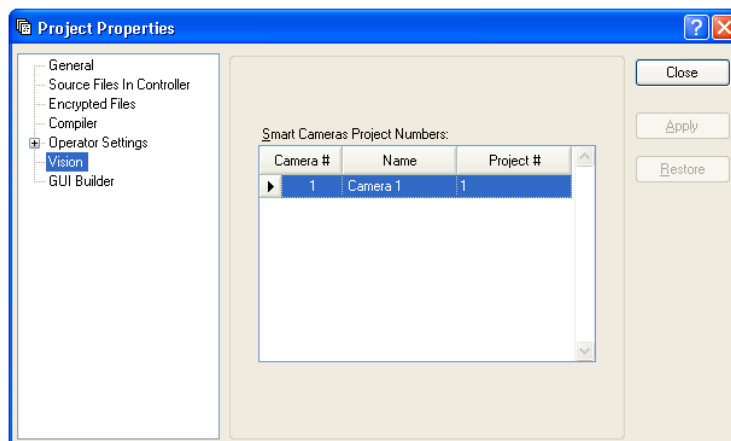
Use this page to configure the I/O Monitor for operators.



Item	Description
<b>Views Enabled for Operators</b>	Configures the I/O views that operators use when opening the I/O Monitor from the operator window. You can configure the custom views.
<b>Allow output status change</b>	Check this box if you want to allow operators to turn outputs on or off.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

### Project: Properties: Vision

The EPSON Smart Camera supports two vision projects simultaneously. Each vision project can be used by one controller, so two controllers can use the same camera. Project 1 is used by default. In this page, you can configure the project number used by each Smart Camera for this project.

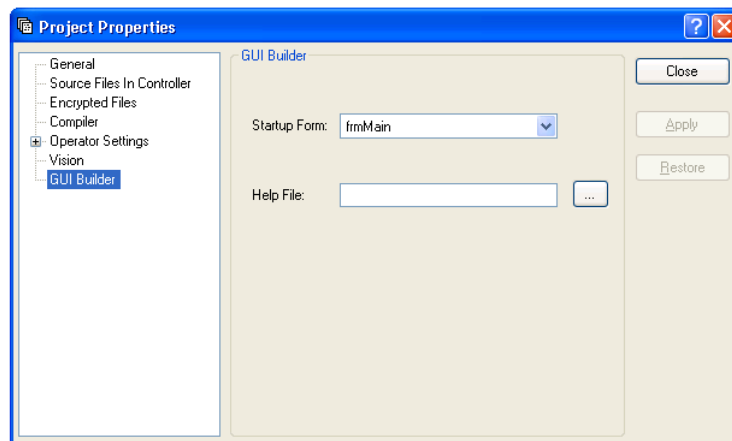


Item	Description
<b>Smart Camera Project Numbers</b>	Select the project number used by each Smart Camera for this project in the "Project #" column.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.



## Project: Properties: GUI Builder

On this page, you can specify the startup form for GUI Builder and also set the value of the help file used in your project.



Item	Description
<b>Startup Form</b>	Select the startup form for the current project. If no forms have been created in GUI Builder, then there will be no forms in the list.
<b>Help File</b>	Set help file that will be used by forms in GUI Builder.
<b>Apply</b>	Set current values after changes have been made.
<b>Restore</b>	Reverts back to the previous values.
<b>Close</b>	Closes the Project Properties dialog.

### 5.10 Run Menu

The EPSON RC+ 6.0 Run Menu includes commands for running and debugging programs.

#### 5.10.1 Run Window Command (Run Menu)

##### Shortcuts

Toolbar:  Key: F5

Opens the [Run] window to run a program.

Before opening the [Run] window, all files will be saved automatically if there are any unsaved files and then the project will be built. If there are any errors during build, the Run window will not be opened.

(If the *Auto File Save* preference is off in Setup | Preferences | Workspace, you will be prompted to save all files if there are any unsaved files.)

After the [Run] window opens, you must click on the Start button to initialize program execution.

For more information, see *7.5.1 The Run Window*.

#### 5.10.2 Operator Window Command (Run Menu)

##### Shortcuts

Keys: Shift + F5

Opens the [Operator] window.

Before opening the [Operator] window, all files will be saved automatically if there are any unsaved files and then the project will be built. If there are any errors during build, the [Operator] window will not be opened.

(If the *Auto File Save* preference is off in Setup | Preferences | Workspace, you will be prompted to save all files if there are any unsaved files.)

If the project is ready to run (last build was successful), then the [Operator] window will be opened.

For more information, see *7.6 The Operator Window*.

#### 5.10.3 Step Into Command (Run Menu)

##### Shortcuts

Toolbar:  Key: F11

Execute the current source line. If the current line is a function, the next step will be the first line in the function.

#### 5.10.4 Step Over Command (Run Menu)

##### Shortcuts

Toolbar:  Key: F10

Execute the current source line. If the current line is a function, the entire function will be executed.

### 5.10.5 Walk Command (Run Menu)

#### Shortcuts

Key: F12

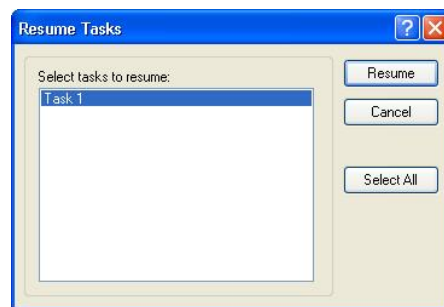
Execute lines until after the next motion command or output command, depending on the *Walk stops for output commands* preference on the Setup | System Configuration | Controller | General page.

### 5.10.6 Resume Command (Run Menu)

#### Shortcuts

Toolbar:  Key: F7


Opens the [Resume Tasks] dialog box. Use this command to resume one or more halted tasks. This command is available only when one or more tasks is in halt mode.



Item	Description
<b>Select tasks to resume</b>	A list of all currently halted tasks. Click on one or more tasks to resume.
<b>Resume</b>	Click to resume.
<b>Select All</b>	Click to select all of the tasks in the list.
<b>Cancel</b>	Cancel the operation and close the dialog.

### 5.10.7 Stop Command (Run Menu)

#### Shortcuts

Toolbar: 

Stops all tasks. This command is disabled when no tasks are running.

### 5.10.8 Toggle Breakpoint Command (Run Menu)

#### Shortcuts

Toolbar:  Key: F9

Sets the selected line as a breakpoint or returns it to normal. When a line is a breakpoint, a breakpoint icon is displayed in the program window left margin.

You can set breakpoints while tasks are running.

If a line cannot be a breakpoint (such as a blank line), then the breakpoint icon will not appear for that line.

## 5.10.9 Clear All Breakpoints Command (Run Menu)

**Shortcuts**

Keys: Ctrl + Shift + F9

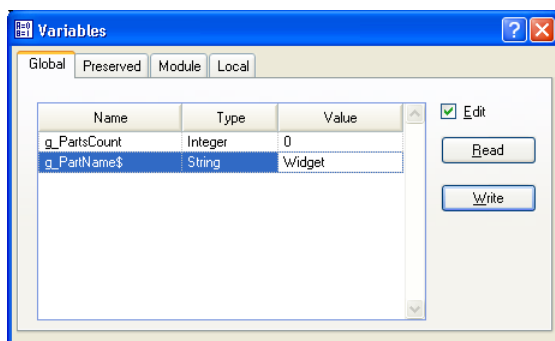
Clears all breakpoints.

## 5.10.10 Display Variables Command (Run Menu)

**Shortcuts**

Key: F4

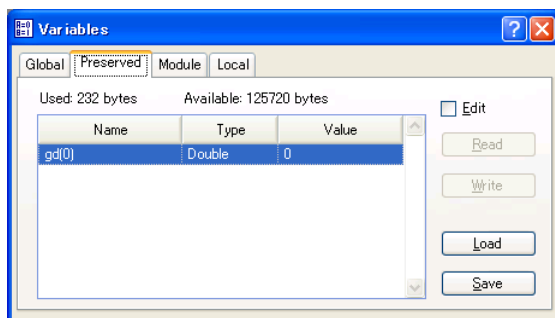
Displays a dialog box that shows the values for all variables in robot controller memory.



To change a variable value

1. Check the **Edit** checkbox.
2. Type the new value in the Value column. As you type in new values, the text color changes to red, indicating that the value is new and as not been written.
3. Click the **Write** button to save the changes. Click **Read** or uncheck Edit to cancel changes and restore the previous values.

When an array is displayed, the first element is shown. You can change which element to view by typing in the desired array subscript and then clicking the **Read** button.



The Preserved page displays the Global Preserve variables. The numbers of used and available bytes for preserved variables are also displayed.

You can save the values of Global Preserve variables in the controller to a file on the PC by clicking the **Save** button. The default file name is "GlobalPreserves.dat".

A "GlobalPreserves.dat" file is also saved by using Backup Controller from the Tools Menu.

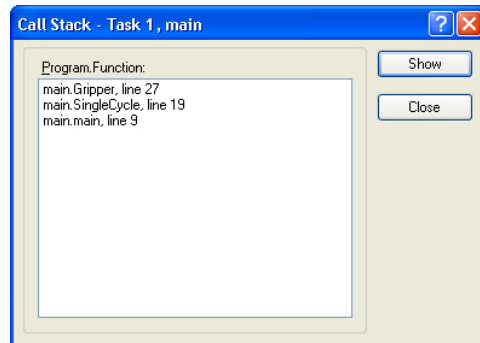
You can load the global preserve variables that are stored in the file on the PC by clicking **Load** button.

For module variables, you must select the desired program.

Local variables are not displayed unless one or more tasks have reached a breakpoint or have been halted from the Task Manager. You can view local variables for each function in the call stack for each halted task.

#### 5.10.11 Call Stack Command (Run Menu)

The Call Stack dialog displays the function call stack for one task.



The Call Stack command is available when a program window is clicked which contains a function that is currently halted.

The most recent function is at the top of the list, and parent functions are listed afterwards in descending order. The last function is the task function.

Each row in the list shows a program, function, and line number.

You can view the code for any of the function calls in the list by selecting a function, then clicking **Show**. The program window for the function you selected is then displayed and the line of the function call is marked by a yellow arrow in the editor left margin.

### 5.11 Tools Menu

EPSON RC+ 6.0 has several GUI tools to support the system development. All tools can be accessed from the Tools Menu. Many also have tool bar buttons and hot keys.

The Tools Menu includes the following selections:

- **Robot Manager**  
Motor control, Jog & Teach, change robot parameters.
- **Command Window**  
Execute SPEL<sup>+</sup> commands directly.
- **I/O Monitor**  
Monitor and change I/O status.
- **Task Manager**  
Monitor and control task status.
- **Macros**  
Opens the Macro Window.
- **I/O Label Editor**  
Edit I/O labels.
- **User Error Editor**  
Edit user errors.
- **Controller**  
Do maintenance on the controller, such as backup, restore, and export status.

#### 5.11.1 Robot Manager Command (Tools Menu)

##### Shortcuts

Toolbar:  Key: F6

This command opens the Robot Manager window. This window contains several tabs that are used to control the robot motors and power, jog the robot and teach points, and view/edit several parameters for the robot.

You can configure how the Robot Manager window can be viewed in the development environment from the Setup | Preferences | Robot Manager | General page.

**MDI window** The Robot Manager is displayed as a child window along with the other child windows inside the EPSON RC+ 6.0 development environment main window.

**Dialog** The Robot Manager is displayed as a modal dialog which is displayed in the foreground over the development environment main window.

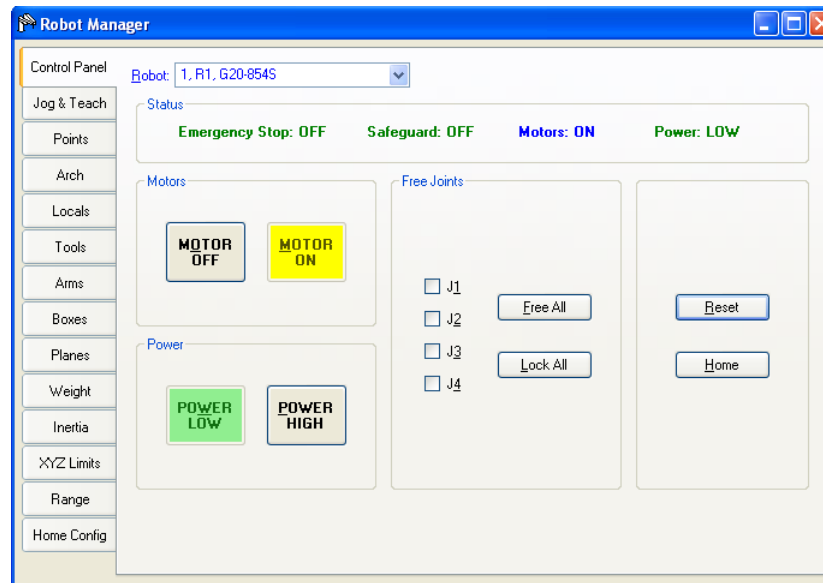
##### NOTE



If the screen resolution is less than 1024 x 768, the Robot Manager will always be displayed in dialog mode so it can fit on the screen.

## Tools: Robot Manager: Control Panel Page

The Control Panel page contains buttons for basic robot operations, such as turning motors on/off and returning the robot to the home position. It also shows status for Emergency Stop, Safeguard, Motors, and Power.



### Status Indicators

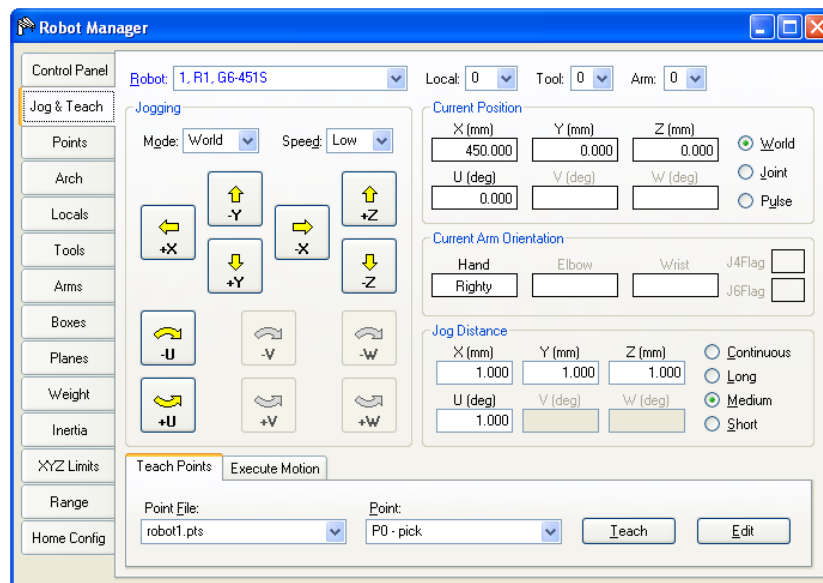
Indicator	Description
<b>Emergency Stop</b>	Indicates if Emergency Stop has occurred. Once an Emergency Stop occurs, you must execute a Reset to clear it.
<b>Safeguard</b>	Indicates whether the Safeguard input is on or off.
<b>Motors</b>	Indicates whether the robot motors are on or off.
<b>Power</b>	Indicates whether the robot motor power is high or low.

Controls	Description
<b>Robot</b>	Select a robot.
<b>MOTOR OFF</b>	Turns off all robot motors for the robot.
<b>POWER LOW</b>	Puts the robot servo system in low power mode.
<b>POWER HIGH</b>	Puts the robot servo system in high power mode.
<b>J1 to J4 checkboxes</b>	You can free one or more joints using the checkboxes. These will be disabled for 6-axis robots.
<b>Free All</b>	Click this button to free all joints from servo control.
<b>Lock All</b>	Click this button to lock all joints under servo control.
<b>Reset button</b>	Resets the robot servo system and Emergency Stop condition.
<b>Home button</b>	Moves the robot to the position specified with the HomeSet command.

## Tools: Robot Manager: Jog and Teach Page

The Jog & Teach page is primarily used for jogging the robot to a desired position and teaching a point using the current coordinates and orientation.

You can jog the robot in World, Tool, Local, Joint, or ECP modes. You can also execute motion commands.



### Jog Controls

The Robot Manager Jog & Teach page contains several controls, described below.

#### Robot

Select a robot.

#### Jogging Group

This group contains controls for setting jog mode, speed, and jog buttons.

#### Mode

This dropdown list contains the following choices jog mode.

- World** Jogs the robot along the X, Y, Z axes in the current local, tool, arm, and ECP. For robots with 4 DOF, you can also jog U (roll). For robots with 6 DOF, you can jog U (roll), V (pitch), and W (yaw). This is the default setting.
- Tool** Jogs the robot in the coordinate system defined by the current tool.
- Local** Jogs the robot in the coordinate system defined by the current local.
- Joint** Jogs each joint of the robot. A separate set of jog buttons will appear when using joint mode when using non-Cartesian robots.
- ECP** Jogs the robot along the axes of the coordinate system defined by the current external control point. Coordinates are World coordinates.



## Speed

The speed for jogging and motion commands can be changed by selecting Low or High. When the Robot Manager is first open, the speed is set to Low. Jogging is always in low power mode. The speeds and accelerations associated with the jog speed settings are shown in the next page.

### SCARA robot RS series PS series

Jog Speed	Jog Method	Speed	Accel	Decel
Low	Continuous World/Tool/ECP XYZ	10 mm/sec	100 mm/sec <sup>2</sup>	200 mm/sec <sup>2</sup>
	Continuous World/Tool/ECP UVW	2 deg/sec	20 deg/sec <sup>2</sup>	40 deg/sec <sup>2</sup>
	Continuous Joint	*	10 deg/sec <sup>2</sup>	20 deg/sec <sup>2</sup>
	Step	1/5 of default PTP speed	Default PTP acceleration	Default PTP deceleration
High	Continuous World/Tool/ECP XYZ	50 mm/sec	100 mm/sec <sup>2</sup>	200 mm/sec <sup>2</sup>
	Continuous World/Tool/ECP UVW	10 deg/sec	20 deg/sec <sup>2</sup>	40 deg/sec <sup>2</sup>
	Continuous Joint	*	10 deg/sec <sup>2</sup>	20 deg/sec <sup>2</sup>
	Step	Default PTP speed	Default PTP acceleration	Default PTP deceleration

\*Continuous joint speed depends on robot model

### 6-Axis robot

Jog Speed	Jog Method	Speed	Accel	Decel
Low	Continuous World/Tool/ECP XYZ	10 mm/sec	200 mm/sec <sup>2</sup>	400 mm/sec <sup>2</sup>
	Continuous World/Tool/ECP UVW	2 deg/sec	20 deg/sec <sup>2</sup>	40 deg/sec <sup>2</sup>
	Continuous Joint	*	20 deg/sec <sup>2</sup>	40 deg/sec <sup>2</sup>
	Step	1/5 of default PTP speed	Default PTP acceleration	Default PTP deceleration
High	Continuous World/Tool/ECP XYZ	*	200 mm/sec <sup>2</sup>	400 mm/sec <sup>2</sup>
	Continuous World/Tool/ECP UVW	15 deg/sec	20 deg/sec <sup>2</sup>	40 deg/sec <sup>2</sup>
	Continuous Joint	*	20 deg/sec <sup>2</sup>	40 deg/sec <sup>2</sup>
	Step	Default PTP speed	Default PTP acceleration	Default PTP deceleration

\*Continuous joint speed and High Continuous XYZ depends on robot model

## Jog Buttons

The jog buttons are used to jog the robot throughout the work envelope. They can be actuated only with the mouse.

You can jog one step at a time by setting the Jog Distance to Long, Medium, or Short and then clicking on a button and releasing. You can step continuously by holding the button down. To jog continuously without stepping, set the Jog Distance to Continuous. See *How to jog robot* for details

You can change the orientation of the jog buttons to align your PC monitor with the robot from Setup: Preferences: Robot Manager: Jogging.

The jog buttons that are displayed depend on the Jog mode. For World, Local, Tool, and ECP jogging, the X, Y, Z, U, V, W buttons appear (V and W are enabled only for 6-Axis robots). For Joint jogging, the joint buttons appear, labeled J1 - J6.

The X, Y, and Z buttons jog the robot along the associated Cartesian axis.

The U buttons rotate the tool coordinate system about the Z axis. This is also known as *roll*.

For 6-Axis robots, the V buttons rotate the tool coordinate system about the Y axis. This is also known as *pitch*. The W buttons rotate the tool coordinate system about the X axis. This is also known as *yaw*.

#### Local

This drop down list is used to select the current Local for jogging and teaching. Only Locals that have been defined are shown in the list. When you teach a point, the Local point attribute defaults to the current local number.

#### Tool

This drop down list is used to select the current Tool for jogging and teaching. Only Tools that have been defined are shown in the list.

#### Arm

This drop down list is used to select the current Arm for jogging and teaching. Only Arms that have been defined are shown in the list. Arms are not used with 6-axis robots.

#### ECP

This drop down list is used to select the current ECP for jogging. Only ECPs that have been defined are shown in the list. ECPs are only allowed if the External Control Point option has been activated.

### Current Position Group

This group displays the current position of the robot. There are three ways to display position. **World** displays the current position and tool orientation in the selected local coordinate system, **Joint** displays the current joint values, and **Pulse** displays the current encoder pulse count for each joint.

### Current Arm Orientation Group

This group displays the current arm orientation.

6-axis robot : Hand orientation, Elbow orientation, wrist orientation,  
J4Flag value, J6Flag value

RS series : Hand orientation, J1Flag value, J2Flag value

Others : Hand orientation

### Jog Distance Group

This group contains text boxes that are used to specify the distance that each axis moves when its corresponding jog button is pressed. There are radio buttons for selecting Continuous, Long, Medium, and Short jog distances. When Continuous is selected, the robot is jogged in continuous mode and the jog distance text boxes are grayed out. When Long, Medium, or Short are selected, the robot is jogged in step mode for the distance specified in the jog distance text box for the axis being jogged.

To change a jog distance, first select the distance to be changed, then type in the new value.

Distance	Set Value *	Default Value
Short	0 to 10	0.1
Medium	0 to 30	1
Long	More than 0 to 180	10

\* If you enter a too large value, an error message appears when you attempt to jog.

When the jog mode is changed, the jog distance units change appropriately between millimeters (mm) and degrees (deg).



When the jog distance is longer than the default, jog distance is reset to default status by rebooting the controller.

### Teach Points Tab

This tab shows the current point file name and point number.

Use the **Teach** button to register the current robot position.

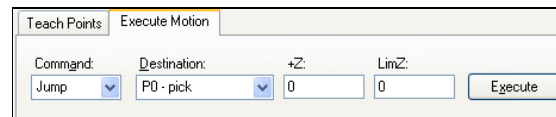
Use the **Edit** button to select and view the current point in the Points tab.

See *How to teach points* for more information.

### Execute Motion Tab

This tab executes motion commands.

Click **Execute** from this group to execute the motion.



The Execute Motion tab can be disabled from Setup | Preferences | Robot Manager| Jog & Teach.

### How to jog

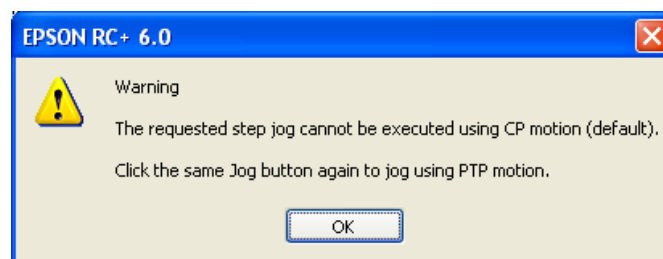
In the upper left hand corner of the Jog & Teach page, you will see a control group called Jogging that contains jog buttons. In the World, Local, Tool, and ECP jog modes, the robot is jogged in the Cartesian coordinate system (X, Y, Z). In the Joint jog mode, each robot joint can be jogged separately.

The jog speed is determined by the Speed setting. In step mode, each time you click a jog button, the robot moves along the appropriate axis by the amount specified in the **Jog Distance** control group. In continuous mode, when a jog button held down, the robot moves continuously using linear interpolated motion.



For robots other than the 6-axis robots, the jog motion in step mode is PTP (point to point) motion. It is difficult to predict exact jog motion trajectory. Therefore, be careful that the robot doesn't collide with peripheral equipment and that the robot arms don't collide with the robot itself during jogging.

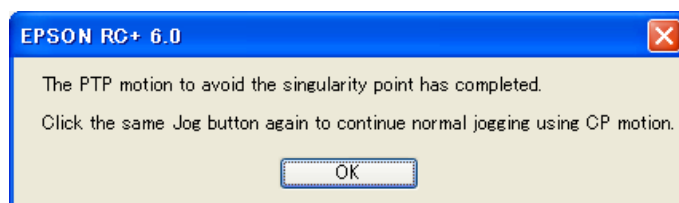
For the 6-axis robots, the jog motion in step mode is CP (Continuous Path) motion. Note that when jogging near the singularity, if you try to pass through the singularity, a warning dialog below will appear.



Click the <OK> button and click the same Jog button again to jog using PTP motion and pass the singularity.

It is difficult to predict exact jog motion trajectory in the PTP motion. Therefore, be careful that the robot doesn't collide with peripheral equipment and that the robot arms don't collide with the robot itself during jogging. Also, if you attempt the other jogs or operations, it cancels the switching to PTP motion. So when jogging near the singularity again, the same warning dialog will appear.

If passing the singularity in the continuous jog motion, the following warning message will appear.



When jogging in continuous mode, if an out of range condition occurs, the robot motors will turn off and an error will be displayed. In this case you must execute a Reset and Motor On from the Control Panel page to continue the jog.

### To jog

Select the jog mode: World, Tool, Local, Joint, or ECP.

Select the jog speed: Low or High.

Select **Continuous**, **Long**, **Medium**, or **Short** jog distance. You can type in the desired jog distance when Continuous is not selected.

Click on one of the jog buttons with the left mouse button. If you hold the mouse button down, the robot will continue to jog.

When jogging is started, the jog button picture color will change from yellow to cyan. After jogging is completed, the jog button picture color will change back to yellow.

If you click any jog button during a step jog, the robot will stop.



**TIP** You can change the orientation of the jog buttons for the robot by selecting Preferences | Robot Manager | Jogging from the Setup Menu. This will allow you to align your PC monitor with the orientation of the robot.

### Jogging in Teach Mode

You can jog and move the robot at slow speed with the safeguard open by using the Teach Pendant.

See the *RC620 Robot Controller manual: 13. Option: Teach pendant TP1*.

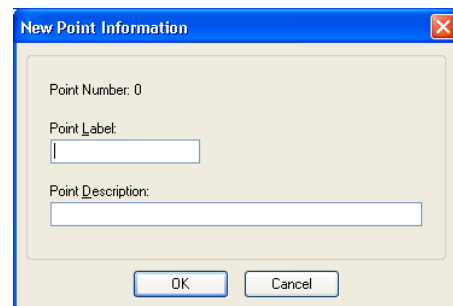
### How to teach points

Teaching a robot point means that you are defining a location for the robot to move to using the current robot position.

Follow these steps to teach points from the Robot Manager:

1. Select the point file you are teaching points for by using the [Point File] dropdown list box on the [Teach] page.
2. Select the point number you want to teach in the [Points] box.
3. Jog the robot to the desired position or free some or all axes and manually move the robot into position.
4. Click on the **Teach** button. This will retrieve the robot's current position and store it in the coordinates for the point. If the Prompt for New Point Data preference is active, you will be prompted for a point label and description.

Point labels can include up to 16 alphanumeric characters and the underscore character. Characters can be upper case or lower case. Only alphabets can be used for the first letter.



5. As an alternative to clicking the **Teach** button, on the **Points** tab you can type in the coordinates of the point.

### Saving your work

#### Robot Manager MDI Child

To save your work, use the File Menu to select Save. You can also execute Project | Save or click the Save all files toolbar button.

When you want to restore the data without saving the point files, select Restore from the File Menu.

#### Robot Manager Dialog

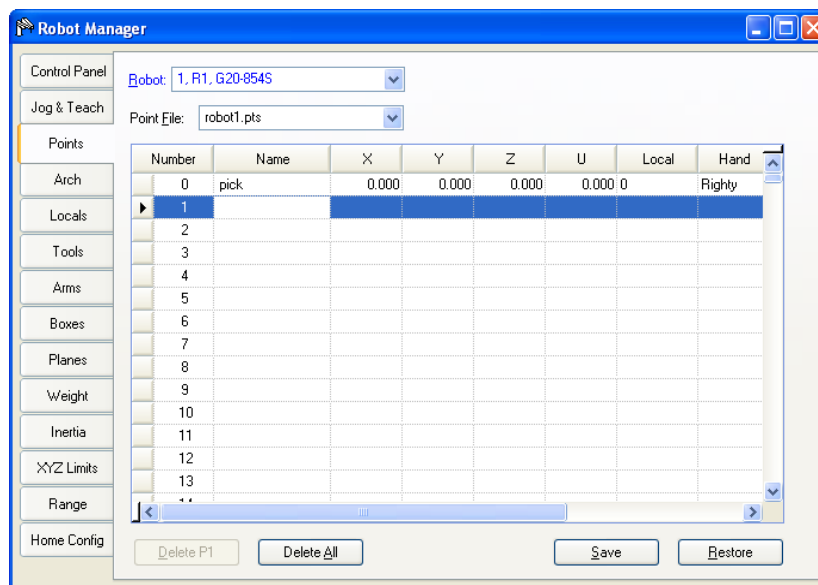
When you close the Robot Manager, you will be prompted if you want to save your changes. Answer Yes to make your changes permanent or No to cancel saving of the changes.

### Tools: Robot Manager: Points Page

You can input/delete the point data. When a point file is selected, the robot controller loads the file into memory.

As points are taught on the Robot Manager Jog & Teach page, the spreadsheet on the Points page is updated.

When the Robot Manager is used as an MDI child window, you can save the point data by typing Ctrl + S to the point file.



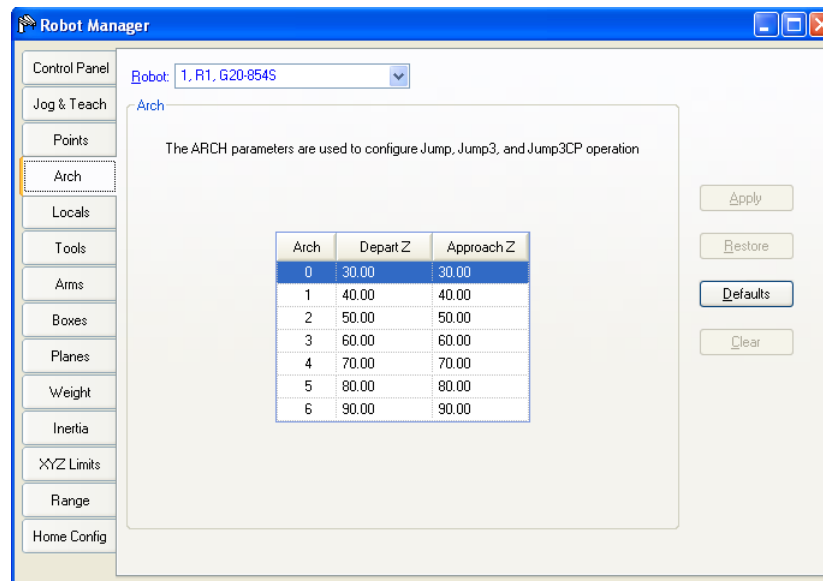
Item	Description
<b>Robot</b>	Select a robot.
<b>Point File</b>	Select a point file.
<b>Delete Pxxx</b>	Deletes the selected point. You will be prompted to confirm the operation.
<b>Delete All</b>	Deletes all points in the file. You will be prompted to confirm the operation.
<b>Save</b>	Saves the current values.
<b>Restore</b>	Reverts back to the previous values. You will be prompted to confirm the operation.

### Tools: Robot Manager: Arch Page

This page allows you to configure the depart Z and approach Z settings in the robot's Arch table. Arch is used for the Jump, Jump3, and Jump3CP motion commands.

For details on using Arch, see the *SPEL<sup>+</sup> Language Reference: Arch Statement*.

There are seven different setting pairs in the Arch table. Each setting pair is one row in the grid on the Arch page.



#### To change Arch settings

1. Put the cursor in the Depart Z or Approach Z cell of the row you want to change.
2. Type in the new value.

Press the TAB key to move to the next cell.

Item	Description
<b>Apply</b>	Set the current values.
<b>Restore</b>	Reverts back to the previous values.
<b>Defaults</b>	Click the defaults button to display factory default settings.

### Tools: Robot Manager: Locals Page

This page allows you to define local coordinate systems for a robot. When the page is selected, the current values are displayed.

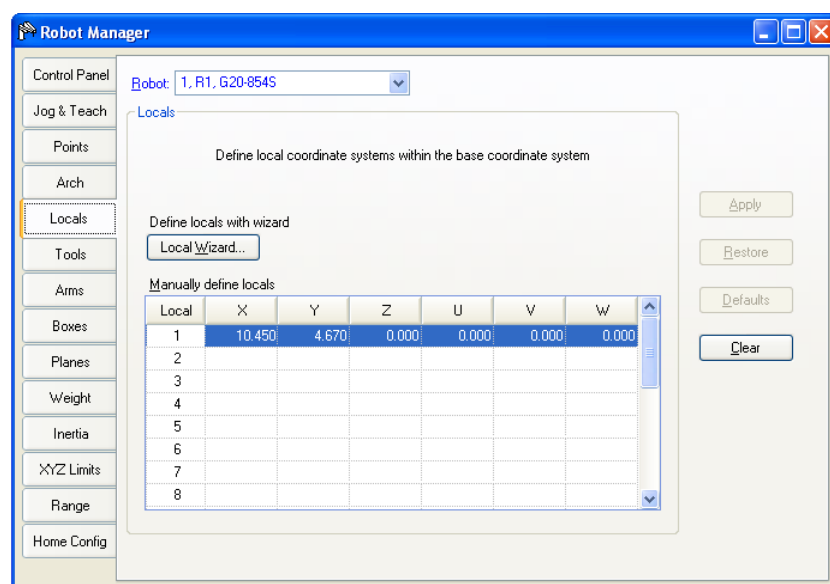
A grid is used to display all of the values for the locals you can define. Local “0” is the base coordinate system and cannot be changed from this page.



To change the base coordinate system, use the Base command from the command window. See the SPEL+ Language Reference for more information.

When a local is undefined, then all fields for that local will be blank. When you enter a value in any of the fields for an undefined local, then the remaining fields will be set to zero and the local will be defined when you click the **Apply** button.

For details on using Local, see the *SPEL+ Language Reference: Local Statement*.



#### Navigating the grid

Use the TAB key to move to the next field. Use the arrow keys or the mouse to move to any field.

Item	Description
<b>Local Wizard</b>	Click this button to start the Local Wizard. Follow the instructions for each step to define a local. See details in the next section.
<b>X</b>	The X coordinate of the local origin in the base coordinate system.
<b>Y</b>	The Y coordinate of the local origin in the base coordinate system.
<b>Z</b>	The Z coordinate of the local origin in the base coordinate system.
<b>U</b>	Rotation angle of the local about the base Z axis (roll).
<b>V</b>	Rotation angle of the local about the base Y axis (pitch).
<b>W</b>	Rotation angle of the local about the base X axis (yaw).
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to the previous values.
<b>Clear</b>	Clears all values for the selected local.

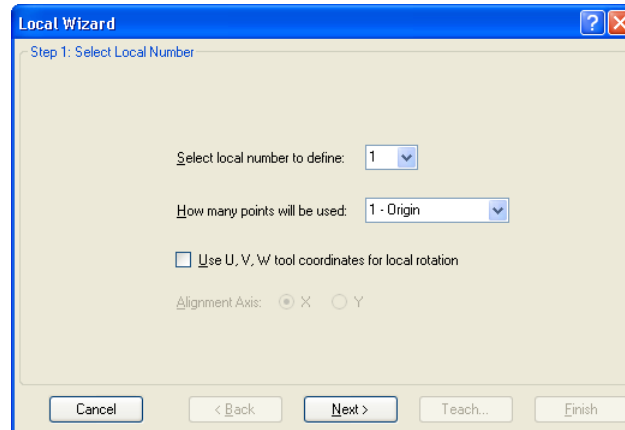


## Using the Local Wizard

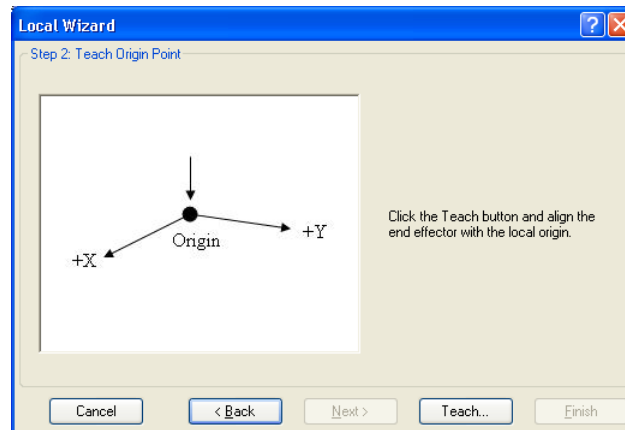
A wizard is provided for defining a local coordinate system. You can define a local using a single point or three points, as described in the following sections.

### Using the Local Wizard to teach a single point local

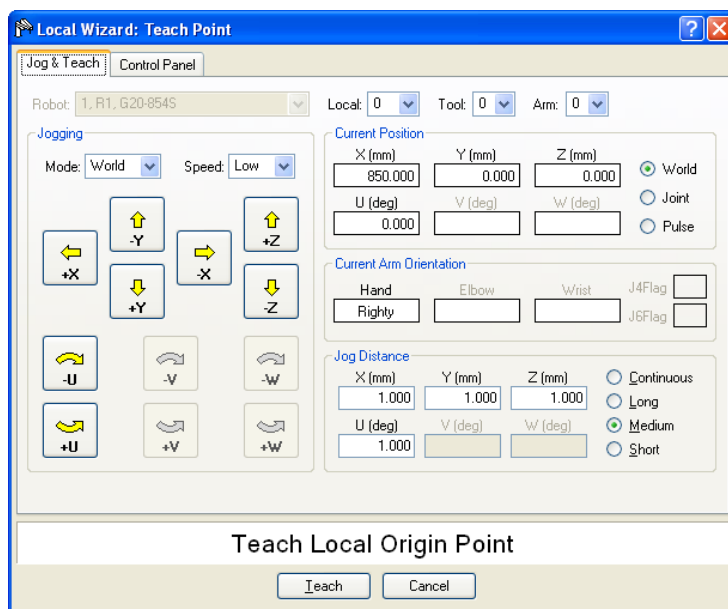
1. Open the Robot Manager and click on Locals to show the Locals page.
2. Click the **Local Wizard** button. You will see the dialog shown below.



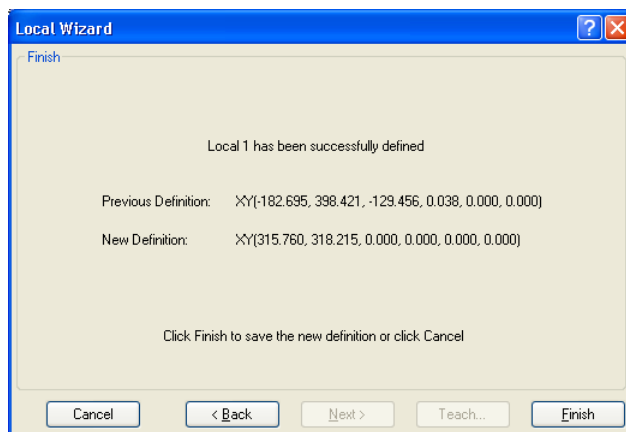
3. Select the local number you want to define. For **How many points will be used**, select **1 - Origin**. Since this is a single point local, you will just teach the origin of the new coordinate system. If you want to use the U, V, or W axes for the orientation of the coordinate system, check the **Use U, V, W tool coordinates for local rotation** checkbox. If this checkbox is unchecked, the new coordinate system is offset from local 0 in X and Y, but is not rotated about any axis. Click the **Next** button.



4. We will now teach the local origin point. Click the **Teach** button to open the Local Wizard Teach Point dialog.

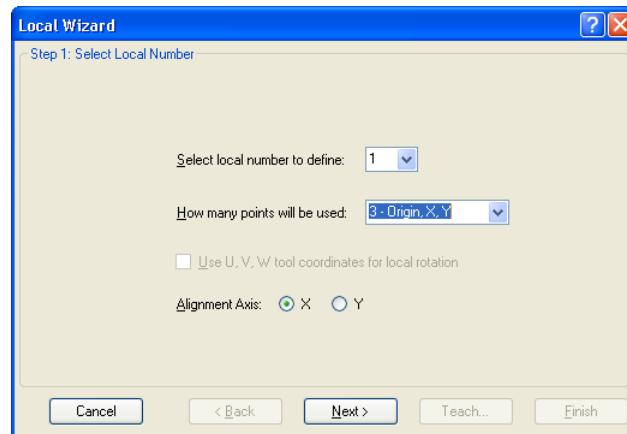


5. Jog the robot until the end effector is aligned with the local origin point. Then click the **Teach** button.
6. The new local definition is displayed as shown below. Click **Finish** to accept the new definition.

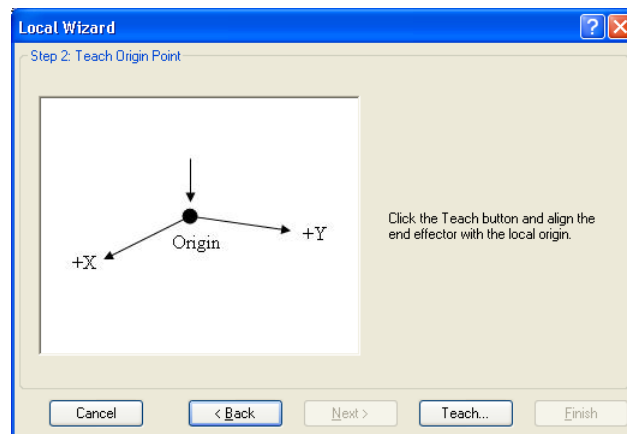


### Using the Local Wizard to teach a three point local

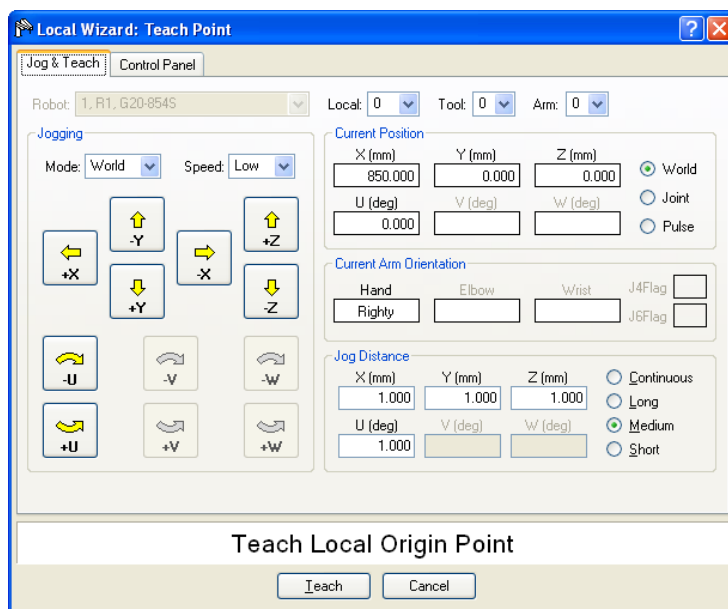
1. Open the Robot Manager and click on Locals to show the Locals page.
2. Click the **Local Wizard** button. You will see the dialog shown below.



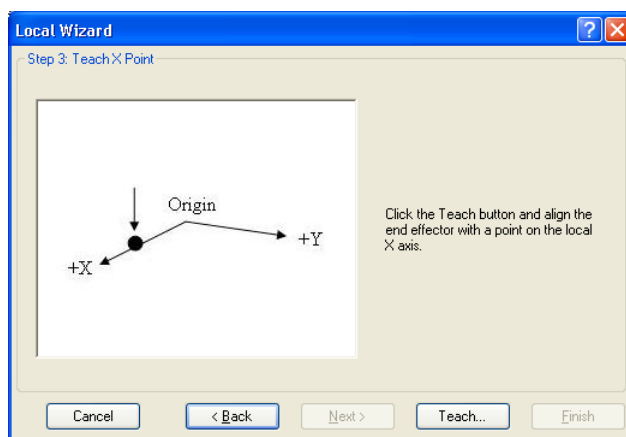
3. Select the local number you want to define. For **How many points will be used**, select **3 - Origin, X, Y**. Since this is a three point local, you will teach the origin of the new coordinate system, and then teach one point anywhere along the X axis and one point anywhere along the Y axis. Select which axis will be used to align the coordinate system. For example, if you select X, then the new coordinate system X axis will be aligned to the X axis point that you will teach in a later step. The Y axis point will be used to determine tilt. Click the **Next** button.



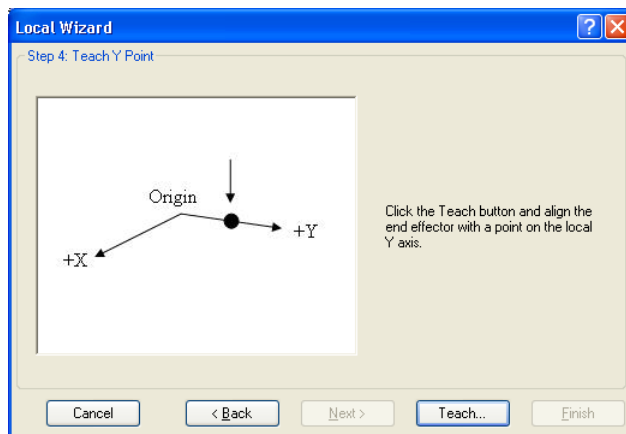
4. We will now teach the local origin point. Click the **Teach** button to open the Local Wizard Teach Point dialog.



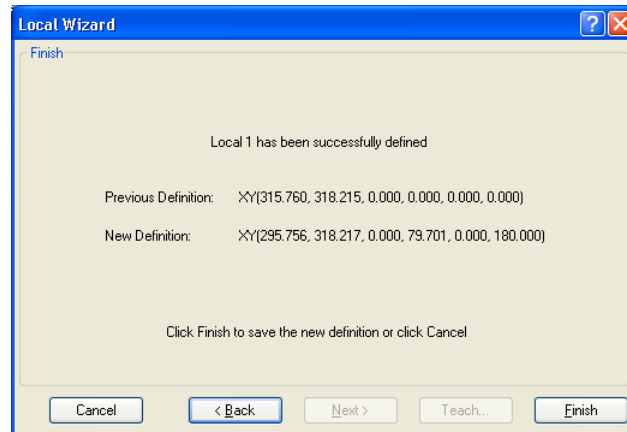
5. Jog the robot until the end effector is aligned with the origin point. Then click the **Teach** button. The next step will be displayed.



6. We will now teach a point on the local X axis. Click the **Teach** button and jog the robot until the end effector is aligned with a point anywhere along the X axis of the new coordinate system. Click the **Teach** button on the Teach Point dialog to continue.



7. We will now teach a point on the local Y axis. Click the **Teach** button and jog the robot until the end effector is aligned with a point anywhere along the Y axis of the new coordinate system. Click the **Teach** button on the Teach Point dialog to continue.
8. The new local definition is displayed as shown below. Click **Finish** to accept the new definition.



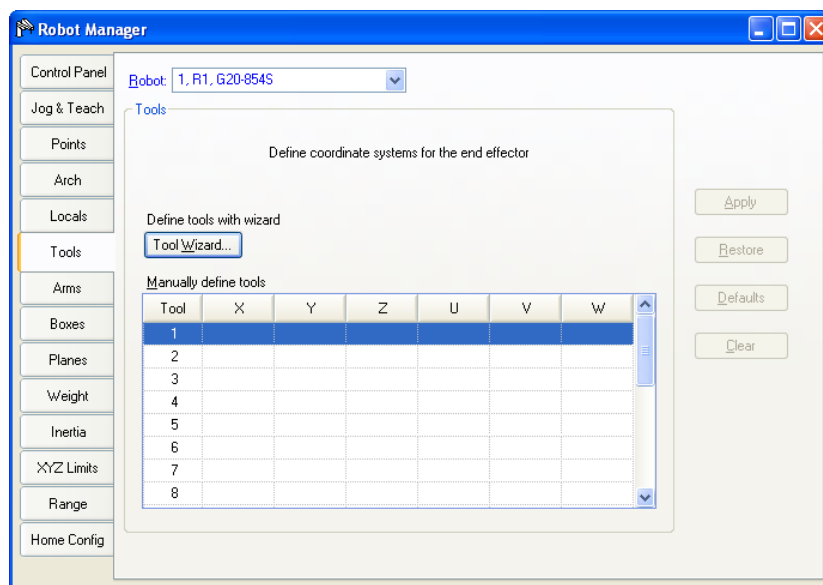
### Tools: Robot Manager: Tools Page

This page allows you to define tool settings for a robot. When the tab is selected, the current values are displayed.

A grid is used to display all the values for all 15 tools you can define.

When a tool is undefined, then all fields for that tool will be blank. When you enter a value in any of the fields for an undefined tool, then the remaining fields will be set to zero and the tool will be defined when you click the **Apply** button.

For more information on tools, see the *SPEL<sup>+</sup> Language Reference: TLSet Statement*.



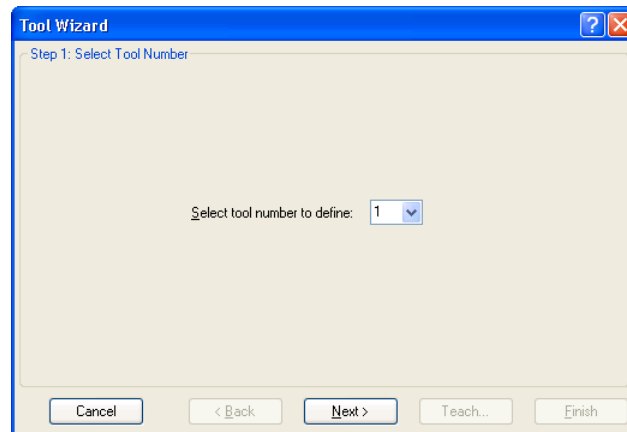
#### Navigating the grid

Use the TAB key to move to the next field. Use the arrow keys or the mouse to move to any field.

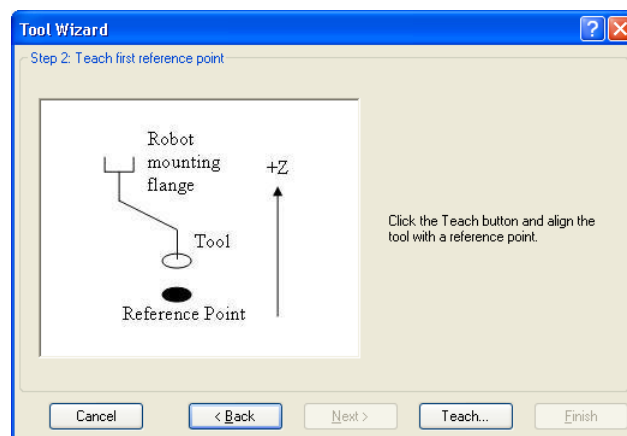
Item	Description
<b>Robot</b>	Select a robot.
<b>Tool Wizard</b>	This button starts the Tool Wizard. Follow the instructions for each step of the wizard to define a tool. See details in the next section.
<b>X</b>	The X coordinate of the tool.
<b>Y</b>	The Y coordinate of the tool.
<b>Z</b>	Z offset of tool.
<b>U</b>	Rotation angle of the tool about the Z axis (roll).
<b>V</b>	Rotation angle of the tool about the Y axis (pitch).
<b>W</b>	Rotation angle of the tool about the X axis (yaw).
<b>Apply</b>	Sets the current values.
<b>Restore</b>	Reverts back to the previous values
<b>Clear</b>	Clears all values for the selected tool.

### Using the Tool Wizard

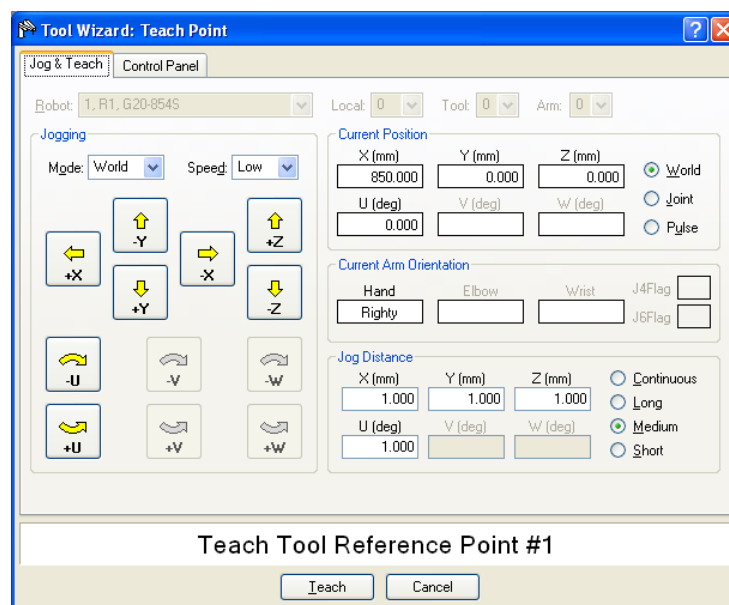
1. Open the Robot Manager and click on Tools to show the Tools page.
2. Click the **Tool Wizard** button. You will see the dialog shown below.



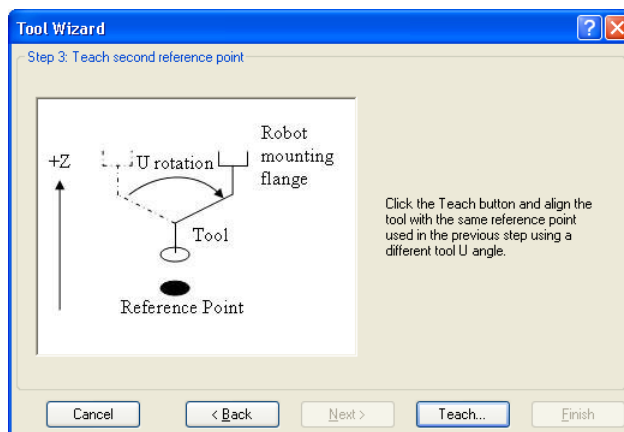
3. Select the tool you want to define and click **Next**.



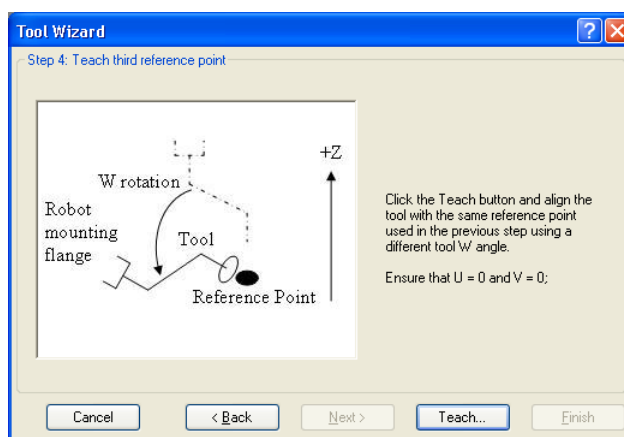
4. Click the **Teach** button to open the Tool Wizard Teach Point dialog.



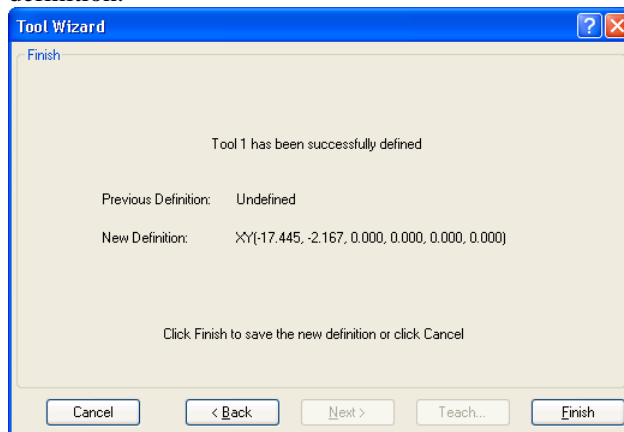
- Jog the robot until the tool is aligned with the reference point. For 6-axis robots, ensure that  $U = 0$ ,  $V = 0$ , and  $W = 180$  at the reference point. Then click the **Teach** button. The next step will be displayed in the wizard.



- Click the **Teach** button. Jog the robot so that the tool is aligned with the reference point, but from another U angle. To do this, jog the U axis several degrees, then jog X & Y until the tool is aligned with the reference point. After you have finished jogging to the reference point, click **Teach**.
- For 6-axis robots only: Jog the robot back to the reference point with  $U = 0$ ,  $V = 0$ , and  $W = 180$ . Jog the W axis several degrees, then jog X and Y until the tool is aligned with the reference point. Ensure that U and V remain at 0 degrees.



- The new tool definition is displayed as shown below. Click **Finish** to accept the new definition.





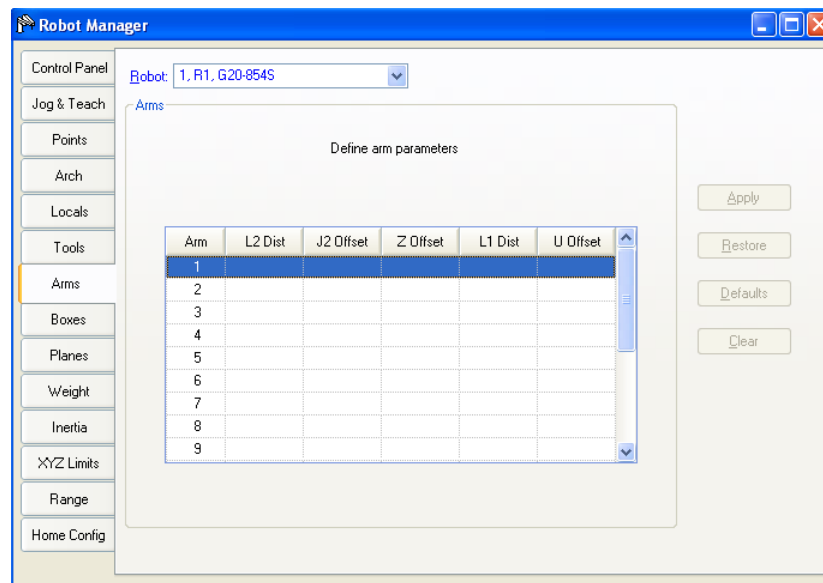
## Tools: Robot Manager: Arms Page

This page allows you to define Arm settings for a robot. When the tab is selected, the current Arm values are displayed. The tab is disabled if the current robot does not support the Arm command.

A grid is used to display all the values for all 15 arm configurations you can define.

When an arm is undefined, then all fields for that arm will be blank. When you enter a value in any of the fields for an undefined arm, then the remaining fields will be set to zero and the tool will be defined when you click the **Apply** button.

For more information on arm parameters, see the *SPEL<sup>+</sup> Language Reference: ArmSet Statement*.



### Navigating the grid

Use the TAB key to move to the next field. Use the arrow keys or the mouse to move to any field.

Item	Description
<b>Robot</b>	Select a robot.
<b>L2 Dist</b>	Distance between the center of joint 2 and the center of the orientation joint in millimeters.
<b>J2 Offset</b>	Angle of the line from the center of joint 2 to the center of the orientation joint in degrees.
<b>Z Offset</b>	The Z offset between the new orientation axis and the standard orientation axis.
<b>L1 Dist</b>	Distance between the center of the shoulder joint and the center of the elbow joint in millimeters.
<b>U Offset</b>	The angle offset between the standard orientation zero position and the new orientation axis zero position in degrees.
<b>Apply</b>	Set current values.
<b>Restore</b>	Revert to the previous values.
<b>Clear</b>	Clear all values for the selected arm

### Tools: Robot Manager: ECP Page

This page allows you to define ECP (external control point) settings for a robot. When the page is selected, the current values are displayed.

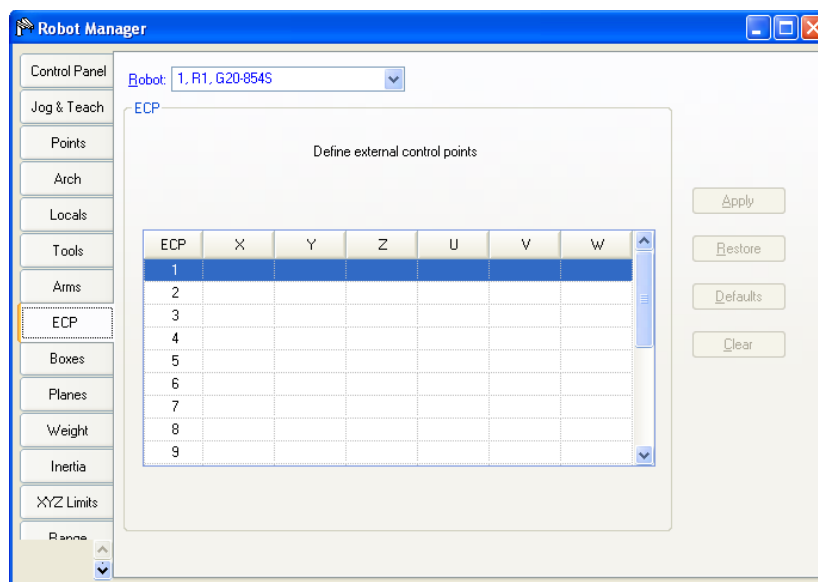


If the ECP option is not enabled in the controller, this page will not be visible.

For detailed information on using external control points in your application, refer to *6.16.5 ECP Coordinate Systems (Option)*.

A grid is used to display all of the values for all ECPs you can define.

When an ECP is undefined, then all fields for that ECP will be blank. When you enter a value in any of the fields for an undefined ECP, then the remaining fields will be set to zero and the ECP will be defined when you press the **Apply** button.



#### Navigating the grid

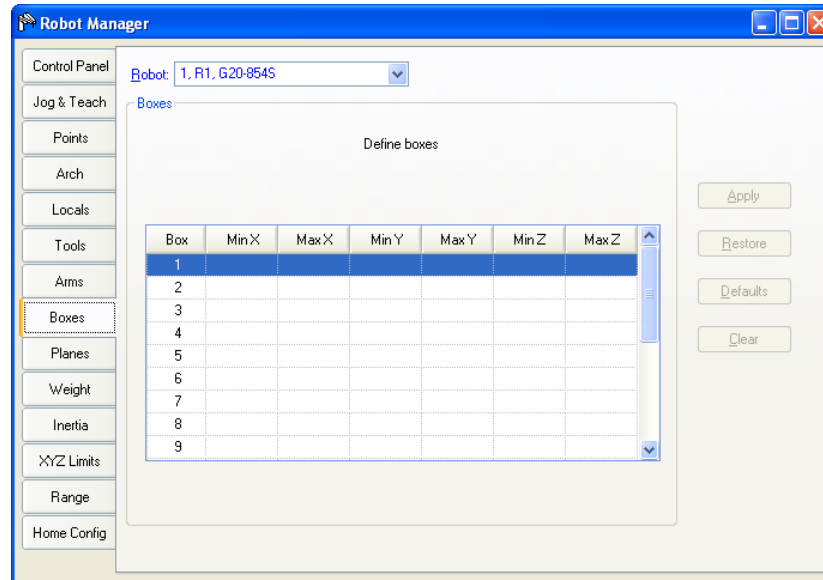
Use the TAB key to move to the next field. Use the arrow keys or the mouse to move to any field.

Item	Description
<b>Robot</b>	Select a robot.
<b>X</b>	The X coordinate of the ECP.
<b>Y</b>	The Y coordinate of the ECP.
<b>Z</b>	The Z coordinate of the ECP.
<b>U</b>	Rotation angle of the ECP about the Z axis (roll).
<b>V</b>	Rotation angle of the ECP about the Y axis (pitch).
<b>W</b>	Rotation angle of the ECP about the X axis (yaw).
<b>Apply</b>	Set current values.
<b>Restore</b>	Revert back to the previous values.
<b>Clear</b>	Clear all values for the selected ECP.

### Tools: Robot Manager: Box Page

This page allows you to define Box (approach check area) settings for a robot. When the page is selected, the current values are displayed. When a Box is undefined, then all fields for that Box will be blank. When you enter a value in any of the fields for an undefined Box, then the remaining fields will be set to zero and the Box will be defined when you press the **Apply** button.

For more information on Box, see the *SPEL<sup>+</sup> Language Reference: Box Statement*.



#### Navigating the grid

Use the TAB key to move to the next field. Use the arrow keys or the mouse to move to any field.

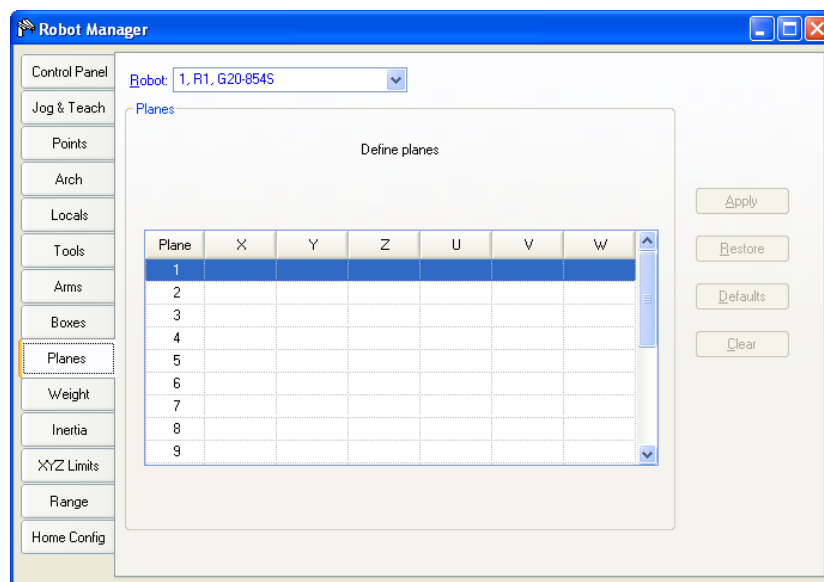
Item	Description
<b>Robot</b>	Select a robot.
<b>Min X</b>	Type in the minimum X limit value in millimeters.
<b>Max X</b>	Type in the maximum X limit value in millimeters.
<b>Min Y</b>	Type in the minimum Y limit value in millimeters.
<b>Max Y</b>	Type in the maximum Y limit value in millimeters.
<b>Min Z</b>	Type in the minimum Z limit value in millimeters.
<b>Max Z</b>	Type in the maximum Z limit value in millimeters.
<b>Apply</b>	Sets current values.
<b>Restore</b>	Reverts back to the previous values.
<b>Clear</b>	Clears all values.

Setting both values to zero disables the limits.

### Tools: Robot Manager: Plane Page

This page allows you to define Plane (approach check plane) settings for a robot. When the page is selected, the current values are displayed. When a Plane is undefined, then all fields for that Plane will be blank. When you enter a value in any of the fields for an undefined Plane, then the remaining fields will be set to zero and the Plane will be defined when you press the **Apply** button.

For more information on Plane, see the *SPEL<sup>+</sup> Language Reference: Plane Statement*.



#### Navigating the grid

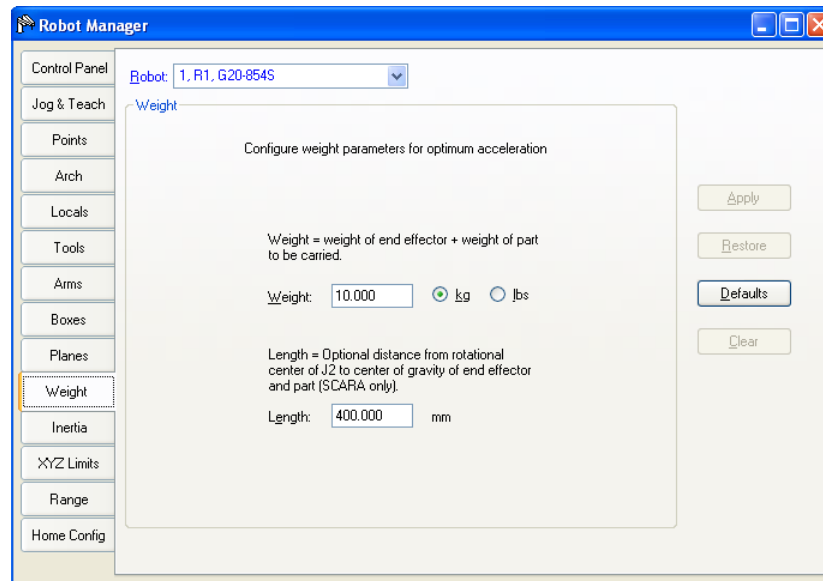
Use the TAB key to move to the next field. Use the arrow keys or the mouse to move to any field.

Item	Description
<b>Robot</b>	Select a robot.
<b>X</b>	Sets the X origin of the coordinate for approach check plane.
<b>Y</b>	Sets the Y origin of the coordinate for approach check plane.
<b>Z</b>	Sets the Z origin of the coordinate for approach check plane.
<b>U</b>	Sets the U origin of the coordinate for approach check plane.
<b>V</b>	Sets the V origin of the coordinate for approach check plane.
<b>W</b>	Sets the W origin of the coordinate for approach check plane.
<b>Apply</b>	Set current values.
<b>Restore</b>	Revert back to the previous values.
<b>Clear</b>	Clear all values.

## Tools: Robot Manager: Weight Page

This page is for changing the Weight parameters for the robot.

For details on the Weight parameters, see the *SPEL<sup>+</sup> Language Reference: Wight Statement*.

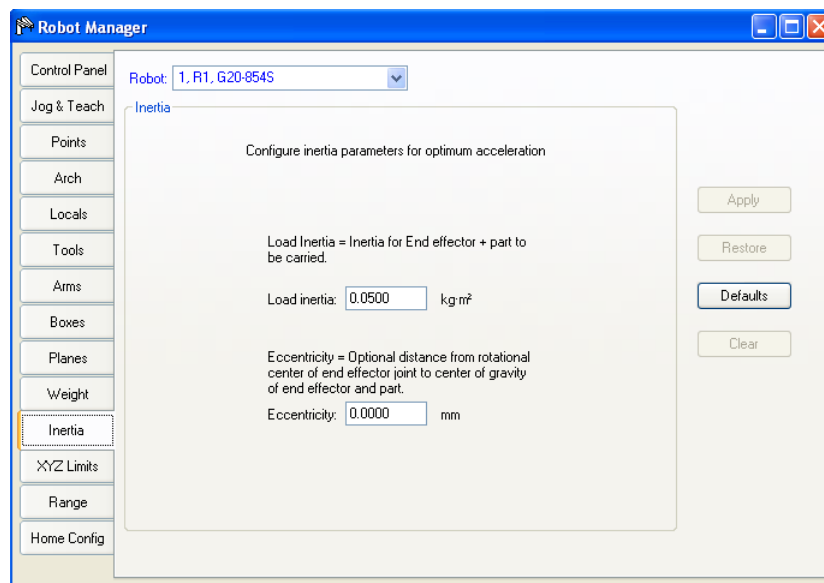


Item	Description
<b>Robot</b>	Select a robot.
<b>Weight</b>	Type in the new total weight of the payload on the robot.
<b>Kg/Lb</b>	Choose which unit the weight is represented in: kilograms or pounds.
<b>Length</b>	Type in the new length.
<b>Apply</b>	Sets the current values.
<b>Restore</b>	Reverts back to the previous values.
<b>Defaults</b>	Displays factory default settings.

## Tools: Robot Manager: Inertia Page

This page is for changing the Inertia parameters.

For details on the Inertia parameters, see the *SPEL<sup>+</sup> Language Reference: Inertia Statement*.

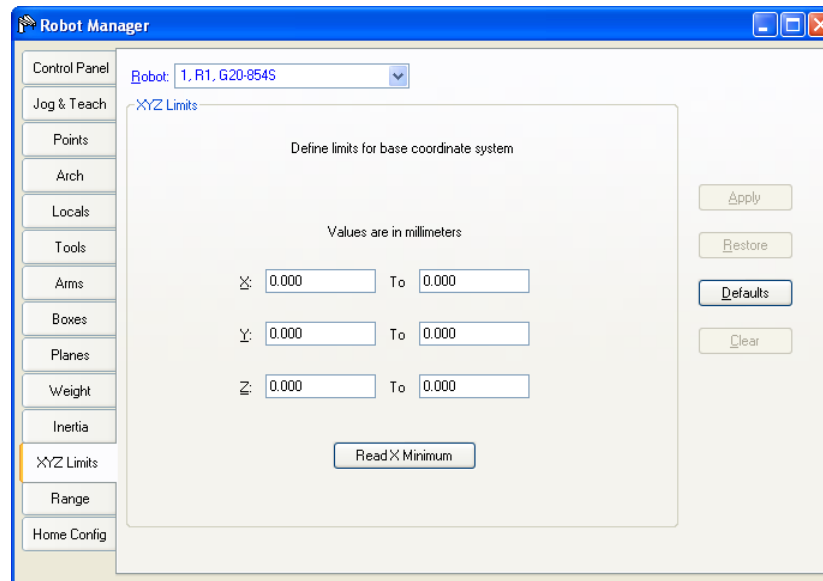


Item	Description
<b>Robot</b>	Select a robot.
<b>Load inertia</b>	Type in the new load inertia of the payload on the robot in kg·m <sup>2</sup> . This includes the inertia of end effector plus the part to be carried.
<b>Eccentricity</b>	Type in the new eccentricity value in millimeters. This is the distance from rotational center of joint 4 to the center of gravity of end effector and part.
<b>Apply</b>	Set the current values.
<b>Restore</b>	Revert back to the previous values.
<b>Defaults</b>	Press the defaults button to display factory default settings.

### Tools: Robot Manager: XYZ Limits Page

This page allows you to configure limits for X, Y and Z motion in the robot envelope.

For details on the XYZ limits, see the *SPEL<sup>+</sup> Language Reference: XYLim Statement*.



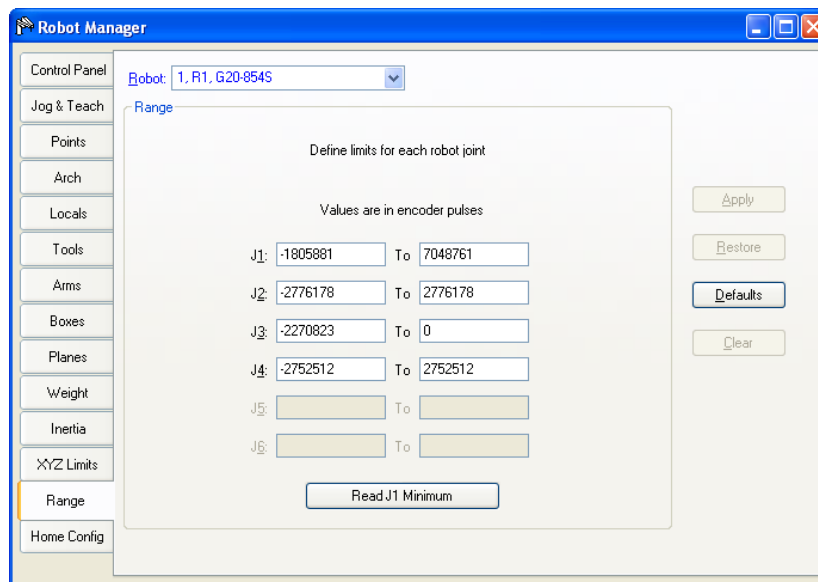
Item	Description
<b>Robot</b>	Select a robot.
<b>X, Y, Z</b>	Type in the minimum and maximum X, Y, and Z limit values. Setting both values to zero disables the limits.
<b>Read Current</b>	Click this button to read the value from the current robot position. The button text shows the axis and minimum or maximum depending on which text field has the current focus.
<b>Apply</b>	Set the current values.
<b>Restore</b>	Revert back to previous values.
<b>Defaults</b>	Set default values.

### Tools: Robot Manager: Range Page

This page allows you to configure the robot joint software limits.

For more information on Range, see the SPEL+ Language Reference and the manual for the robot you are using.

For details on configuring the motion range, see the *SPEL+ Language Reference: Range Statement*.



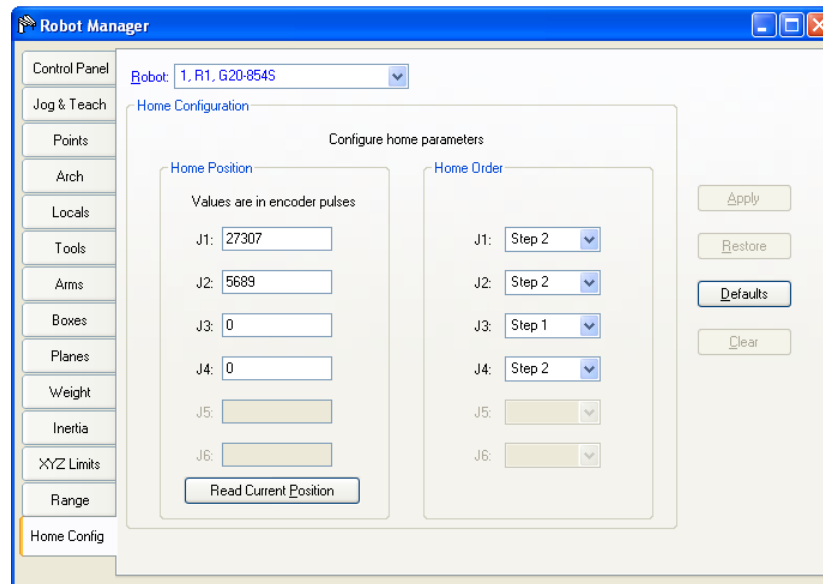
Item	Description
<b>Robot</b>	Select a robot.
<b>J1 - J6</b>	Type in the minimum and maximum encoder pulse values for each joint.
<b>Read Current</b>	Click this button to read the current joint value of the robot into the currently field. The button text will change depending on which text field has focus.
<b>Apply</b>	Save the current changes.
<b>Restore</b>	Revert back to the previous values.
<b>Defaults</b>	Set the default values.



## Tools: Robot Manager: Home Config Page

Home Config allows you to configure the optional user home position.

For details on configuring the home position, see the *SPEL<sup>+</sup> Language Reference: HomeSet Statement*.



### Changing home position

When you select the Home Config tab, the current home position is read from the robot controller and displayed in the text boxes. If the home position has never been defined, then the text boxes will be blank.

To define the home position, you can enter an encoder position value for each of the four robot joints in the text boxes, or you can select the Jog & Teach page to jog the robot to the desired home position, then select the Home Config page and click the **Read Current Position** button to read the current encoder position values.

### Changing home order

The home command executes in steps. The number of steps equals the number of joints on the robot. Select the home step number for each joint using the dropdown list for each joint. More than one joint can be homed in the same step.

### Testing home

After making changes to the home position and home order, you can click the Robot Manager **Control Panel** tab and click the **Home** button.

Item	Description
<b>Robot</b>	Select a robot.
<b>Read Current</b>	Click this button to read the current position encoder pulse value into the currently selected text field. The button text will change according which text field is selected.
<b>Defaults</b>	Set the value of the [Home order] group box to the default value.
<b>Apply</b>	Save the current changes.
<b>Restore</b>	Revert back to the previous values.

### 5.11.2 Command Window Command (Tools Menu)

You can execute SPEL<sup>+</sup> commands from the robot controller and view the results.

#### To open the Command window

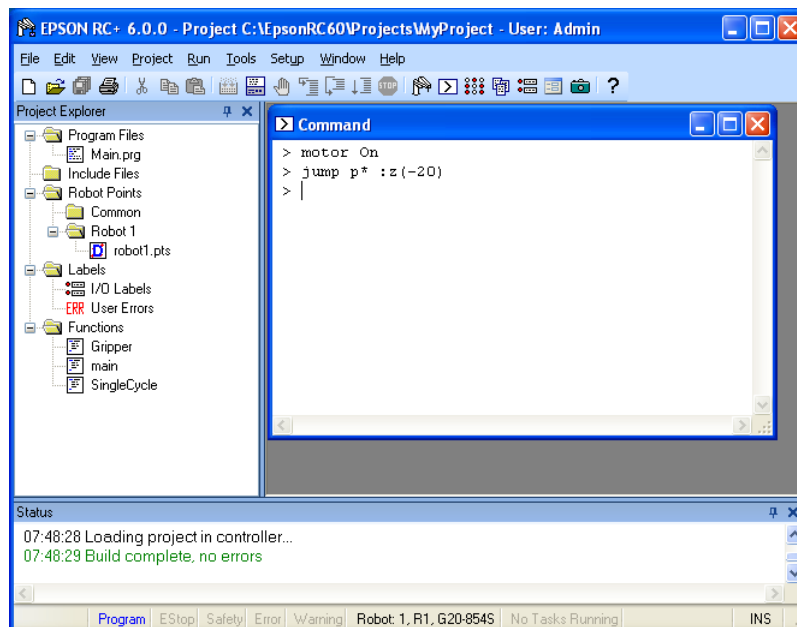
Select Command Window from the Tools Menu

Or

Click on the  button on the toolbar.

Or

Type Ctrl+M



#### To execute SPEL+ commands from the Command window

1. Type in the desired command after the prompt (>). Commands can be entered in upper or lower case.
2. Press the Enter key to execute the command. The cursor can be anywhere on the line when you press Enter.
3. Wait for the prompt to return before typing in a new command.

When an error occurs, an error number will be displayed along with an error message.

You can use the arrow keys or the mouse to move the cursor to any line in the window that starts with a prompt (>) character and execute it by pressing Enter.

#### Command Window Keys

Key	Action
<b>Ctrl+A</b>	Select entire window.
<b>Ctrl+C</b>	Stop the program and initialize robot controller. If a robot motion command is in progress, the prompt will return when the command has been completed.

Key	Action
<b>Ctrl+V</b>	Execute Paste command. Paste from Clipboard to current selection.
<b>Ctrl+W</b>	Re-display last command line after the prompt.
<b>Ctrl+X</b>	Execute Cut command. Cut current selection and put in Clipboard.
<b>Ctrl+Z</b>	Undo last change.
<b>Ctrl+Home</b>	Go to the top of the window.
<b>Ctrl+End</b>	Go to last prompt at end of the window.
<b>?</b>	Translates to "PRINT " when used as the first character of a command. This can be used to display variables or any statement that requires a PRINT command.

### 5.11.3 I/O Monitor Command (Tools Menu)

The I/O Monitor window lets you monitor all controller hardware inputs and outputs and also memory I/O. There are up to four views available: one standard view and three custom views.

On the standard view, there are two grids. For each grid you can specify which type and size of I/O to monitor.

For each custom view, you can specify a list of any combination of input, output, or memory. By default, there is one custom view available. To use the other two custom views, right click on a tab and check the views you want to be visible. See the section *Custom I/O Views* later in this chapter.

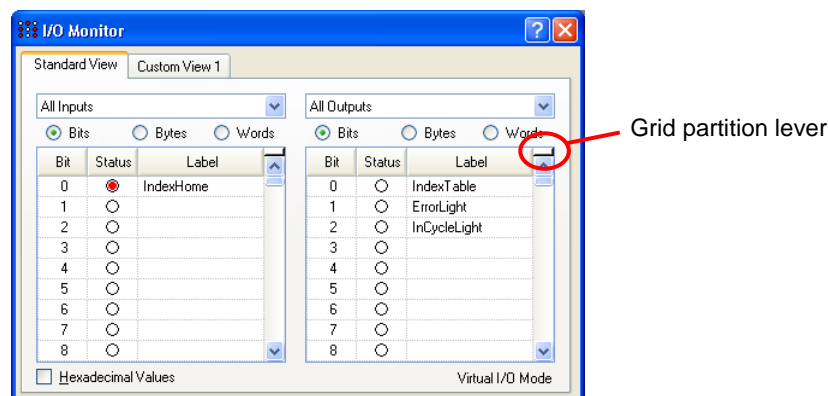
Labels that have been defined using the I/O Label Editor are displayed next to each bit, byte, or word.

After the I/O Monitor window has been opened, the input and output status for the current view is constantly updated.

The I/O monitor will always be displayed on top of other child windows, such as program windows and point windows.

If a description has been entered for an I/O port (bit, byte, or word) in the I/O Label Editor, then a tool tip will be displayed when the mouse pointer is over the row containing the port.


You can turn outputs on and off by double clicking on the output LED images in the Status column.



**To open the I/O Monitor**

Select I/O Monitor from the Tools Menu.

Or

Click on the  tool bar button.

Or

Type Ctrl + I.

**Using the I/O Monitor**

Select the **Standard View** tab.

Scroll through the grids to locate the desired inputs or outputs to monitor.

You can split each grid into two scroll regions by selecting the split bar in the upper right corner of the grid and dragging it down. Each scroll region can be individually scrolled.

To turn an output off or on, double click on the LED image for the desired output.

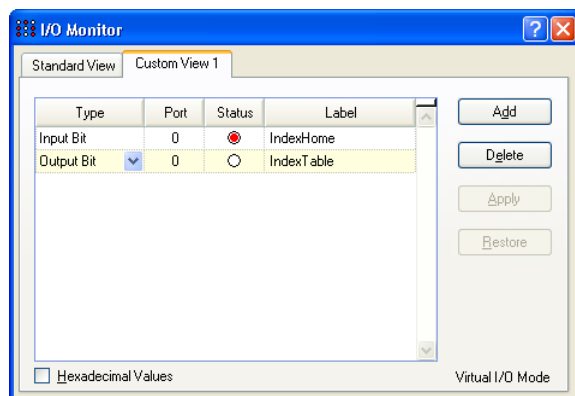
When Virtual I/O is active, you can turn input bits on and off by double clicking on the input LED images in the Status column.

To view bytes and words in hexadecimal format, check the **Hexadecimal Values** checkbox.

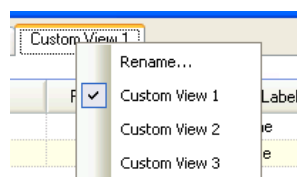
You can resize the I/O monitor in the vertical direction to show more data. Move your mouse pointer to the lower right corner of the window to activate a size handle, then click down and drag the window down or up to the desired size.

**Custom I/O Views**

You can configure up to three custom I/O views. In each view, you can add any combination of I/O. You can also change the name of each view and hide each view.

**To change a view**

1. Click on a custom view tab. If none are currently shown, right click on the **Standard View** tab and select one of the three custom views to show it.



2. Click the **Add** button to add a new row to the list.
3. Select the **Type** by clicking in the **Type** column, then click the arrow to view a list of I/O types and select one.

4. In the **Port** column, select the port (bit, byte, or word, depending on I/O type).
5. Add more rows as needed by repeating steps 2 - 4.
6. Save the changes by clicking the **Apply** button.
7. Click the **Delete** button to delete a row.
8. Use the **Restore** button to cancel changes.

#### To rename a view

1. Click on a custom view tab. If none are currently shown, right click on the **Standard View** tab and select one of the three custom views to show it.
2. Right click on the view tab and select **Rename**.
3. Enter the new name for the view.

#### 5.11.4 Task Manager Command (Tools Menu)

The Task Manager window allows you to Halt (suspend), Resume (continue), and Quit (abort) tasks.

To open the Task Manager

Select Task Manager from the Tools Menu.

Or

Type Ctrl + T.

Or

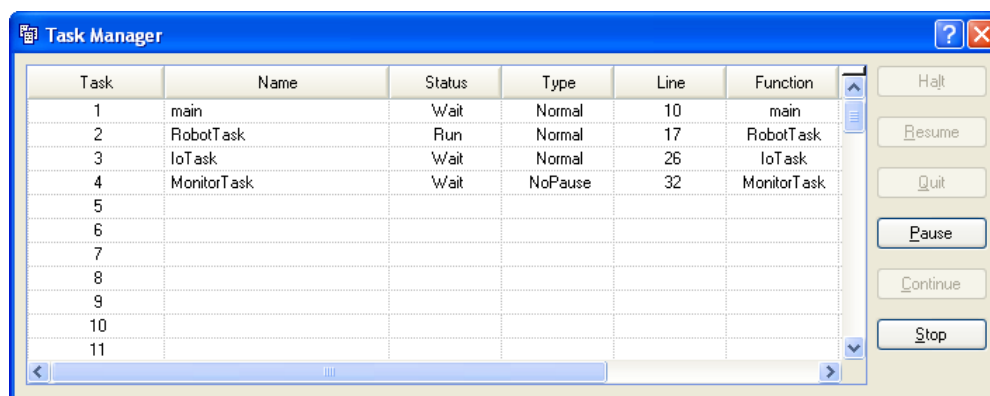
Click on the  button on the toolbar.

#### Operation

The Task Manager is used for suspending, resuming, stepping, and stopping tasks.

When Task Manager is started, you will see a grid containing status information for 32 tasks standard tasks and 11 trap tasks. Also, you will see the status information of 16 background tasks if the background task is enabled. There are 8 items shown for each task. To view all of the columns, use the scroll bar or resize the window.

<b>Task</b>	Number of task from 1 to 32 and 11 trap tasks.	
<b>Name</b>	Name of the function that was started as a task.	
<b>Status</b>	Current task status: Run, Wait, Halt, Pause, Aborted, Finished.	
<b>Type</b>	Task types	
	Normal	This task is a normal task
	NoPause	This task does not pause with Pause statement or when Pause input or Safety Door open occur.
	NoEmgAbort	This task continuously processes during the Emergency Stop or error occurrence.
<b>Line</b>	Current task line number.	
<b>Function</b>	Current task function name.	
<b>Program</b>	Current task program name.	
<b>Start</b>	The date and time that the task was started.	



Item	Description
<b>Halt</b>	<p>Suspends the selected task. This temporarily stops the task so that it can continue with the <b>Resume</b> button. <b>Halt</b> can only be executed when a task is running (status is Run). When <b>Halt</b> is executed, the <b>Resume</b> button will be enabled. If a motion command is executing with Halt occurs, the motion will be completed before the task reaches the Halt state.</p> <p>The task also temporarily stops when the task is NoPause type or NoEmgAbort type.</p>
<b>Resume</b>	<p>After one or more tasks have been suspended with the <b>Halt</b> button, clicking on <b>Resume</b> will make the halted tasks continue where they left off. First, a confirmation dialog is displayed.</p>
<b>Quit</b>	<p>This button stops the selected task permanently. You cannot <b>Resume</b> a task once you have executed <b>Quit</b>. To restart the task, you must start it from within a program or from the Run window. The task also stops when the task is NoPause type or NoEmgAbort type.</p>
<b>Pause</b>	<p>This button pauses tasks that can be paused. After pause, you must use either <b>Continue</b> or <b>Stop</b>.</p> <p>The task does not pause when the task is NoPause type or NoEmgAbort type.</p>
<b>Continue</b>	<p>This button continues all tasks that were paused with the <b>Pause</b> button.</p>
<b>Stop</b>	<p>This button stops all tasks.</p>

To Halt, Step, Walk, and Resume a task

The **Halt** button will become active after you select a running task.

Click the **Halt** button to stop the task you selected for a moment.

After a task has been halted, the source code will be displayed and the next step will be indicated. You can click on the **Resume** button to resume execution. (You can also execute Step Into, Step Over, or Walk from the Run Menu.)

### To Pause and Continue tasks

Pause allows you to "suspend" all tasks that can be suspended.

Click on the **Pause** button to pause available tasks. The robot will decelerate to a stop immediately.

After executing Pause, click on **Continue** to resume all suspended tasks.

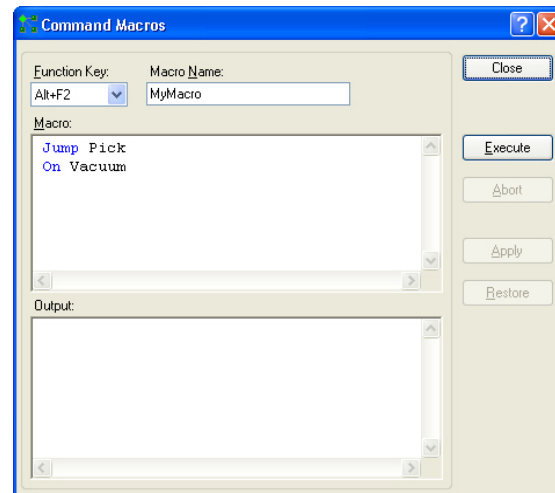
### To view source code at the current execution line

Select a task row. Then right click and select Go To Line. The program editor will be opened at the current execution line

#### 5.11.5 Macros Command (Tools Menu)

You can create SPEL+ command macros using the Macro Editor. Macros consist of one or more SPEL+ statements that can be executed from the command window. A macro statement may use global variables, I/O labels, and point labels. You can assign a macro to each of the Alt function keys except for Alt+F4, which is a Windows shortcut to close the application.

1. Select Tools | Macros to open the Command Macros dialog.



2. Type the macro statements in the **Macro** text box.
3. Click the **Apply** button to save changes.
4. Click **Execute** to run the macro.
5. Click **Close** to close the dialog. You will be prompted to save the macros you have created or changed.

To open a macro and execute it, type Alt + function key. Then click **Execute** to run it. Macros never execute by pressing the function key. The separate execute step is provided for safety, since macros can move the robot and control I/O.

Macros can be executed while tasks are running. If invalid commands are executed while tasks are running, an error will occur.

### 5.11.6 I/O Label Editor Command (Tools Menu)

The I/O label editor lets you define meaningful names for inputs, outputs, and memory I/O for each project. The labels can be used in your programs, from the Command window, or in macros. They are also displayed in the I/O Monitor window.

To open the I/O Label Editor

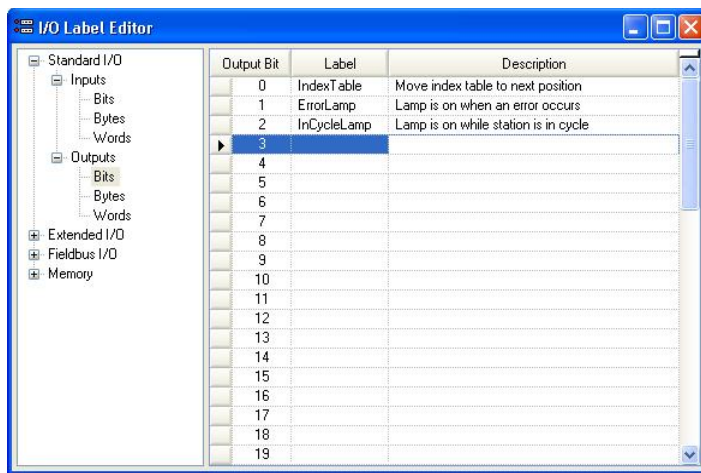
Select I/O Label Editor from the Tools Menu.

Or

Type Ctrl + L.

Or

Click on the  button on the toolbar.



#### The I/O Label Spreadsheet

When you select I/O Label Editor from the Tools Menu, a window opens that contains a tree and a spreadsheet editor.

The tree on the left side of the window shows the various types of I/O for the controller. For each type of I/O you can view and edit labels for bits, bytes (8 bits), and words (16 bits).

The first column of the spreadsheet shows the bit, byte, or word number, depending on which type of I/O you are viewing.

The second column contains the label for each bit, byte, or word in column 1. You can type in up to 16 characters for a label. Label characters can be alphanumeric or underscore.

The third column contains the description associated with the label.

If you add a description to an I/O point, then the description will be displayed as a tool tip on the I/O Monitor.




- The I/O Label Editor shows all available I/O types on your controller.
- For the Editor version, the I/O Label Editor shows the all I/O types. For example, you can edit Fieldbus I/O labels, but you may not have a Fieldbus board installed in the controller.



**To add or edit a label**

Select the type of I/O you want to label in the tree. After you select the I/O type, the spreadsheet will be refreshed to display the labels for that type. The number of rows in the spreadsheet equals the number of bits, bytes, or words available for the type you have selected.

Use the mouse to scroll through the spreadsheet and put the cursor in the Label field next to the bit, bytes, or words number that you want to add a label to. Type in the label, which can be up to 16 alphanumeric characters without any spaces. Optionally, you can type a description for the label in the Description field.

After adding or editing labels, save the changes by executing Save from the File Menu or by clicking on the Save Project  toolbar button. If any duplicate labels are detected, an error message will be displayed and the save operation will be aborted. You must correct the duplication before you can save the labels successfully.

**To cut and paste labels and descriptions**

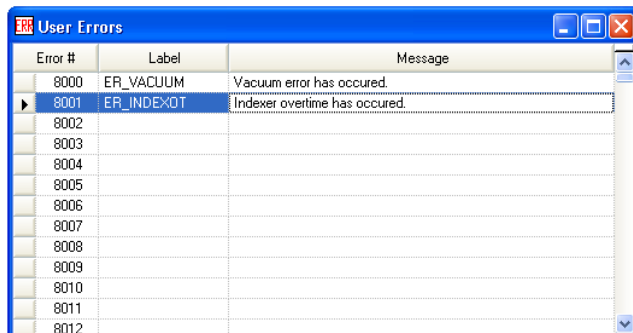
You can cut and paste labels and descriptions by selecting with the mouse, then executing Copy, Cut, and Paste from the Edit Menu.

You can also cut and paste entire rows using the following steps:

1. Select one or more rows by using the row selectors on the left and execute either the Cut or Copy command from the Edit Menu. When selecting multiple rows, hold down the shift or control key while selecting rows with the mouse.
2. Select the row where you want to start the paste by clicking the row selector on the left of the row.
3. Execute the Paste command from the Edit Menu.

### 5.11.7 User Error Editor Command (Tools Menu)

The User Error Editor allows you to define user errors.



User error numbers can be from 8000 to 8999.

Labels can be up to 16 characters in length.

It is recommended that you use the ER\_ prefix for each error label and use all caps for the label. This makes it easy to see error labels in your code.

Some user error examples:

Error #	Label	Message
---------	-------	---------

8000	ER_VACUUM	Vacuum error has occurred.
------	-----------	----------------------------


8001	ER_INDEXOT	Indexer overtime has occurred.
------	------------	--------------------------------

In your program code, use the Error statement to generate a user error. For example:

```
On Vacuum
Wait Sw(VacOn), 1
If Tw = 1 Then
    Error ER_VACUUM
EndIf
```

The user error information is stored in the current project directory in a file called UserErrors.dat.

You can use the Import command from the File Menu to import user errors from other projects.

After adding new error definitions, save the changes by executing Save from the File Menu or by clicking on the Save Project  toolbar button. If any duplicate labels are detected, an error message will be displayed and the save operation will be aborted. You must correct the duplication before you can save the labels successfully.

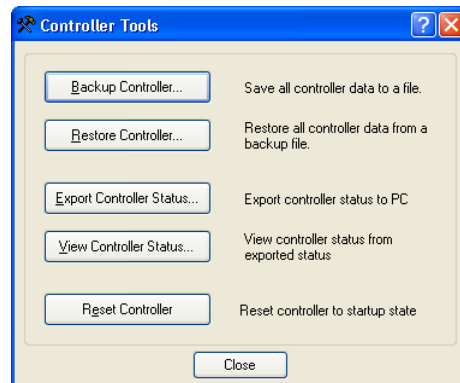
### 5.11.8 Controller Command (Tools Menu)

Select Controller from the Tools Menu to open the Controller Tools dialog.

From the Controller Tools dialog, you can save and restore the complete controller configuration and the project using the Backup Controller and Restore Controller commands. You can also save and view controller status, and reset the controller.

Before servicing the system, you should execute Backup Controller and store the system configuration on an external media such as a USB memory key.

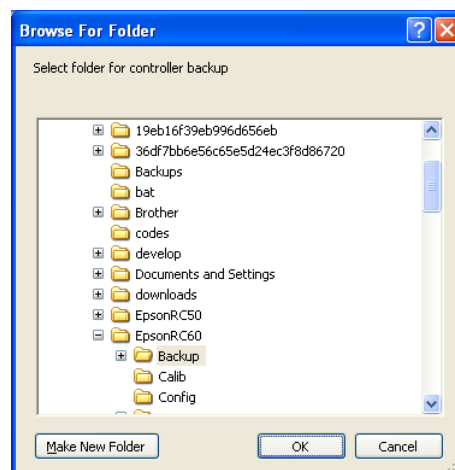
If required, you can use Restore Controller to restore previously stored data.



#### Backup Controller

Use Backup Controller to save controller configuration data on your PC. You cannot backup the controller data while tasks are running. If you attempt to do so, an error message will be displayed.

1. Select Tools | Controller.
2. Click on the **Backup Controller** button to open the **Browse For Folder** dialog.



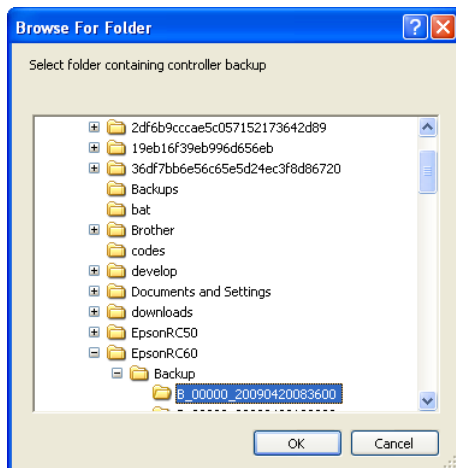
3. Select the disk drive and parent folder where you want to save the information. You can create a new parent folder by clicking the **Make New Folder** button.
4. Click **OK**. A new folder containing the backup files will be created in the selected folder named "B\_" plus the controller serial number followed by the current date / time.

## Restore Controller

Use Restore Controller to load controller settings from previously saved backup data. You cannot restore the controller data while tasks are running. If you attempt to do so, an error message will be displayed.

To restore controller configuration:

1. Select Tools | Controller.
2. Click on the **Restore Controller** button to open the **Browse For Folder** dialog.



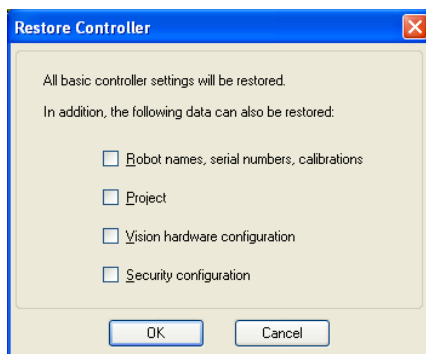
3. Select the drive and folder where the information is stored. Controller backup information is stored in a folder that is named “B” plus the controller serial number followed by the date / time.



You can also select a folder containing export controller status information.

Export controller status is stored in a folder that is named “S” plus the controller serial number followed by the date / time.

4. Click **OK** to display the dialog to select the restore data.



### Robot names, serial numbers, calibrations checkbox

This checkbox allows you to restore the robot name, robot serial number, Hofs data, and CalPIs data. Make sure that the correct Hofs data is restored. If the wrong Hofs data is restored, the robot may move to wrong positions.

The default setting is unchecked.

### Project checkbox

This checkbox allows you to restore the files related to projects.

The default setting is unchecked.

When the project is restored, all the values of Global Preserve variables are restored.

For details about Global Preserve variable backup, refer to *5.10.10 Display Variables Command (Run Menu)*.

**Vision hardware configuration checkbox**

This checkbox allows you to restore the vision hardware configuration.

For details, refer to the *EPSON RC+ 6.0 option: Vision Guide 6.0*.

This is not checked by the default setting.

**Security configuration checkbox**

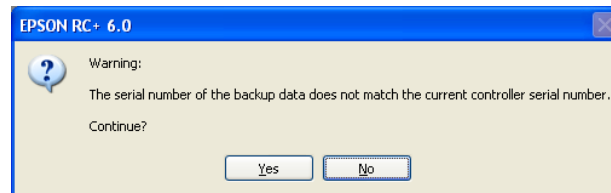
This checkbox allows you to restore the security configuration.

For details, refer to *14. Security*.

This is not checked by the default setting.

5. Click the <OK> button to restore the system information.

Restore the system configuration saved using Backup Controller only for the same system.  
When different system information is restored, the following warning message appears.



Click the **No** button to cancel restoration of data except for special situations such as controller replacement.

**Export Controller Status**

You can export the current status of the controller using this button. Tasks can be running when this command is executed. The current status is saved in a folder containing several files. The controller configuration settings, task status, I/O status, robot status, etc. are saved in these files. This is useful for an end user to send a snapshot of the controller status to a system vendor or to Epson technical support, should the need arise.



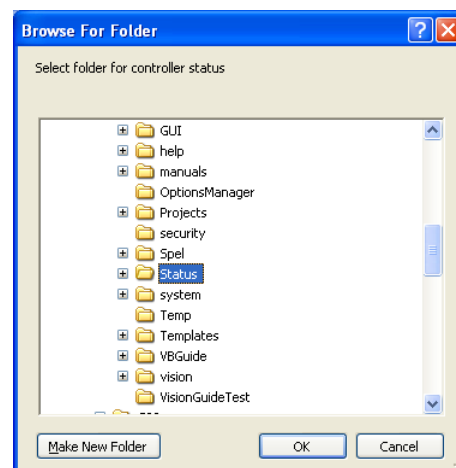
Export Controller Status is equivalent to connecting a USB memory key to the controller and pressing the TRIG button on the controller to save controller status.

You can configure the controller whether to save the project files in the status folder or not. See Setup | System Configuration | Controller | Preferences.

You can view the exported status using the **View Controller Status** button (see next section).

**To export controller status**

1. Select Tools | Controller.
2. Click on the **Export Controller Status** button to open the **Browse For Folder** dialog.



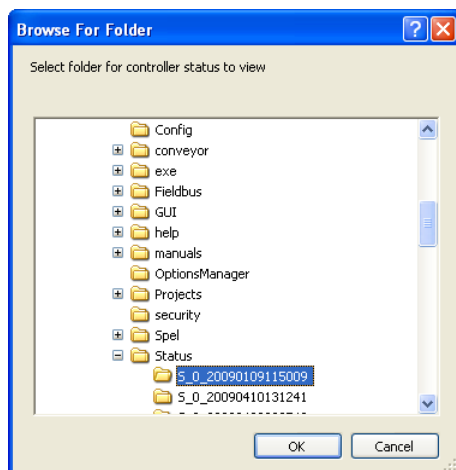
3. Select the drive and parent folder where the information is to be saved.
4. Click **OK** to save the current controller status. A new folder will be created inside the parent folder using the controller serial number and the current date / time.

### View Controller Status

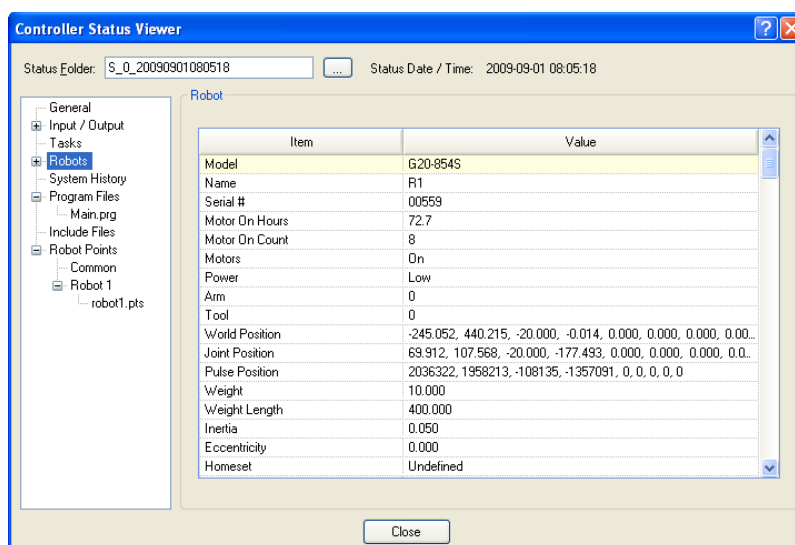
Click the **View Controller Status** button to view the status data stored from a previous status export (see the Export Controller Status section above).

To view controller status:

1. Select Tools | Controller.
2. Click on the **View Controller Status** button to open the **Browse For Folder** dialog.



3. Select the drive and folder where the information is stored. Controller status information is stored in a folder that is named "S\_" plus the controller serial number followed by the date / time.
4. Click **OK** to view the selected controller status.
5. The Controller Status Viewer dialog will be displayed.



6. Select items to view from the tree on the left side of the dialog.
7. To view another controller status, click the ellipses button next to the Status Folder name and select a new status folder.

### Reset Controller

Use the **Reset Controller** button to reset the SPEL controller.

## 5.12 Setup Menu


The Setup Menu contains the following commands:

- System Configuration
- Preferences
- Options

For details on the Vision command, refer to *EPSON RC+ 6.0 Option Vision Guide manual*.

### 5.12.1 System Configuration Command (Setup Menu)

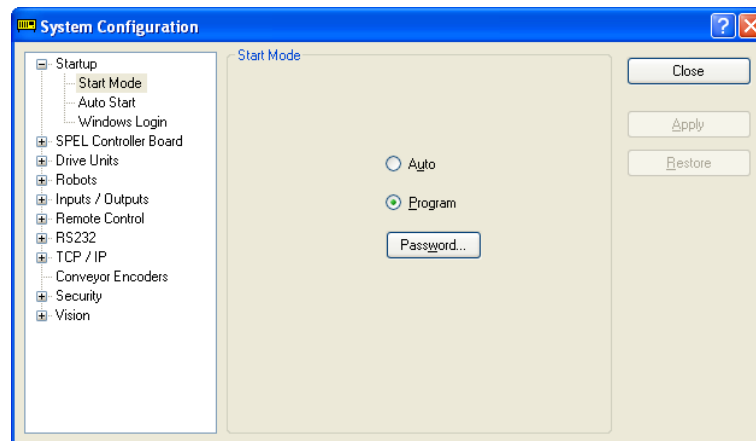
The System Configuration command opens a dialog that contains several pages that are used to configure the system for the EPSON RC+ 6.0 environment.

To open the System Configuration dialog, select Setup | System Configuration .

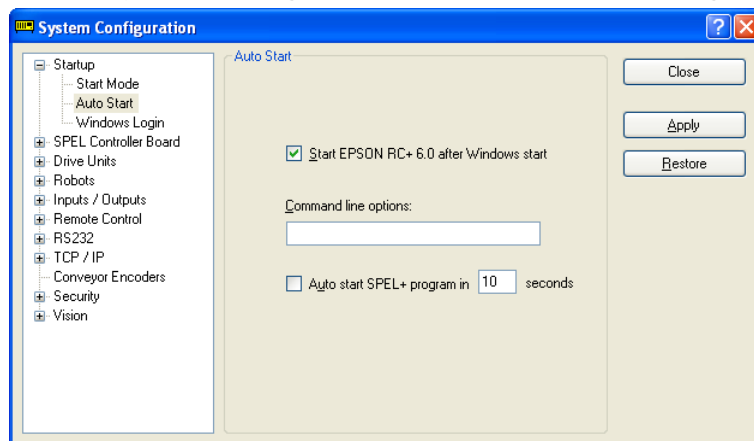
#### Setup: System Configuration: Startup

#### Setup: Preferences: Startup: Start Mode Page

From the Start Mode page, you can choose whether EPSON RC+ 6.0 starts in Auto mode or Program mode.



Item	Description
<b>Auto</b>	Select Auto to start EPSON RC+ 6.0 in Auto mode. Refer to 4. <i>Operation</i> for details.
<b>Program</b>	Select Program to start EPSON RC+ 6.0 in Program mode. Refer to 4. <i>Operation</i> for details.
<b>Password</b>	Click this button to change the password required to enter Program mode from Operator mode when EPSON RC+ 6.0 starts.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Defaults</b>	Click this button to set the default startup mode.
<b>Close</b>	Closes the System Configuration dialog.

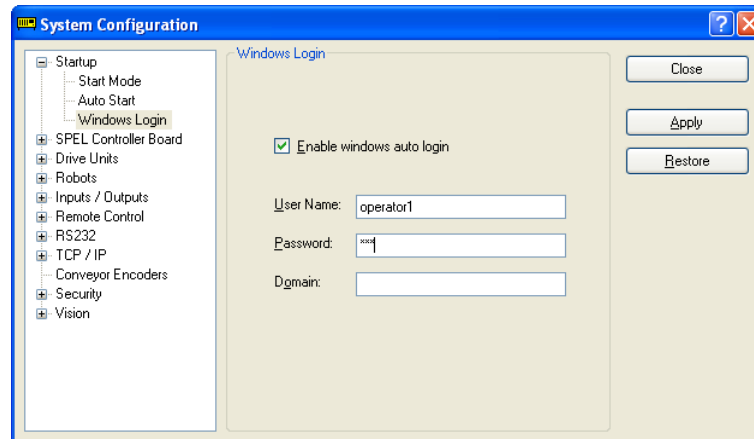
**Setup: System Configuration: Startup: Auto Start Page**

Item	Description
<b>Start EPSON RC+ after Windows start</b>	Check this box if you want EPSON RC+ 6.0 to automatically start after Windows starts.
<b>Command line options</b>	Enter the command line options used when EPSON RC+ 6.0 is automatically started. This is active only when the Start EPSON RC+ 6.0 with Windows start checkbox is not checked.
<b>Auto start SPEL+ program</b>	Check this box if you want to execute the main program after a delay.  This is active only when starting in Operator mode and the control device is "Self".
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Close</b>	Closes the System Configuration dialog.



### Setup: System Configuration: Startup: Windows Login Page

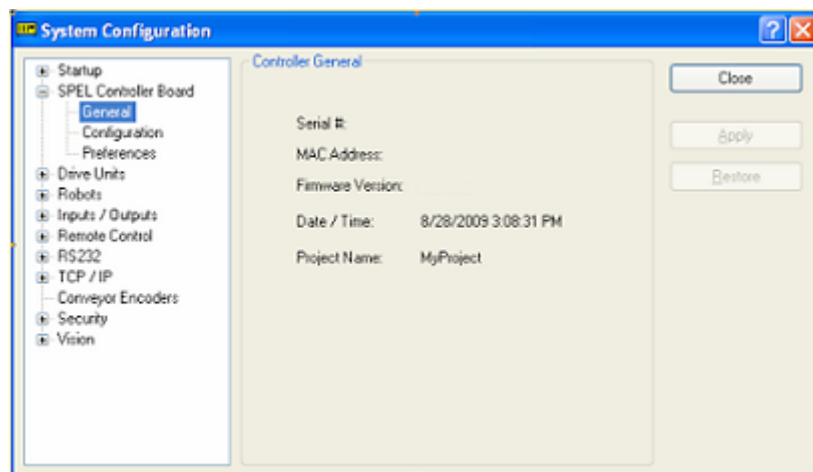
The Windows Login page allows you to configure the automatic login when Windows starts.



Item	Description
<b>Enable windows auto login</b>	Check this box if you want to automatically login to Windows when it starts. You must supply a valid user name, password, and domain.
<b>User Name</b>	Enter the name of a valid Windows user on the system.
<b>Password</b>	Enter the login password for the user.
<b>Domain</b>	Optional. If the PC is the member of a domain, enter the name here.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Close</b>	Closes the System Configuration dialog.

**Setup: System Configuration: SPEL Controller Board****Setup: System Configuration: SPEL Controller Board: General Page**

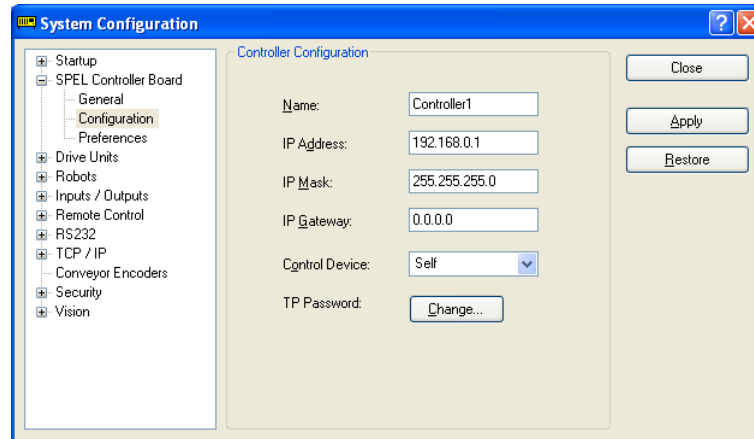
This page allows the user to view general information about the controller.



Item	Description
<b>Serial #</b>	Displays the serial number of the current controller.
<b>MAC Address</b>	Displays the MAC Address of the controller.
<b>Firmware Version</b>	Displays the firmware version used in the current controller.
<b>Date / Time</b>	Displays the current date and time in the controller.
<b>Project Name</b>	Displays the name of the project in the controller.
<b>Close</b>	Closes the Setup Controller dialog.

**Setup: System Configuration: SPEL Controller Board: Configuration Page**

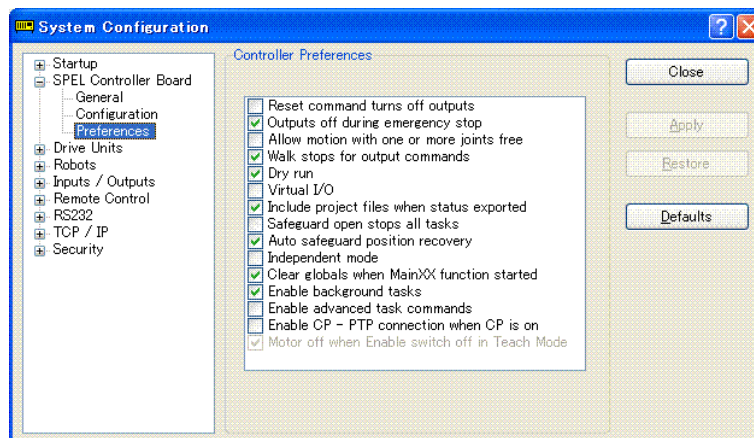
This page allows the user to view and change the controller configuration settings.



Item	Description
<b>Name</b>	Use this text box to change the controller name. You may use any name up to 16 characters long using alphanumeric characters and underscore.
<b>IP Address</b>	Use this text box to set current IP address of the LAN-1 port. The IP Address must be on the same subnet as the PC.
<b>IP Mask</b>	Use this text box to set the IP mask of the LAN-1 port. Note that the IP Mask must match the IP mask used for your network.
<b>IP Gateway</b>	Use this text box to set the IP gateway of the LAN-1 port. This is only required if you will be accessing the controller from outside of the local network.
<b>Control Device</b>	Allows you to select the Control Device.
<b>TP Password</b>	Allows you to change the TP password.
<b>Apply</b>	Saves the current changes. If necessary, the controller will be reset to use the new settings.
<b>Restore</b>	Reverts back to previous settings.
<b>Close</b>	Closes the Setup Controller dialog.

**Setup: System Configuration: SPEL Controller Board: Preferences Page**

This page contains controller preference settings.

**RESET command turns off outputs**

When this preference is turned on, all outputs other than remote control outputs will be turned off when a Reset instruction is executed. The default setting is off.



The outputs of the standard I/O, expansion I/O, and Fieldbus I/O are included in the “outputs” mentioned in the above preferences **RESET command turns off outputs** and **Outputs off during Emergency Stop**. Memory I/O is not affected by these preferences. Therefore, memory I/O bits are not turned off by the RESET command execution or during Emergency Stop.

**Outputs off during Emergency Stop**

When this preference is turned on, all outputs other than remote control outputs will be turned off when emergency stop occurs. Also, no outputs can be turned on until the emergency stop condition is cleared. The default setting is on.

Uncheck this preference to execute I/O On/Off using the NoEmgAbort task or background task after Emergency Stop. If it remains checked, the execution order of turn Off by this preference and turn On using the task are not guaranteed.



You should design your system to always remove all power to output devices when emergency stop occurs. Even if the controller turns off outputs, the I/O hardware could malfunction.

**Allow motion with one or more joints free**

When this preference is turned on, motion commands can be executed after SFree has been used to free one or more joints. The default setting is off.

**Walk stops for output commands**

When checked, the Walk command from the Run Menu will execute lines until after the next motion or output statement (whichever comes first). When unchecked, the Walk command will execute lines until after the next motion statement and will not stop for output statements. The default setting is on.

**Dry run**

This preference allows you to run programs without a robot connected to the controller. All program statements will work. Motion statements will execute approximately the same amount of time as when connected to a robot.

**Virtual I/O**

This preference allows you to run programs using virtual I/O. When Virtual I/O is enabled, I/O commands do not affect the hardware I/O. There are also several commands available for turning on inputs from within a program. The default setting is off.



Remote function is also available when virtual I/O is enabled.

**Include project files when status exported**

This preference allows you to configure whether project files are included or not when the controller status is exported. Refer to 5.11.8 *Controller Command- Export Controller Status*. The default setting is on.

**Safeguard open stops all tasks**

Check this option to cause all normal tasks and NoPause task to stop when the safeguard is open. Only NoEmgAbort task and background tasks will continue.

This option can be used in applications where pause / continue are not required.

The default setting is off.

**Auto safeguard position recovery**

This preference allows you to move a robot back to the position where it was at the safeguard opened when continuing the program execution.

**Auto recover ON** Automatically turns ON a motor and moves a robot in low power status to the position where it was when the safeguard opened.

Continues the usual cycle. (Default)

**Auto recover OFF** In the Run Window and Operator Window, when an operator clicks the **Continue** button, a dialog with a **Recover** button will be shown. The operator needs to hold down the **Recover** button until the motor is ON and the robot's return is finished. Otherwise the robot will stop before reaching final position.

After verifying that the robot's return is finished, the operator clicks the **Continue** button to continue the usual cycle.

For more information, refer to 14. *Security – Recover motion to safeguard open position*.

**Independent Mode**

This preference allows you to use the controller without interfacing with the Windows (Independent mode).

Usually you can use the controller after the EPSON RC+ 6.0 becomes active. Use this option when you want to use the controller through the external device using Remote I/O. This is turned off by default.

**Initialize global variables when Main XX function started**

This preference allows you to initialize the global variables as the main function becomes active.

Turn off this preference when you sue the global variables from the background task.

Otherwise, the variables will be initialized by the controller and the variable-access conflict from tasks will occur. This is turned on by default.

**Enable background tasks**

This preference allows you to execute background tasks.

This is turned off by default.

**Enable advanced task commands**

This preference allows you to execute StartMain, Cont, Recover, Reset, Error commands.

This is turned off by default.



CAUTION

- Before you execute StartMain, Cont, Recover, Reset, Error commands, you should understand each command's specification and verify that the system has the appropriate condition to execute these commands.

Improper use, such as executing commands continuously in a loop, can reduce the security of system.

**Enable CP – PTP connection when CP is on**

This preference allows you to overlap the trajectories of CP motion and PTP motion during CP ON.

NOTE



Over-speed error or Over-acceleration-speed error may occur according to the motion acceleration / deceleration speed setting. If the error occurs, adjust the acceleration / deceleration speed setting or uncheck this checkbox.

**Auto LJM (Least Joint Motion)**

This preference allows you to enable Auto LJM at the controller startup. To disable Auto LJM temporarily, use AutoLJM Off command.

NOTE



If Auto LJM is enabled at all times, this function automatically adjusts the posture of the manipulator to reduce the motion distance, even when you intended to move the joint widely. Therefore, it is recommended to disable Auto LJM at the controller startup and operate the manipulator as you desired using AutoLJM On command or LJM function.

**Disable Point flag check**

This preference allows you to continue operation even when point flags, one was specified as a target point and the other one after the motion completion, do not match in a CP motion.

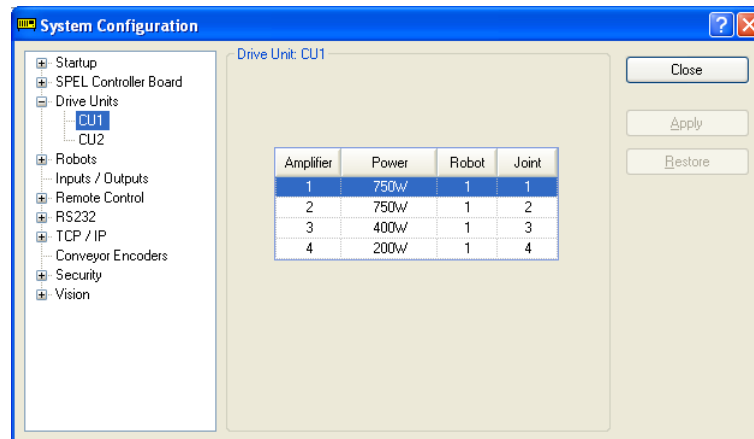
However, if the flags do not match at the transferring point while CP On is used, the manipulator will stop at the point and the motion will not become a path motion.

**Motor off when Enable switch off in Teach Mode**

This preference is read-only. It shows whether motors will be turned off when the Enable switch is off during Teach Mode.

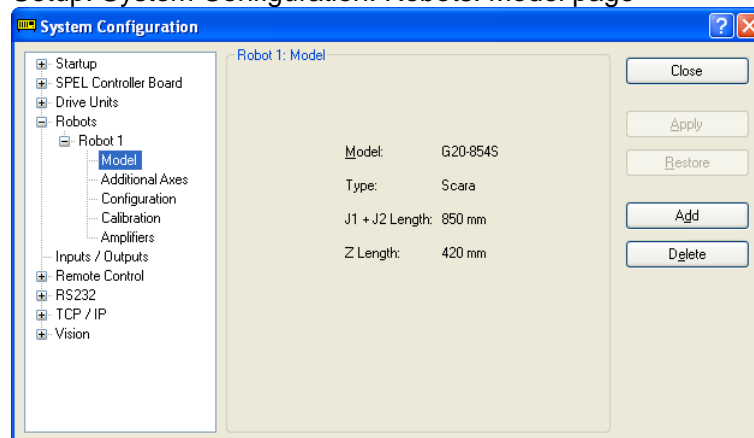
## Setup: System Configuration: Drive Units page

This page shows the Drive Unit status. The power, robot, and joint settings are shown for each amplifier.



## Setup: System Configuration: Robots

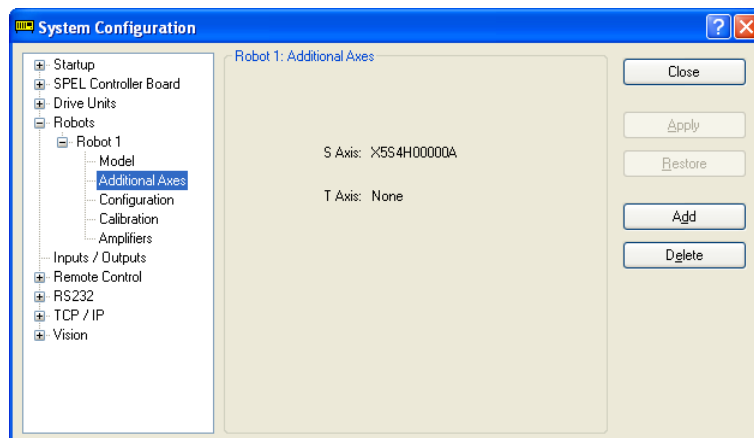
Setup: System Configuration: Robots: Model page



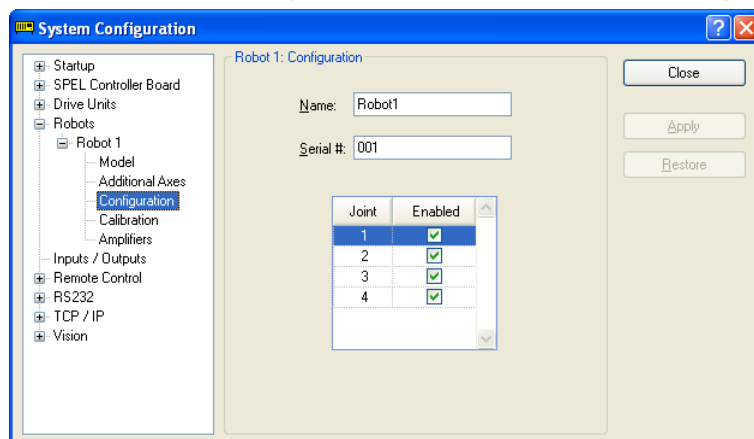
Item	Description
<b>Model</b>	Displays the robot model.
<b>Type</b>	Displays the robot type.
<b>Length</b>	Displays the robot length (J1 + J2 for SCARA robots) or reach for 6-axis robots.
<b>Add</b>	Adds a robot.
<b>Remove</b>	Deletes a robot.
<b>Close</b>	Closes the System Configuration dialog.

**Setup: System Configuration: Robots: Robot\*\*: Additional Axes**

For details of the additional ST axis, refer to 9.2 *Configuration of Additional Axes*.



Item	Description
<b>S Axis</b>	Displays the configuration of additional S axis.
<b>T Axis</b>	Displays the configuration of additional T axis.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Add</b>	Adds an additional axis.
<b>Remove</b>	Deletes an additional axis.
<b>Close</b>	Closes the dialog.

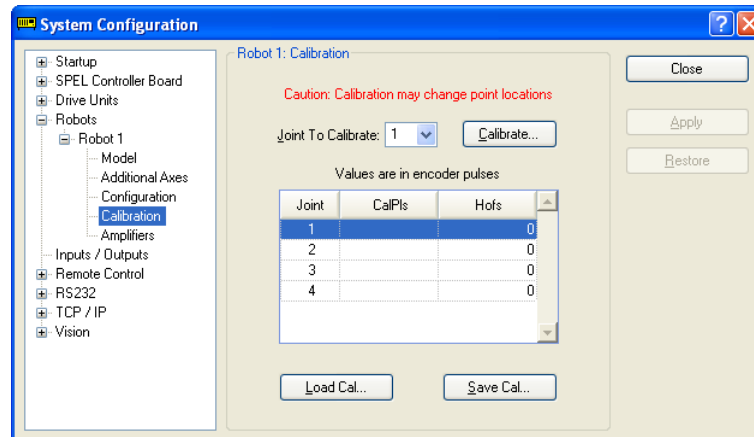
**Setup: System Configuration: Robots: Robot\*\*: Configuration page**

Item	Description
<b>Name</b>	Enter a Name for the robot.
<b>Serial #</b>	Enter the Serial number of the robot.
<b>Joint</b>	These checkboxes determine if the respective joint is enabled or disabled.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Close</b>	Closes the Setup Controller dialog.



**Setup: System Configuration: Robots: Robot\*\*: Calibration Page**

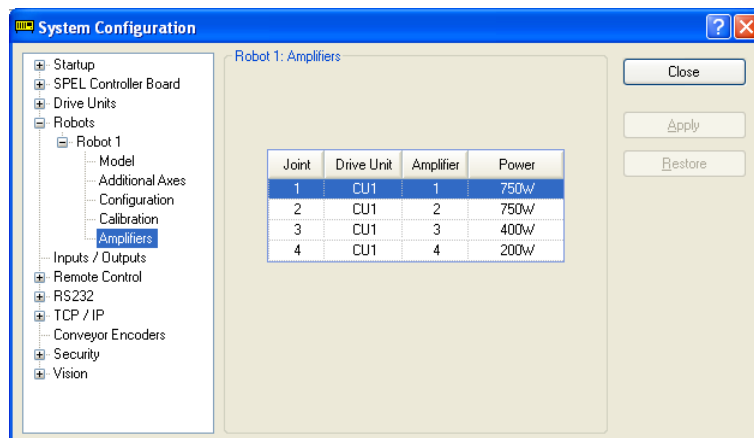
You can calibrate each joint of the robot from this page.



Item	Description
<b>Joint to Calibrate</b>	Select the joint that you want to Calibrate.
<b>Calibrate</b>	Starts the Calibration Wizard dialog that walks you through the calibration process.
<b>Calpls</b>	These are the Calpls settings for each joint. Normally, the calibration wizard will calculate these values.
<b>Hofs</b>	These are the Hofs settings for each joint. Normally, the calibration wizard will calculate these values.
<b>Load Cal</b>	Use this button to load data from a previously save calibration file. After the data is loaded, the grid will be refreshed to show the values.
<b>Save Cal</b>	Use this button to save the calibration data to a calibration file.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Close</b>	Closes the Setup Controller dialog.

**Setup: System Configuration: Robots: Robot\*\*: Amplifiers Page**

This page shows the power values for the motor amplifiers installed in the controller.



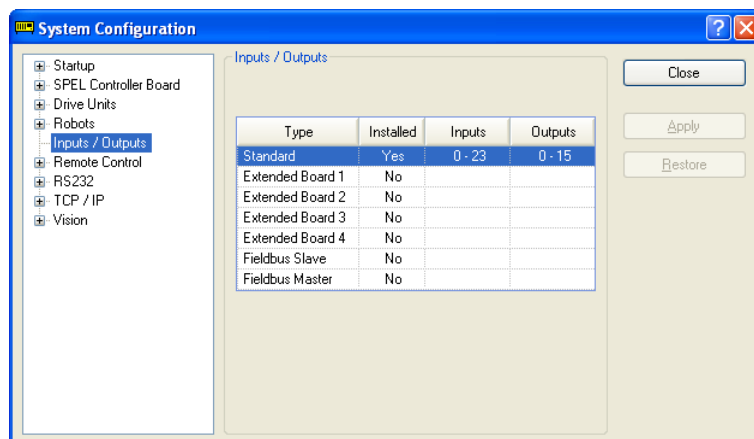
Item	Description
------	-------------

<b>Robot Amplifiers</b>	This shows the power for each robot amplifier currently in the controller along with the associated drive unit and amplifier number.
-------------------------	--

<b>Close</b>	Closes the System Configuration dialog.
--------------	---

**Setup: System Configuration: Inputs / Outputs Page**

This page shows the I/O hardware installed in the controller. There are no settings to configure.

**Setup: System Configuration: Inputs / Outputs Page: Fieldbus Master**

For details of Fieldbus master, refer to the *Robot Controller RC620 option: Fieldbus I/O manual*.

**Setup: System Configuration: Inputs / Outputs Page: Fieldbus Slave**

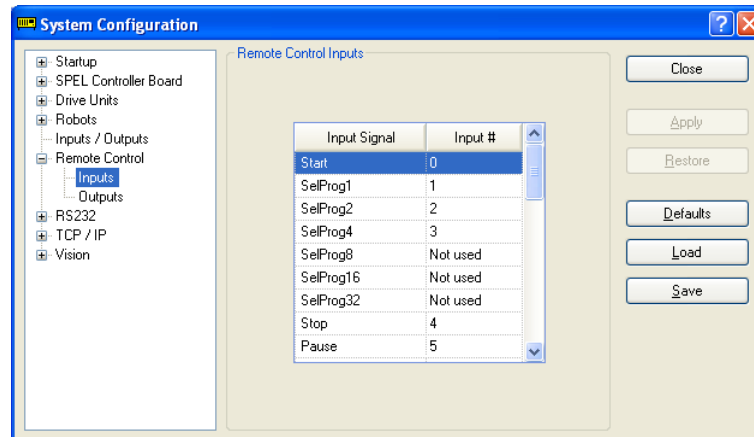
For details of Fieldbus slave, refer to the *Robot Controller RC620 option: Fieldbus I/O manual*.

**Setup: System Configuration: Remote**

For details of Remote function, refer to *11. Remote Control*.

## Setup: System Configuration: Remote Control Inputs Page

Use this page to configure the controller remote control inputs.



Item	Description
<b>Input #</b>	Select an input bit to use for the corresponding input signal. Select "Not used" to disable the remote input. For example, if "Start" is assigned to I/O input bit 0, select "Not used" to use this as a normal I/O input.

Input Signal	Input #
Start	Not used
SelProg1	Not used
SelProg2	0
SelProg4	1
SelProg8	2
SelProg16	3
SelProg32	4
Stop	5
Pause	6
	Not used
	Not used

**Apply** Saves the current changes.

**Restore** Reverts back to previous settings.

**Defaults** Click this button to set the default remote inputs. First, a dialog box will be displayed asking you which type of inputs to use for defaults: Standard, Fieldbus master, or Fieldbus slave I/O. You can also select Clear All to set all remote inputs to Not used.

**Load** Reads the assigned remote inputs and outputs from a file on the PC and save it in the controller.

**Save** Saves the assigned remote inputs and outputs shown in the dialog to a file on the PC.

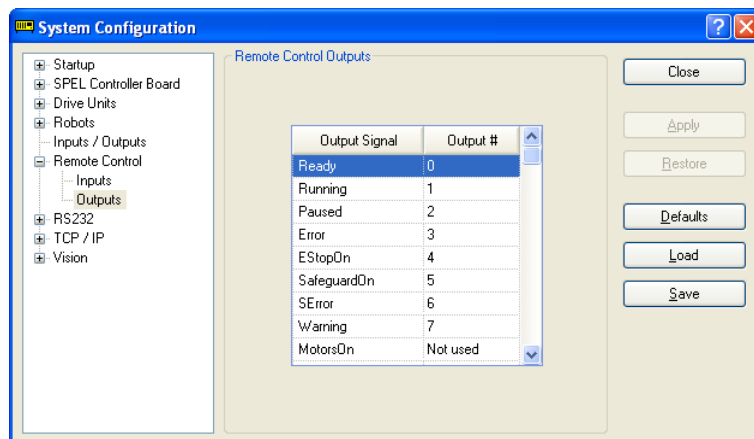
**Close** Closes the Setup Controller dialog.



Both the remote inputs and outputs are loaded or saved together when using **Load** or **Save**.

## Setup: System Configuration: Remote Control Outputs Page

Use this page to configure the controller remote control outputs.



Item	Description
<b>Output #</b>	Select an output bit to use for the corresponding output signal. Select "Not used" to disable the remote output. For example, if "Ready" is assigned to I/O output bit 0, select "Not used" to use this as a normal I/O output.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Defaults</b>	Click this button to set the default remote outputs. First, a dialog box will be displayed asking you which type of outputs to use for defaults: Standard, Fieldbus master, or Fieldbus slave I/O. You can also select Clear All to set all remote outputs to Not used.
<b>Load</b>	Reads the assigned remote inputs and outputs from a file on the PC and save it in the controller.
<b>Save</b>	Saves the assigned remote inputs and outputs shown in the dialog to a file on the PC.
<b>Close</b>	Closes the Setup Controller dialog.

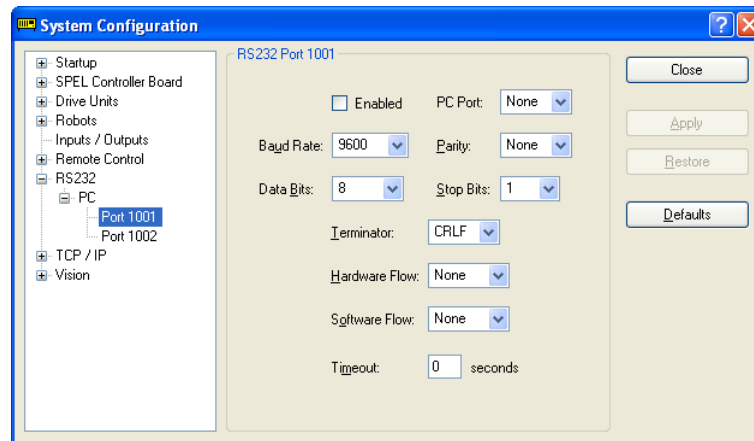


Both the remote inputs and outputs are loaded or saved together when using **Load** or **Save**.

## Setup: System Configuration: RS232

### Setup: System Configuration: RS232: PC Page

Use this page to configure the RS232 ports on PC.

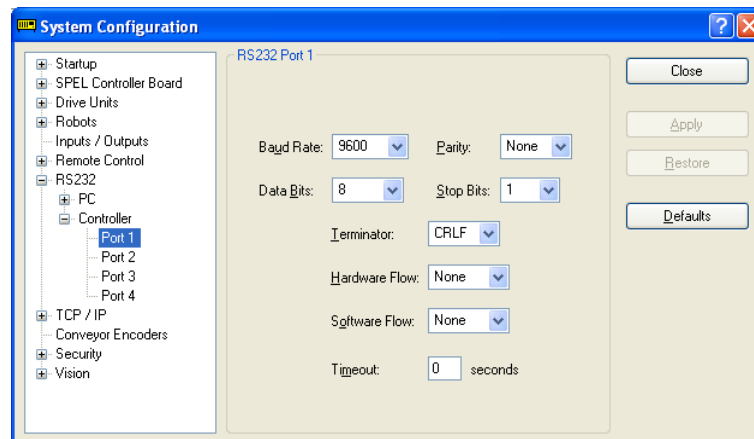


#### To configure an RS-232 port

1. Select System Configuration from the Setup Menu and select the page for the RS232C port you want to configure.
2. Select the [PC port] and change the settings as desired.
3. Set the [Enable] check box.
4. Click **Apply** to save the new settings and click **Close**.

### Setup: System Configuration: RS232: Controller Page

There is one page for each RS232C port. If there are no RS232C ports installed in the special slot, then no selections are visible in the tree.

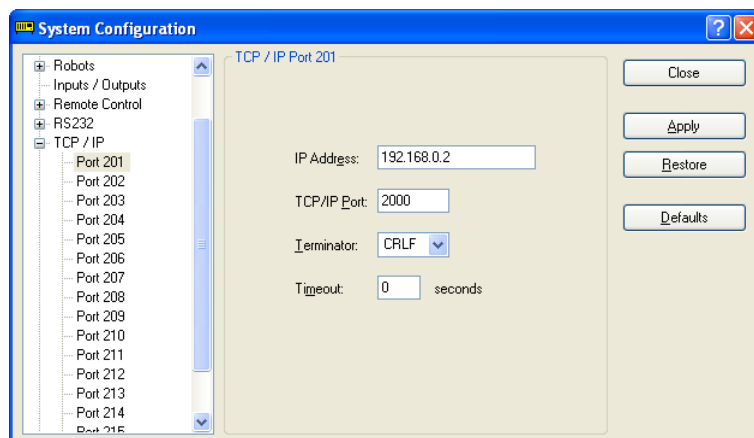


#### To configure an RS-232 port

1. Select System Configuration from the Setup Menu and select the page for the RS232C port you want to configure.
2. Change the settings as desired.
3. Click **Apply** to save the new settings and click **Close**.

### Setup: System Configuration: TCP/IP Pages

There is one page for each TCP / IP port in the controller.



#### To configure a TCP/IP port

1. Select System Configuration from the Setup Menu and select the page for the TCP/IP port you want to configure.
2. Enter the host name or IP address for the controller or PC that you want this controller to communicate with.
3. Enter the TCP/IP port number. This must be the same port number that is used on the host device. It must be different from any of the other TCP/IP port numbers used for the other TCP/IP ports.
4. Change the other settings as desired.
5. Click **Apply** to save the new settings and click **Close**.

### Setup: System Configuration: Conveyor Encoder

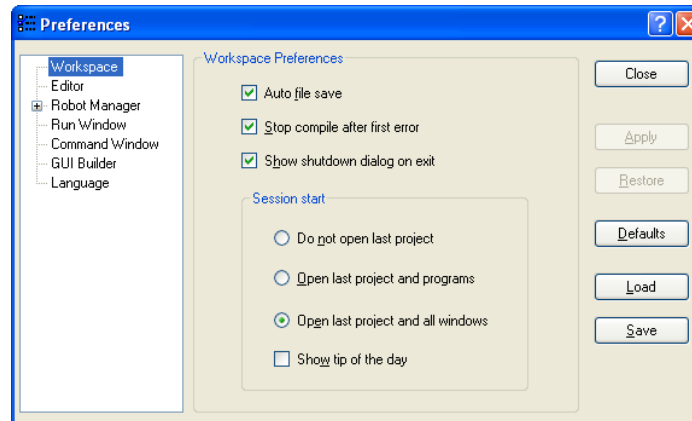
For details, refer to *15. Conveyor Tracking*.

### 5.12.2 Preferences Command (Setup Menu)

The Preferences command opens a dialog that contains several pages that are used to configure user preferences for the EPSON RC+ 6.0 environment. To open the Preferences dialog, select Setup | Preferences.

#### Setup: Preferences: Workspace Page

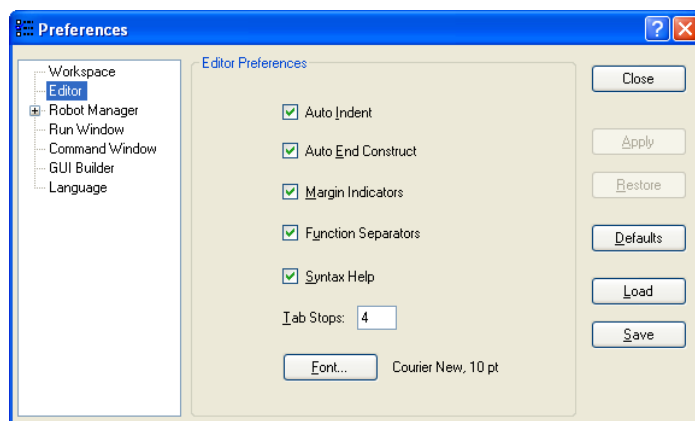
From this page, you can configure your workspace preferences.



Item	Description
<b>Auto file save</b>	Checking this box will cause EPSON RC+ 6.0 to automatically save any open files before executing a command that requires the file to be saved. For example, if a file needs to be saved before executing a project build, the file will automatically be saved before running the build. Default is On.
<b>Stop compile after first error</b>	Stops compile after first error occurs. This makes it easier to see the first error in the status pane and allows you to fix one error at a time. Default is On.
<b>Display the shutdown dialog on exit</b>	Displays the shutdown dialog when closing the EPSON RC+ 6.0. For details, refer to <i>EPSON RC+ Online Help</i> or <i>SPEL<sup>+</sup> Language Reference manual: Shutdown</i> . Default is ON.
<b>Do not open last project</b>	If this radio button is selected, the last project will not be opened when EPSON RC+ 6.0 is started.
<b>Open last project and program file</b>	If this radio button is selected, the last project will be opened and any program windows that were previously opened will be opened.
<b>Open last project and all windows</b>	If this radio button is selected, the last project will be opened and all windows will be restored to their previous locations. This is the default setting.
<b>Show Tip of the Day</b>	If this check box is on, the Tip of the Day dialog will be displayed when EPSON RC+ 6.0 is started.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to previous settings.
<b>Default</b>	Sets the default values.
<b>Load</b>	Reads the preferences previously saved on the PC.
<b>Save</b>	Saves the preferences to a file on the PC.
<b>Close</b>	Closes the Preferences dialog.

## Setup: Preferences: Editor Page

This page is used to configure your preferences for the program editor windows.

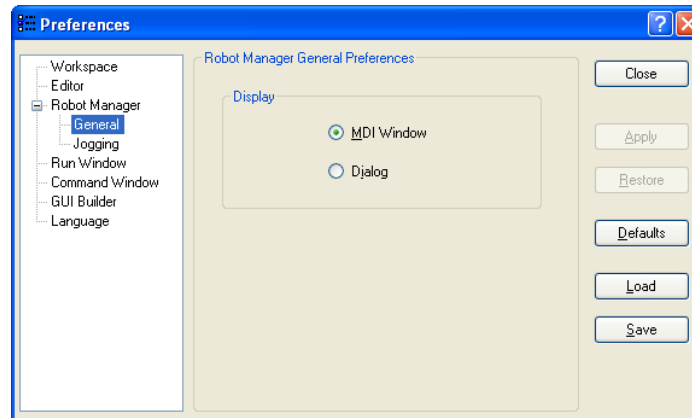


Item	Description
<b>Auto Indent</b>	Check this box if you want new lines to follow the indentation for the previous line. Also, lines will automatically be indented after Do, If, Else, For, Select, and Case statements. Default is on.
<b>Auto End Construct</b>	Check this box if you want EPSON RC+ 6.0 to add the end construct statement for a loop construct. For example, if you enter a For statement, then a Next statement will be added automatically. Default is on.
<b>Margin Indicators</b>	Check this box to display a margin on the left side. This margin is used to indicate lines with breakpoints, current step line, current execution line. Default is on.
<b>Function Separators</b>	Check this box to display a line after each Fend statement. Default is on.
<b>Syntax Help</b>	Check this box to enable the Syntax Help window. The Syntax Help window displays syntax for a keyword after it has been typed. Default is on.
<b>Tab Stops</b>	Type in the number of columns to move for the TAB key. Default is 4.
<b>Font</b>	Click on the Font button to open the fonts dialog. Choose the font you desire for the editor. The monitor window also uses the editor font. The current font name and size is displayed next to the Font button.
<b>Apply</b>	Applies the current settings.
<b>Restore</b>	Reverts back to the previous settings.
<b>Defaults</b>	Sets default value.
<b>Load</b>	Reads the preferences previously saved on the PC.
<b>Save</b>	Saves the preferences to a file on the PC.
<b>Close</b>	Closes the Preferences dialog.



## Setup: Preferences: Robot Manager: General Page

This page lets you configure your preferences for the Robot Manager.

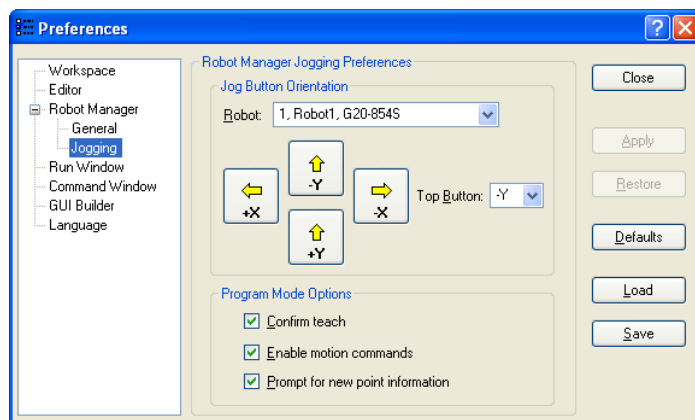


Item	Description
<b>Display</b>	Choose if you want the Robot Manager to be displayed as an MDI Window or as a Dialog.
<b>Apply</b>	Applies the current settings.
<b>Restore</b>	Reverts back to the previous settings.
<b>Defaults</b>	Sets default values.
<b>Load</b>	Reads the preferences previously saved on the PC.
<b>Save</b>	Saves the preferences to a file on the PC.
<b>Close</b>	Closes the Preferences dialog

The Robot Manager can be displayed as an MDI child window (default) or as a dialog. When displayed as an MDI child, the Robot Manager is displayed in the MDI document area and can remain open while you work with other windows and dialogs. When displayed as a dialog, you can only work with the Robot Manager controls until you close the dialog. When using screen resolutions less than 1024 × 768, only the Dialog mode is allowed.

## Setup: Preferences: Robot Manager: Jogging Page

This page lets you configure the Robot Manager Jog and Teach page.



### Setting Jog Button Orientation

Item	Description
------	-------------

**Robot** Select a robot.

The jog button orientations are useful for “aligning” your PC monitor with the robot’s Cartesian coordinate system. Align the buttons so that the robot moves in the direction of the arrows.

You can change the orientation of the jogging buttons and arrow keys for the X and Y axes by selecting the desired top button from the **Top Button** dropdown list.

You can also click on one of the buttons to change it to the top button position.

### Program Mode Options



These options affect the Robot Manager Jog & Teach page when used from program mode.

These settings do not affect the Robot Manager when used for operators in auto mode, such as for the Operator Window or from VB Guide. To configure the Robot Manager for operators, see *Project / Properties / Operator Settings / Robot Manager*.

Item	Description
------	-------------

**Confirm teach** Check this box if you want a confirmation prompt each time you press the Teach button on the Robot Manager Jog & Teach page.

**Enable motion commands** Check this box if you want to execute motion commands (Go, Jump, etc.) from the Robot Manager Jog & Teach page.

**Prompt for new point information** Check this box if you want to be prompted for point label and description when a new point is taught using the Teach button.

**Apply** Applies the current settings.

**Restore** Reverts back to the previous settings.

**Defaults** Sets default values.

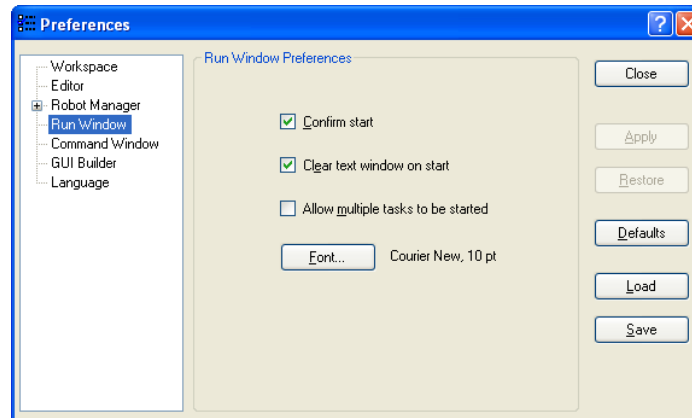
**Load** Reads the preferences previously saved on the PC.

**Save** Saves the preferences to a file on the PC.

**Close** Closes the Preferences dialog.

## Setup: Preferences: Run Window Page

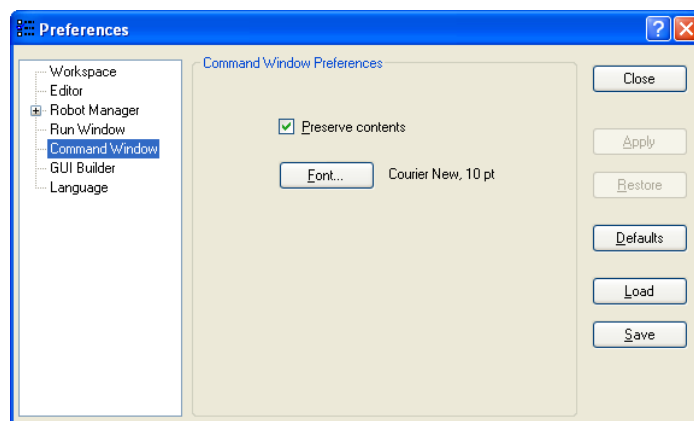
This page allows you to change preferences for the Run Window.



Item	Description
<b>Confirm Start</b>	This checkbox allows you to select if you want to see a confirmation message box before a program is started.
<b>Clear text window on start</b>	Checking this will cause the Run Window text pane to be cleared each time the <b>Start</b> button is clicked.
<b>Allow multiple tasks to be started</b>	Checking this allows you to start a task from the Run window while other tasks are running. The Start button will not be disabled after starting a task.
<b>Font</b>	Click on the Font button to open the fonts dialog. Choose the font you desire for the Run window. The current font name and size is displayed next to the Font button.
<b>Apply</b>	Applies the current settings.
<b>Restore</b>	Reverts back to the previous settings.
<b>Defaults</b>	Sets default values.
<b>Restore</b>	Reverts back to the previous settings.
<b>Defaults</b>	Sets default values.
<b>Load</b>	Reads the preferences previously saved on the PC.
<b>Save</b>	Saves the preferences to a file on the PC.
<b>Close</b>	Closes the Preferences dialog.

## Setup: Preferences: Command Window Page

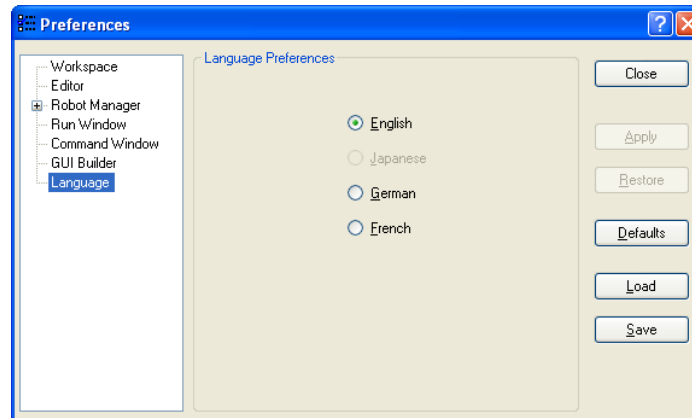
This page allows you to change preferences for the Command Window.



Item	Description
<b>Preserve contents</b>	Checking this option will cause the command window to preserve its contents between sessions.
<b>Font</b>	Click on the Font button to change the font for the Command window.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to the previous values.
<b>Defaults</b>	Set default values.
<b>Load</b>	Reads the preferences previously saved on the PC.
<b>Save</b>	Saves the preferences to a file on the PC.
<b>Close</b>	Closes the Preferences dialog.

## Setup: Preferences: Language

This page allows you to change the EPSON RC+ 6.0 GUI language.



When EPSON RC+ 6.0 is installed on a Windows system using a Western language, then the English, German, and French selections are available.

If it is installed on a Windows system using Japanese, then English and Japanese are available.

After selecting the desired language, you must reboot EPSON RC+ 6.0.

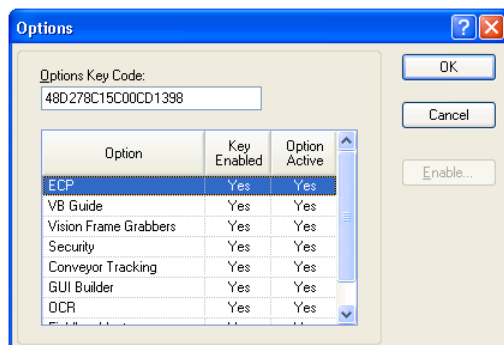
If you select English for EPSON RC+ 6.0, but you are using Japanese Windows, then messages boxes and some file navigation dialogs will be in Japanese.

Item	Description
<b>Language</b>	This set of option buttons allows you to choose which language to use for the EPSON RC+ 6.0 GUI.
<b>Apply</b>	Saves the current changes.
<b>Restore</b>	Reverts back to the previous values.
<b>Defaults</b>	Set the default language.
<b>Load</b>	Reads the preferences previously saved on the PC.
<b>Save</b>	Saves the preferences to a file on the PC.
<b>Close</b>	Closes the Preferences dialog.

### 5.12.3 Options Command (Setup Menu)

This dialog allows you to view and enable options in the controller.

EPSON RC+ 6.0 uses a key that is stored in the Spel controller board to enable options on the system.



If an option is not enabled, you can enable it by purchasing it from your distributor. When you call to purchase, you must give the **Options Key Code** to the operator. You will then be given a code to enable the option for the current software options key.

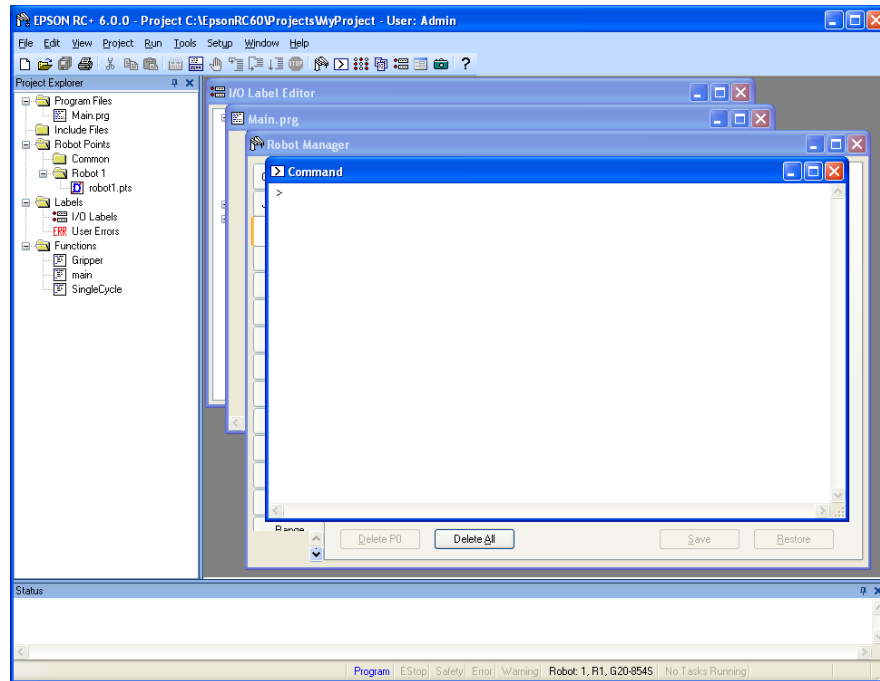
After receiving the code, click the **Enable** button and enter the code. The option you purchased should now be enabled.

## 5.13 Window Menu

The Window Menu contains selections for managing the currently open EPSON RC+ 6.0 child windows.

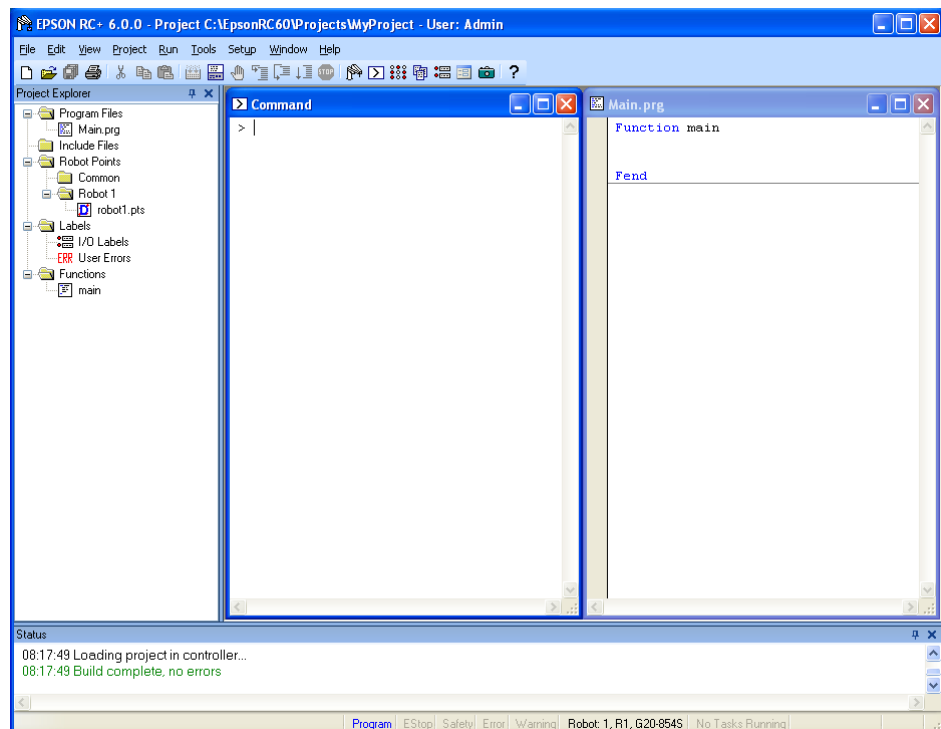
### 5.13.1 Cascade Command (Window Menu)

Use Cascade to show all of the currently open files in windows of the same size, stacked one on top of another.



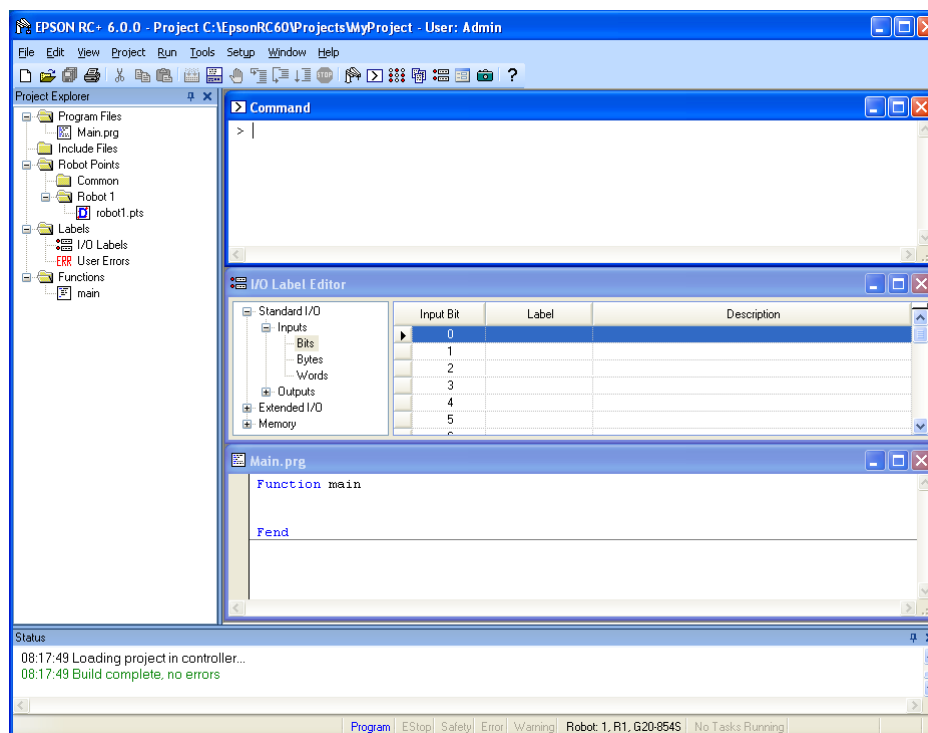
### 5.13.2 Tile Vertical Command (Window Menu)

Use Tile Vertical to evenly display all open windows vertically.



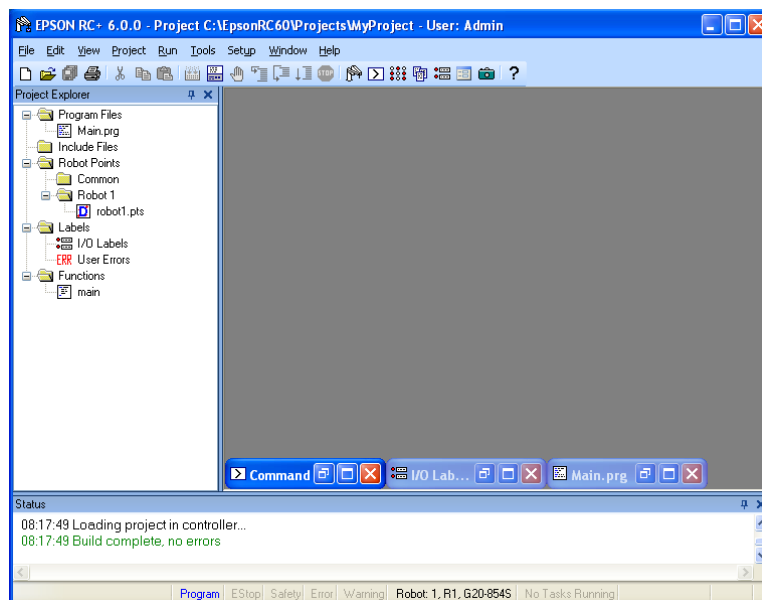
### 5.13.3 Tile Horizontal Command (Window Menu)

Use Tile Horizontal to evenly display all open windows horizontally.



### 5.13.4 Arrange Icons Command (Window Menu)

Arrange the icons for all child windows that have been minimized.



### 5.13.5 Close All Command (Window Menu)

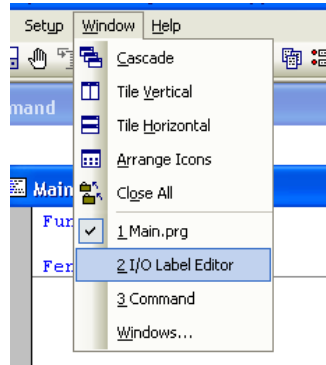
This command closes all EPSON RC+ 6.0 child windows.



### 5.13.6 1, 2, 3 Command (Window Menu)

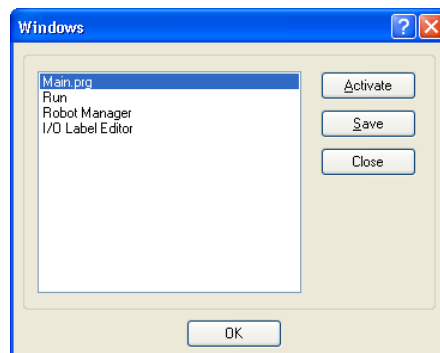
A listing of currently open document windows is displayed at the bottom of the Window Menu.

When you choose an open window from the listing, you make that document active. A check mark appears in front of the document name of the currently active window.



### 5.13.7 Windows Command (Window Menu)

This command displays a dialog that contains a list of all currently open EPSON RC+ 6.0 windows.



Item	Description
<b>Activate</b>	Brings the selected window into focus.
<b>Save</b>	Saves the contents of the selected windows.
<b>Close</b>	Closes the selected windows.
<b>OK</b>	Closes the dialog.

### 5.14 Help Menu

The Help Menu contains selections for accessing the help system and manuals along with version information.

#### 5.14.1 How Do I Command (Help Menu)

Select How Do I to view topics that contain information for performing common tasks in EPSON RC+ 6.0.

##### Shortcuts

Keys: Ctrl + F1

#### 5.14.2 Contents Command (Help Menu)

This command opens the Contents view for the EPSON RC+ 6.0 online help system.

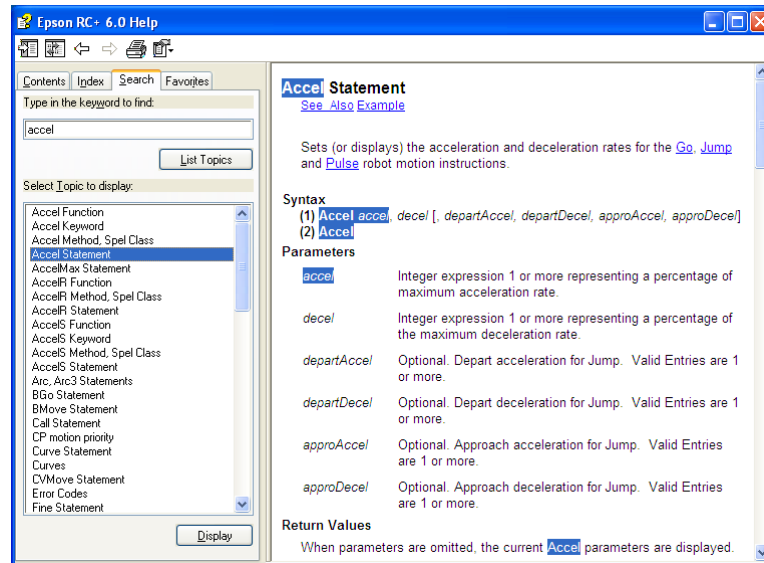
From the Contents view, you can navigate through all of the topics in the help system. Double-click on a book icon to open or close the subtopic list contained within the book folder.



### 5.14.3 Search Command (Help Menu)

This command opens the Search view for the EPSON RC+ 6.0 online help system.

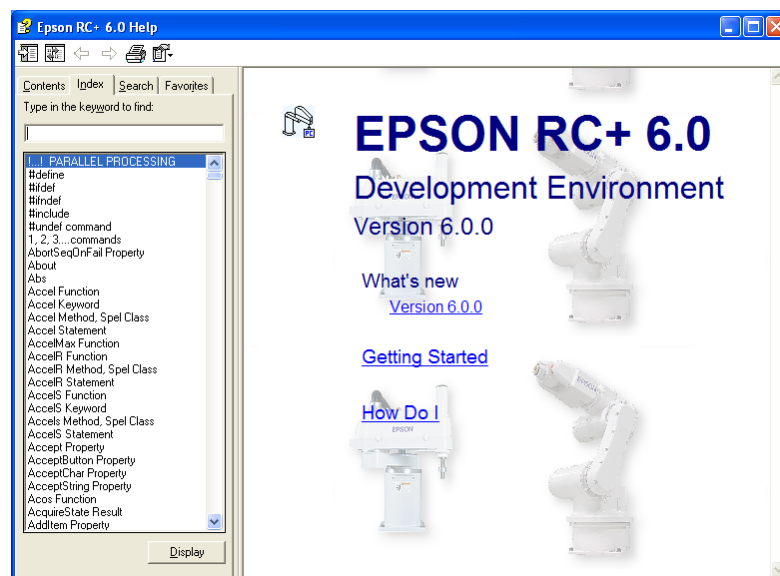
From the Search view, you can type in one or more keywords and click List Topics to show a list of all topics containing one or more of the keywords. The keywords are highlighted in the topics as shown below.



### 5.14.4 Index Command (Help Menu)

This command opens the Index view for the EPSON RC+ 6.0 online help system.

From the Index view, as you begin typing in a keyword, the alphabetical topic list will show the keywords starting with the letters you have typed.



### 5.14.5 Manuals Submenu (Help Menu)

The Help Menu Manuals submenu contains selections for each of the manuals in Adobe PDF format. These include manuals for EPSON RC+ 6.0, SPEL<sup>+</sup> Language Reference, Controller, Manipulator, and the Options.

### 5.14.6 About EPSON RC+ 6.0 Command (Help Menu)

The About command displays a dialog box showing the current version of the EPSON RC+ 6.0 software, along with copyright and license information. When calling technical support about EPSON RC+ 6.0, you should report the version you are using from this dialog.



## 6. The SPEL<sup>+</sup> Language

This chapter contains information about the SPEL<sup>+</sup> Language.

### **Contents**

- Overview
- Program structure
- Commands and statements
- Function and variable names
- Data types
- Operators
- Working with variables
- Working with strings
- Multi-statements
- Labels
- Comments
- Error handling
- Multi-tasking
- Robot coordinate systems
- Robot arm orientations
- Robot motion commands
- Working with robot points
- Input and output control
- Using Traps

### 6.1 Overview

SPEL<sup>+</sup> is a BASIC-like programming language that runs in the controller. It supports multitasking, motion control, I/O control.

Programs are written in ASCII text and then compiled into executable object files. Several language instructions can also be executed in immediate mode from the Command window.

### 6.2 Program Structure

#### 6.2.1 What is a SPEL<sup>+</sup> program?

A SPEL+ program is a collection of functions, variables, and macros. You can put one or more statement in each line of a program (Multi-Statement). Every program file has a PRG extension and is stored in the project directory.

Each project must include at least one program and define the function called "main". "Function main" is the default definition, and an error will occur if Function main is not found.

In addition, you can define other 63 main functions in the same project. Each program has its own start function: main1, main2...main63. Each of the main functions can be started from the Operator window, the remote console, or VB Guide.

A function definition begins with the Function statement and ends with the Fend statement.

The following program file contains two function definitions. Function Main calls function Func1.

```
MAIN.PRG
Function Main
    Call Func1
    ...
Fend
Function Func1
    Jump pickpnt
    ...
Fend
```

#### 6.2.2 Calling functions

You can execute a user function by using the Call statement. The function can reside in any program file in the current project. You can also omit the Call statement if you don't need the return value. When Call is omitted, then parentheses for the arguments must not be supplied. To get a return value, use the function in the right hand side of an expression.

Here are some examples:

```
Call MyFunc(1, 2)
MyFunc 1, 2
Print MyFunc(1, 2)
```

## 6.3 Commands and Statements

Commands and statements consist of a SPEL<sup>+</sup> instruction followed by the parameters for that instruction.

A command is executed immediately. You can execute commands from the Command window or from the Macros dialog box.

Statements can be used only in programs.

Statements can include more than one SPEL<sup>+</sup> instruction. When you put several statements in a line of a program (Multi-Statement), use a semi-colon (;) to separate instructions.

The maximum length for a line is 512 characters.

## 6.4 Function and Variable Names (Naming restriction)

The function name can include up to 64 characters. The variable name can include up to 32 alphanumeric, Japanese, or underscore characters. Characters can be upper case or lower case.

The following names are valid:

Function main

Real real\_var

Integer IntVar

Function and variable names **cannot** begin with an underscore.

SPEL<sup>+</sup> keywords cannot be used as function or variable names.

String variables must have an additional dollar sign ('\$') suffix, as shown in the example below:

```
Function Test
  String modname$
  Print "Enter model name:"
  Line Input modname$
  Print "model is ", modname$
Fend
```

### Restrictions for naming in SPEL+ language

- Characters can be alphanumeric, Japanese, or underscore character.
- Use alphabets for the first letter.
- Characters can be upper case or lower case.
- No keywords can be used.
- Maximum limits of names are as follows.

Name	Max. limit
Point label	16
I/O label	16
User error label	16
Function name	64
Variable name	32
Line label	32

## 6.5 Data Types

You can declare different types of data in your program. All variables must be declared.

The following table shows the different data types for the SPEL<sup>+</sup> language.

Data Type	Size	Range
Boolean	2 byte	True or False
Byte	2 byte	−128 to +127
Double	8 bytes	−1.79E+308 to 1.79E+308 Number of significant figure is 14
Integer	2 bytes	−32768 to +32767
Long	4 bytes	−2147483648 to +2147483647
Real	4 bytes	−3.40E+38 to 3.40E+38 Number of significant figure is 6
String	256 bytes	All ASCII characters Up to 255 characters

## 6.6 Operators

The following table shows the different operators for the SPEL<sup>+</sup> language.

Keyword or Symbol	Description
+	Addition
−	Subtraction
*	Multiplication
/	Division
**	Exponentiation
=	Equal
>	Greater than
<	Less than
>=	Greater or equal
<=	Less or equal
<>	Not equal
And	Performs logical and bitwise AND operation.
Mod	Returns the remainder obtained by dividing a numeric expression by another numeric expression.
Not	Performs logical or bitwise negation of the operand.
Or	Performs the bitwise Or operation on the values of the operands.
Xor	Performs the bitwise Xor operation on the values of the operands.



## 6.7 Working with Variables

### 6.7.1 Variable scopes

There are three different scopes for variables in SPEL<sup>+</sup>:

- Local
- Module
- Global

### 6.7.2 Local variables

Local variables are available to all statements in the same function. Functions using local variable names cannot refer to the same local variables in other functions. This is why they are called locals, because they are local to the function they are being used in.

To declare local variables in a function, use one of the variable declaration instructions at the beginning of the function after the Function statement:

Boolean, Byte, Integer, Long, Real, Double, String

For example, the following function declares several local variables:

```
Function test
  Integer intVar1, intVar2
  Real realVar
  String dataStr$
  Integer array(10)
  .....
Fend
```

### 6.7.3 Module variables

Module variables are available to all functions in the same program file.

To declare module variables in a program, use one of the variable declaration instructions at the beginning of the program before any Function statements:

Boolean, Byte, Integer, Long, Real, Double, String



In order to indicate that a variable is module level, precede the name with "m\_", as shown in the example below. With this, you can improve the program readability.

For example, the following function declares several module level variables:

```
' Module level vars, used by all functions in this file
Integer m_IntVar1, m_IntVar2
Real m_RealVar
String m_DataStr$
Integer m_Array(10)

Function main
  m_IntVar1 = 25
  Call test
Fend

Function test
  Print m_IntVar1
Fend
```

### 6.7.4 Global variables

Global variables can be shared between all functions in a project. The Global instruction is used to declare a global variable.

To declare global variables in a program, use the Global instruction with the desired variable type (Boolean, Byte, Integer, Long, Real, Double, String) at the beginning of the program before any Function statements:



In order to indicate that variables are global, precede the name with "g\_", as shown in the example below. With this, you can improve the program readability.

**Program: MAIN.PRG**

```
Global Integer g_TotalCycles  
Function main  
    Call LoadPart  
    :::  
Fend
```

**Program: LOADPART.PRG**

```
Function LoadPart  
    Jump pick  
    On gripper  
    Wait .1  
    Jump place  
    Off gripper  
    Wait .1  
    g_TotalCycles = g_TotalCycles + 1  
Fend
```

For more information, see Data Types.

### 6.7.5 Global Preserve variables

You can preserve global variable values by using the optional Preserve parameter when you declare global variables.

Preserved variables are stored in the controller's SRAM.

If the data type of a preserved variable is changed, or the number of dimensions is changed, then the variable values will be cleared.



Be careful about the backup battery power, because you will lose the data of global preserve variables stored in SRAM if the battery is weak.

### 6.7.6 Arrays

You can declare local, module, and global variables with up to three dimensions as arrays for all data types.

To declare an array, use this syntax:

***dataType*** *name* ( *ubound1* [ , *ubound2* [ , *ubound3* ] ] )

SPEL<sup>+</sup> arrays are zero based. The first element is referenced with a value of zero.

The total available number of array elements for local variables is 200 for strings and 2000 for all other types.

The total available number of array elements for global preserve variables is 400 for strings and 4000 for all other types.

The total available number of array elements for global and module variables is 10,000 for strings and 100,000 for all other types.

To calculate the total elements used in an array, use the following formula. (If a dimension is not used, substitute 0 for the ubound values.)

total elements = (ubound1 + 1) \* (ubound2 + 1) \* (ubound3 + 1)

Array declaration examples:

```
' Global string array
Global String gData$(10)
Function main
  ' Arrays local to this function
  Integer intArray(10)
  Real coords(20, 10)
```

Use Redim to change the bounds of an array at run time.

```
Integer a(10)
Redim a(20)
```

To preserve values when using Redim, add the Preserve optional argument.

```
Integer a(10)
Redim Preserve a(20)
```

Use UBound to get the maximum element number.

```
Integer i, a(10)
For i = 1 to UBound(a)
  a(i) = i
Next i
```

### 6.7.7 Initial values

All variables are initialized when first used except for Global Preserve variables. Strings are set to empty, and all other variables are set to zero.

### 6.7.8 Clearing arrays

Execute Redim (without Preserve) to clear all of the elements of array variables.

## 6.8 Working with Strings

A string in SPEL<sup>+</sup> is a set of ASCII characters (Code &h01 ~ &hff) with a maximum length of 255.

You must declare strings in your programs with the String instruction.

All string variable names must end with a dollar sign (\$) suffix.

The following table shows the string commands available in SPEL<sup>+</sup>.

Keyword	Description
<b>Asc</b>	Returns the decimal ASCII value of the first character in a string.
<b>Chr\$</b>	Converts an ASCII value into a one character string.
<b>FmtStr</b>	Formats a numerical or date/time expression.
<b>Hex\$</b>	Returns a string containing the hexadecimal value of a number.
<b>InStr</b>	Returns the position of a substring within a string.
<b>LCase\$</b>	Returns the specified string in lower case characters.
<b>Left\$</b>	Returns a substring beginning with the first character of a string.
<b>Len</b>	Returns the length (number of characters) of a string.
<b>LTrim\$</b>	Returns the specified string with left spaces removed.
<b>Mid\$</b>	Returns a substring of a string.
<b>ParseStr</b>	Parses a string into an array of tokens.
<b>Right\$</b>	Returns a substring from the end of a string.
<b>RTrim\$</b>	Returns the specified string with right spaces removed.
<b>Space\$</b>	Returns a string containing a specified number of space (ASCII 32) characters.
<b>Str\$</b>	Converts a number to a string.
<b>String</b>	Declare a string variable in a program.
<b>Tab\$</b>	Returns a tab string.
<b>UCase\$</b>	Returns the specified string in upper case characters.
<b>Val</b>	Converts a string to a number.

## 6.9 Working with Files

SPEL<sup>+</sup> has several commands for handling files.

Keyword	Description
AOpen	Opens a file for append.
BOpen	Opens a file for binary access.
Close	Closes a file.
FileExists	Checks if a file exists.
FolderExists	Check if a folder exists.
FreeFile	Returns an unused file handle.
Input	Inputs one or more variables from a file
Kill	Deletes a file.
Line Input	Inputs line from a file.
Read	Reads a specified number of bytes into a string variable.
ReadBin	Reads binary data.
ROpen	Opens a file for reading.
Seek	Sets the current file pointer.
Flush	Writes data buffer to disk.
WOpen	Opens a file for writing.
Write	Writes out a variable at the current file pointer without appending a line terminator.
WriteBin	Writes binary data.

Before using a file you must open it with one of the following commands: AOpen, Bopen, ROpen, and WOpen. And specify a file number in the Open statement. File number can be 30 ~ 63.

Here is an example to save a text file and read it.

```
Function SaveData(ByRef data$() As String)
    Integer fNum, i

    fNum = FreeFile
    WOpen "c:\mydata\data.txt" As #fNum
    ' Store the count
    Print #fNum, UBound(data$)
    For i = 0 To UBound(data$)
        Print #fNum, data$(i)
    Next i
    Close #fNum
Fend
```

```
Function LoadData(ByRef data$() As String)
    Integer fNum, i

    fNum = FreeFile
    ROpen "c:\mydata\data.txt" As #fNum
    Input #fNum, i
    Redim data$(i)
    For i = 0 To UBound(data$)
        Input #fNum, data$(i)
    Next i
    Close #fNumFend
```

### NOTE



The files on the networks are not available to access.

## 6.10 Multi-statements

A program statement can contain several statements separated by semi-colons. The total length of a multi-statement program line cannot exceed 255 characters.

For example:

```
Function Test
    Pass P1; Pass P2; Go P3      ' Multi-statement
Fend
```

It is not recommended to use multi-statements. Multi-statements can make your code more difficult to read and debug.

## 6.11 Labels

A program line is an alphanumeric name followed by a colon (":") that marks a location in a program for a GoTo or GoSub statement. The name may be up to 32 characters long and can include alphanumeric characters and the underscore ("\_") character if it is not the first character. You cannot use any SPEL+ keywords as labels.

For example:

```
Function Main
    Do
        Jump P1
        Jump P2
        If Sw(1) Then GoTo MainAbort
    Loop
MainAbort:    ' Program label
    Print "Program aborted"
Fend
```

## 6.12 Comments

Use comments to add notes to your programs. An apostrophe character (') starts a comment.

### Examples of comments

```
Function Main
  ' ***** Main Demo Program *****
  Xgt conveyor      ' Start up the task for conveyor
  Do
    Print "Press ENTER to run demo cycle"
    Print "Press CTRL+C to quit"
    Input dummy
    Call demo      ' Execute the demo function
  Loop             ' Return to start of main loop
```

## 6.13 Error Handling

When an error occurs in a SPEL<sup>+</sup> function, you can cause execution to be transferred to an error handling routine for processing the error. The routine must be inside a function definition.

The table on the next page shows the program instructions that are used for error handling.

Item	Purpose
<b>OnErr</b>	Use the OnErr statement to define the location of the error handling routine.
<b>Err</b>	Use Err to retrieve the number for the current error status. Use this in the error handling routine to determine which error has occurred.
<b>Error</b>	Generate a user defined error which can be caught by an error handler.
<b>Era</b>	Use Era to retrieve the axis number for which the error occurred. This is normally used in the error handling routine.
<b>Erl</b>	Use Erl to retrieve the line number in which the error occurred. This is normally used in the error handling routine.
<b>Ert</b>	Use Ert to retrieve the task number in which the error occurred. This is normally used in the error handling routine.
<b>ErrMsg\$</b>	Use ErrMsg\$ to retrieve the error message associated with a specified error number.

### User Errors

You can define your own error messages by using the User Error Editor which is available from the Tools Menu. For details refer to *5.11.7 User Error Editor (EPSON RC+ 6.0 GUI)*.

### Example

The following example shows a simple error handling routine. When an error occurs, program execution goes to the ErrHandler label, where the error handler starts. The error number is displayed and the operator is asked to continue or not. If the operator enters "N" then the program executes the Quit All statement to end the program.

```
Function Main
  String cont$
  Integer i
  OnErr Goto Errhandler
  For i = 1 To 10
    Jump P(i)
  Next i
  Exit Function
' *** Error handler ***
Errhandler:
  enum = Err
  Print "Error #", enum, " occurred"
  Print "Continue (Y or N)?"
  Line Input cont$
  Select cont$
    Case "y", "Y"
      EResume Next
    Default
      Quit All
  Send
Fend
```



## 6.14 Multi-tasking

For some applications, you may want to control other equipment besides the robot, such as conveyors, pick and place units, etc. By using multi-tasking, you can control this other equipment with their own tasks.

SPEL<sup>+</sup> supports up to 32 normal tasks and 16 background tasks (48 tasks in total) running simultaneously. A task is a function that has been started by the system or by the **Xqt** statement.

Use the **Xqt** statement to start another task from within a function. You can optionally specify a task number from 1 to 32 in the **Xqt** statement.

A task started from a background task is started as a background task. You can execute up to 16 background tasks simultaneously.

The table below shows the program instructions that are used for multitasking.

Statement	Purpose
<b>Xqt</b>	Starts a function as a task.
<b>Halt</b>	Temporarily suspends execution of a task.
<b>Resume</b>	Resumes a task that has been halted.
<b>Quit</b>	Stops a task.
<b>Signal</b>	Sends a signal to one or more tasks that are waiting for the signal using <b>WaitSig</b> .
<b>SyncLock</b>	Locks a resource for use by the current task and blocks other tasks from using the resource until <b>SyncUnlock</b> is executed.
<b>WaitSig</b>	Waits for a signal from another task.
<b>Pause</b>	Pause all tasks.

One example for starting another task is to run a conveyor system for the robot work cell.

### Program: MAINTASK.PRG

```
Function Main
  Xqt Conveyor          ' Start the conveyor task
  Do
    ...
    ...
  Loop
Fend
```

### Program: CONVTASK.PRG

```
Function Conveyor
  Do
    Select True
      Case Sw(10) = On
        Off convCtrl
      Case Sw(11) = On
        On convCtrl
    Send
  Loop
Fend
```

### 6.15 Using Multiple Robots

You can control more than one robot in the same project. Use the Robot statement to switch the current robot for the current task. For most applications, you should use a separate task for each robot.

Each robot has its own set of point files. You can configure which point files to use in the Project Editor. The default point file you configure for each robot is automatically loaded into memory when the main task is started.

The following program is an example where two robots run simultaneously, each with its own task.

```
Function main
```

```
    Xqt Robot1
```

```
    Xqt Robot2
```

```
Fend
```

```
Function Robot1
```

```
    Robot 1
```

```
    Speed 50
```

```
    Do
```

```
        Jump pick
```

```
        On gripper1
```

```
        Wait .1
```

```
        Jump place
```

```
        Off gripper1
```

```
        Wait .1
```

```
    Loop
```

```
Fend
```

```
Function Robot2
```

```
    Robot 2
```

```
    Speed 50
```

```
    Do
```

```
        Jump pick
```

```
        On gripper2
```

```
        Wait .1
```

```
        Jump place
```

```
        Off gripper2
```

```
        Wait .1
```

```
    Loop
```

```
Fend
```

## 6.16 Robot Coordinate Systems

### 6.16.1 Overview

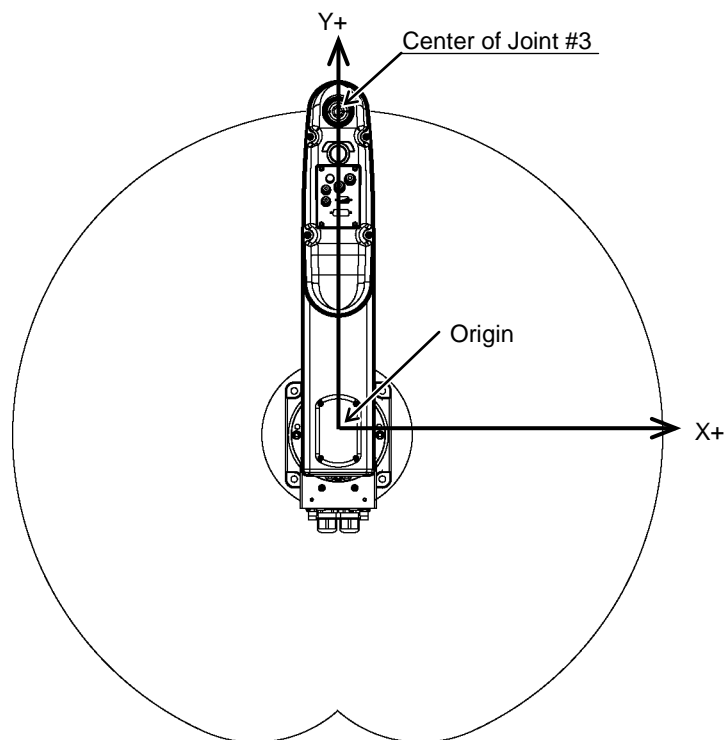
In this section we discuss the coordinate systems for different types of robots supported in SPEL<sup>+</sup>.

The following coordinate systems are used in SPEL<sup>+</sup>:

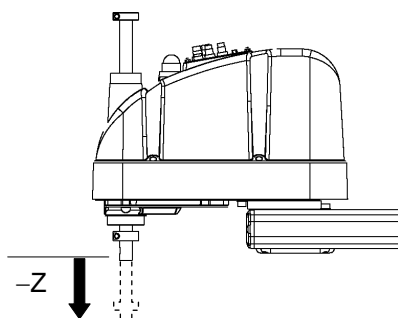
<b>Robot Coordinate System</b>	This is the native coordinate system of the robot. This is also known as the default base coordinate system.
<b>Local Coordinate System</b>	This is a user defined coordinate system located somewhere within the working envelop.
<b>Tool Coordinate System</b>	This is the coordinate system of the tool mounted on the robot end-effector.

### 6.16.2 Robot Coordinate System

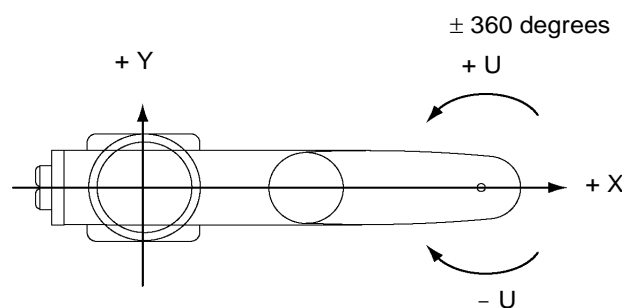
#### Robot Coordinate System of SCARA Robot



Robot coordinate system Z axis

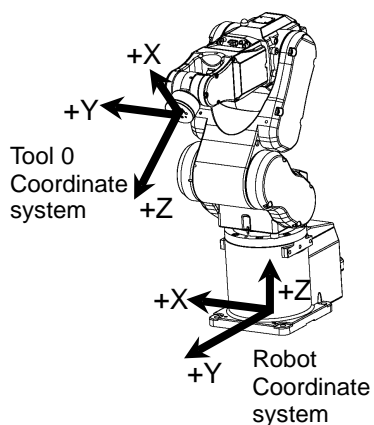


Robot coordinate system U axis

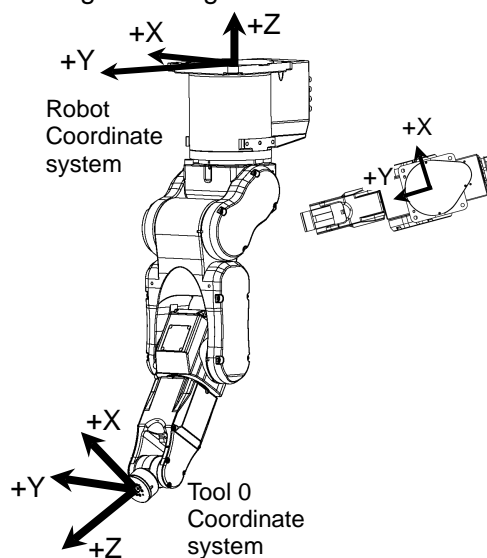


## Robot Coordinate Systems for 6-Axis Robot

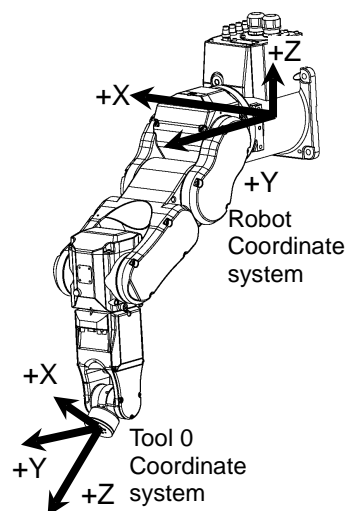
Table Top Mounting



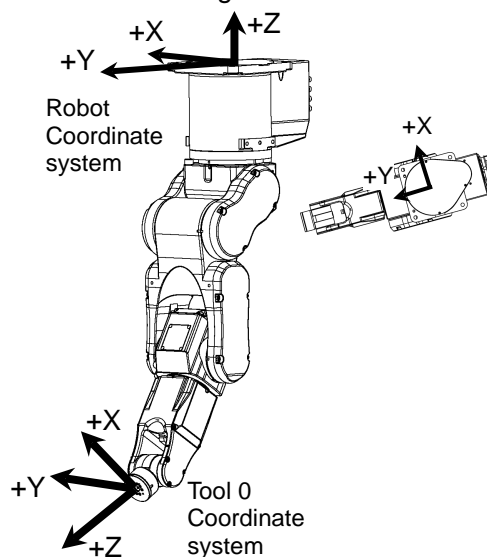
Ceiling Mounting



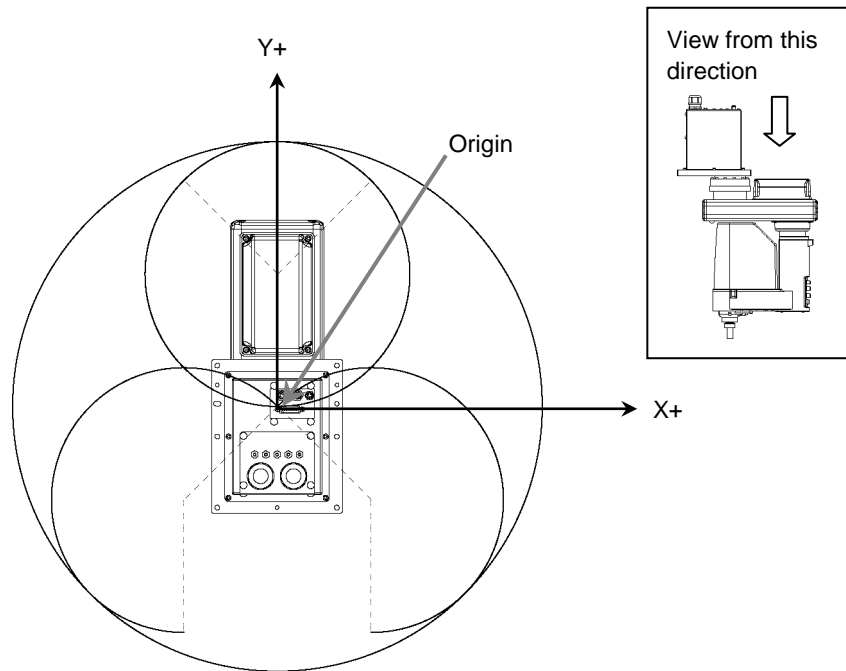
Side (Wall) Mounting



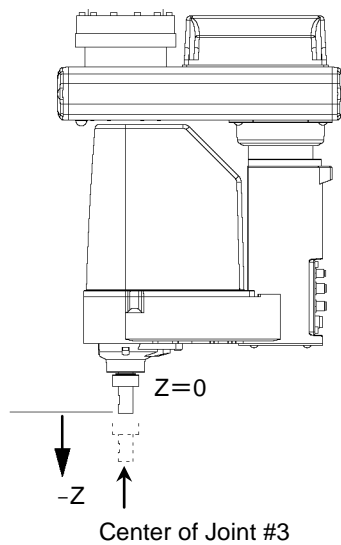
Skewed Mounting



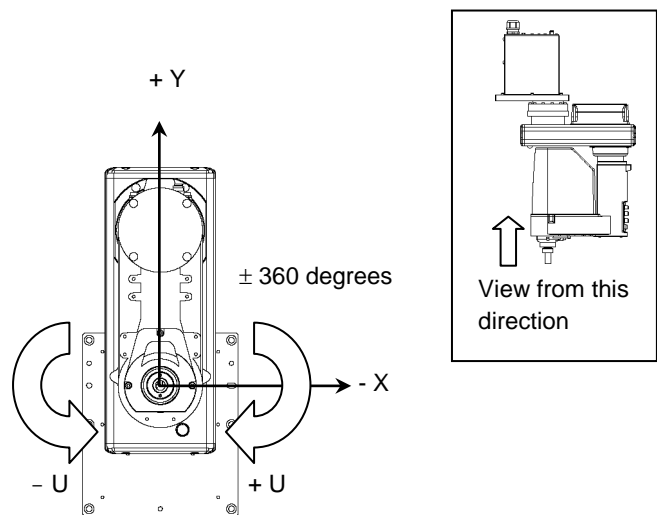
### Robot Coordinate Systems for RS series



Robot coordinate system Z axis



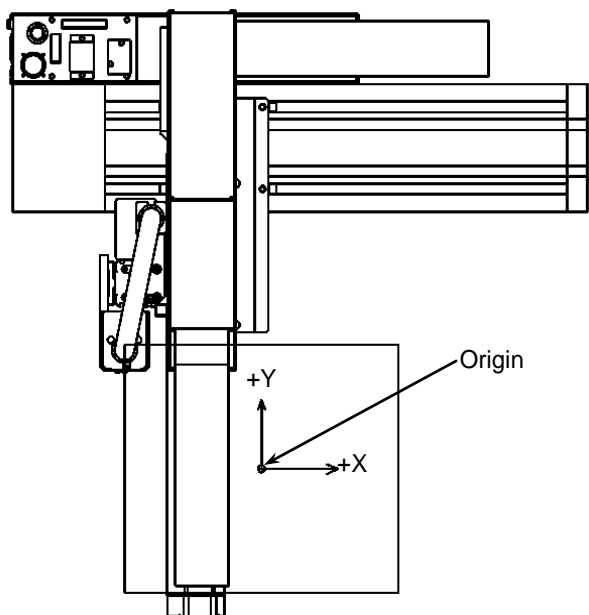
Robot coordinate system U axis



### Robot Coordinate Systems for Cartesian Robot

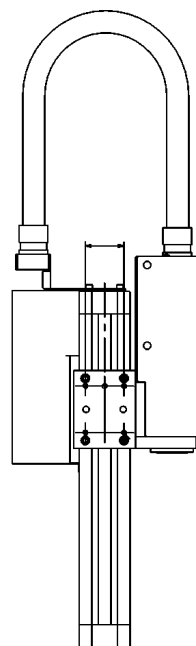
For the EZ Modules X5 series, the robot coordinate system differs depending on the robot model and mounting direction. Refer to the EZ Modules X5 series manipulator manual for details of each robot coordinate system.

*E.g. X5 series RU-HMSz A type*



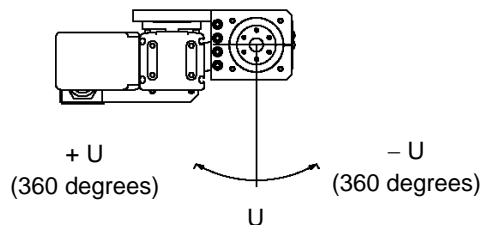
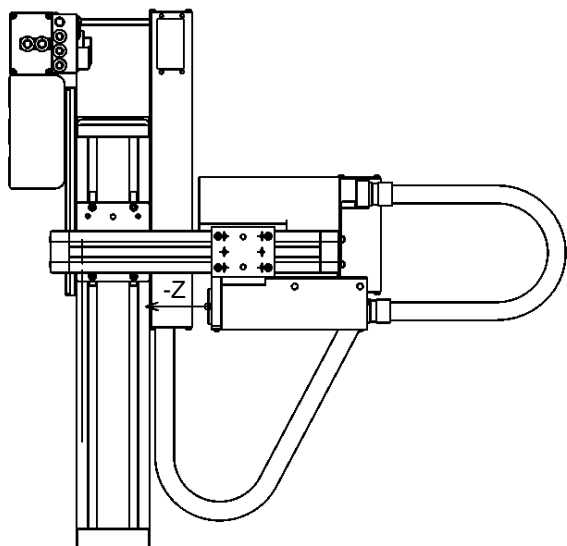
U axis in Robot Coordinate System of Cartesian Robot

*E.g. X5 series RU-HMSz A type*



Z axis in Robot Coordinate System of Cartesian Robot

*E.g. X5 series RU-HMSz A type*



### 6.16.3 Local Coordinate Systems

With SPEL<sup>+</sup>, a maximum of 15 local coordinate systems can be defined. SPEL<sup>+</sup> correlates robot coordinate systems and local coordinate systems by defining in advance the relative positional relationship of the local coordinate system from the robot coordinate system, assigning local numbers (1 to 15), and then assigning the local numbers to coordinate system attributes (local).

To define a local coordinate system, use the Local statement.

### 6.16.4 Tool Coordinate Systems

Point data is defined by the position and orientation of the tool coordinate system with respect to some reference rectangular coordinate system. The position is specified by the position data (X, Y, Z) and the orientation is specified by the orientation data (U, V, W) that correspond with roll, pitch, and yaw.

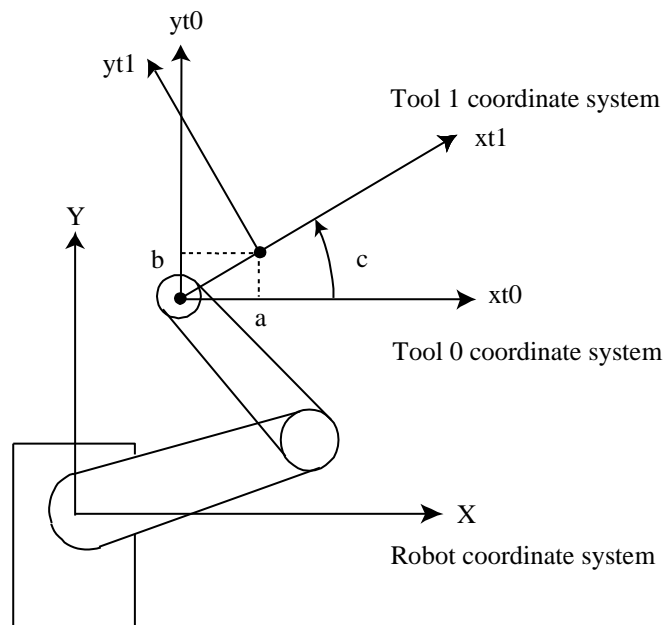
You can also define and use your own tool coordinate systems. To define the tool coordinate systems, use Tlset.

The default TOOL 0 coordinate systems are defined as follows according to the robot type.

#### SCARA Tool 0 coordinate system

The origin of tool 0 for SCARA robots is the center of the forth joint (rotation joint). When the fourth joint is adjusted to the position of 0 degrees, the tool 0 coordinate system axes are parallel to the robot coordinate system axes (see the figure below.)

The tool 0 coordinate system rotates as the fourth joint rotates.



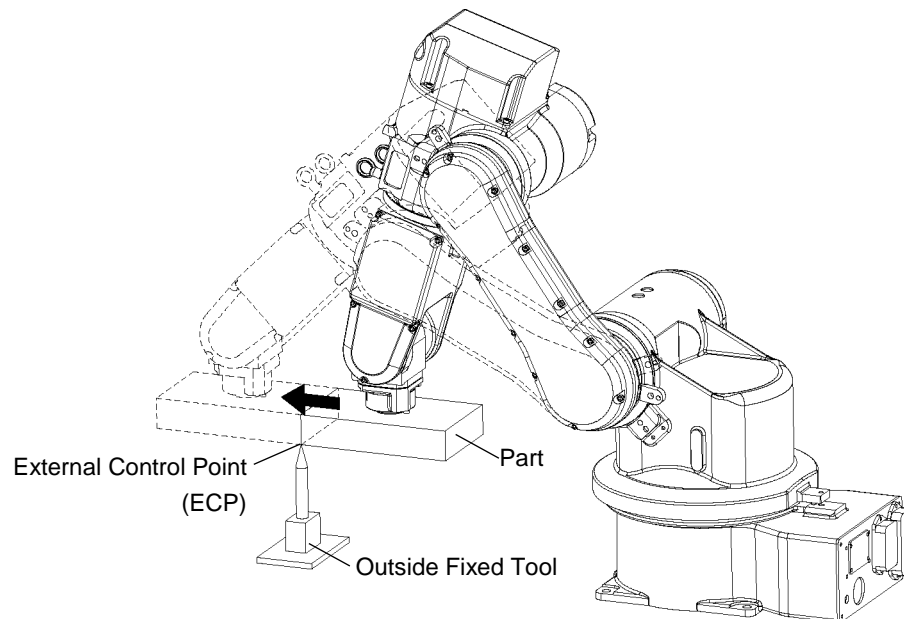
#### 6-axis Tool 0 coordinate system

For table mounting, the origin of TOOL 0 is the flange side center of the sixth joint. In TOOL 0, the tool Z axis is perpendicular to the sixth joint flange. (See the figure in the previous section *Robot Coordinate Systems*). The TOOL 0 coordinate system moves as the 6-axis robot changes its orientation.

For ceiling mounting and wall mounting robots, the TOOL 0 coordinate systems are defined as shown in the figures in the section *Robot Coordinate Systems*.

### 6.16.5 ECP Coordinate Systems (Option)

Specify a coordinate system whose origin point is on the tip of the outside fixed tool (hereafter referred to as the external control point or ECP) to move the robot arm holding a part in the trajectory made on the external control point along with the part's edges.



Use the ECPSet statement for defining an ECP coordinate system. A maximum of 15 ECP coordinate systems can be defined.

The following commands are available for optional ECP:

- Move command
- Arc3 command
- Curve and CVMove commands
- ECP jog motion

For details, refer to *16. ECP Motion*.

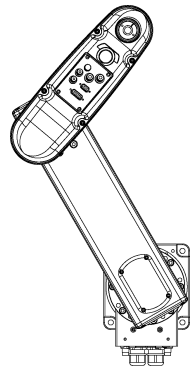


## 6.17 Robot Arm Orientations

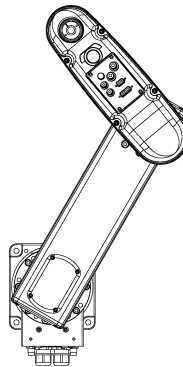
When developing a robot program, it is necessary to specify the point data taught for a particular arm orientation. If you fail to do so, the position can deviate slightly depending on the arm orientation, which in turn can cause the arm to follow an unexpected path, resulting in interference with peripheral equipment. This can be dangerous! To prevent this from happening, the orientation that the arm will be in when moved to the given point should be specified ahead of time in the point data. Such information can also be changed from the program.

### 6.17.1 SCARA robot arm orientations

With two types of arm orientation, a SCARA robot can move to nearly any position and orientation within a given work envelope. Examples are shown in the figures on the next page.



Lefty arm orientation



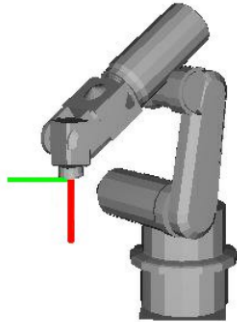
Righty arm orientation

*Examples of moving to the same point using Lefty and Righty arm orientations*

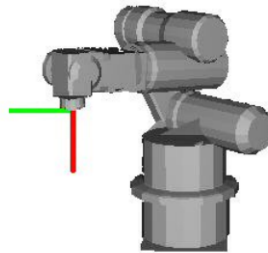
### 6.17.2 6-axis robot arm orientations

The 6-axis robot can be operated in various arm orientations within a given work envelope as shown below:

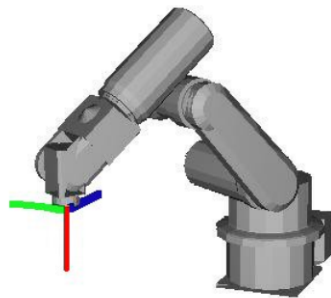
Righty hand orientation



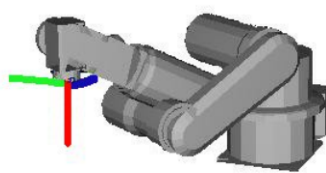
Lefty hand orientation



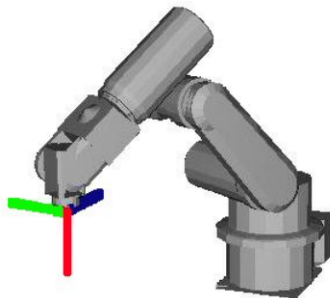
Above elbow orientation



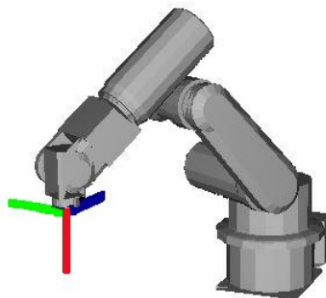
Below elbow orientation



NoFlip wrist orientation



Flip wrist orientation



To specify orientation for the 6-axis robot, add a forward slash (/) followed by L (for Lefty hand orientation) or R (Righty hand orientation), A (Above elbow orientation) or B (Below elbow orientation), and NF (NoFlip wrist orientation) or F (Flip wrist orientation).

There are eight available orientations as shown below, however, the 6-axis robot cannot be operated in all of the orientations depending on point.

Available Orientation

1	/R /A /NF	5	/R /A /F
2	/L /A /NF	6	/L /A /F
3	/R /B /NF	7	/R /B /F
4	/L /B /NF	8	/L /B /F

At some points in the work envelope, the 6-axis robot can have the same position and orientation even if the fourth joint or the sixth joint is rotated 360 degrees. To distinguish these points, the J4Flag and J6Flag point attributes are provided.

To specify the J4Flag, add a forward slash (/) followed by  
 J4F0 (-180 < the forth joint angle <= 180), or  
 J4F1 (the forth joint angle <= -180 or 180 < the forth joint angle).

To specify the J6Flag, add a forward slash (/) followed by

J6F0 (-180 < the sixth joint angle <= 180),  
 J6F1 (-360 < the sixth joint angle <= -180 or 180 < the sixth joint angle <= 360), or  
 J6Fn (-180\*(n+1) < the sixth joint angle <= 180\*n or 180\*n < the sixth joint angle <= 180\*(n+1)).

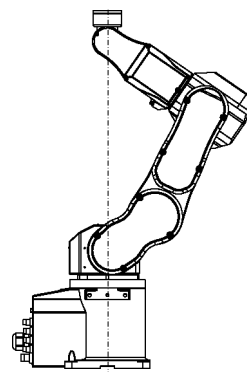
### Singularity

The orientation in the boundary where the arm orientation switches to the other

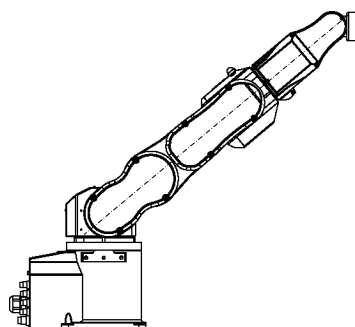
Hand singularity : The boundary where Righty hand orientation and Lefty hand orientation switch

Elbow singularity : The boundary where Above elbow orientation and Below elbow orientation switch

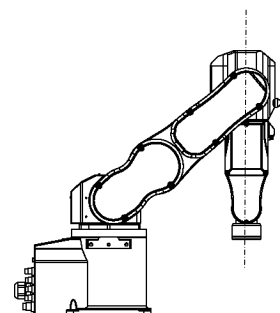
Wrist singularity : The boundary where NoFlip wrist orientation and Flip wrist orientation switch



Hand singularity



Elbow singularity



Wrist singularity

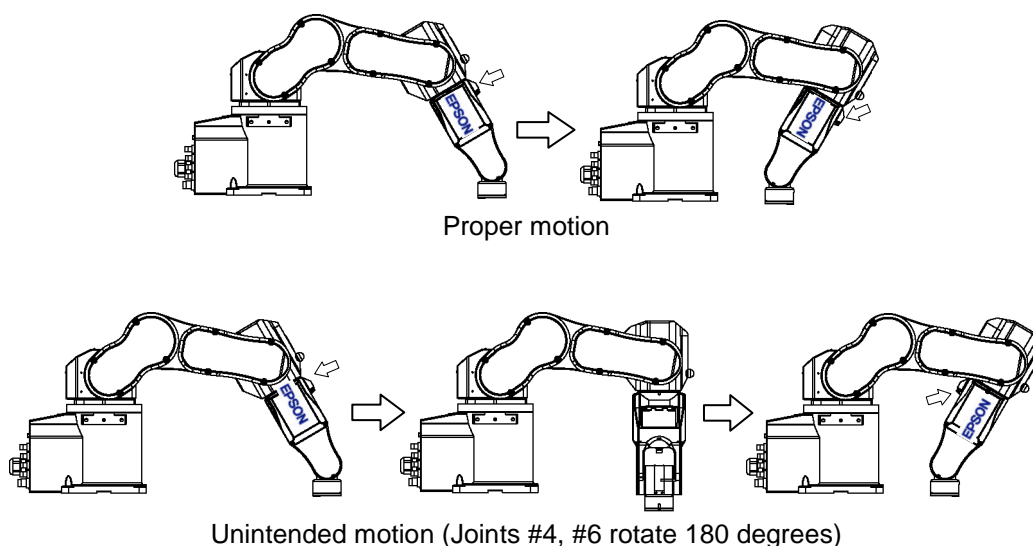
For the 6-axis robot, Hand / Wrist singularities exist also inside the motion range. When jogging near the singularity, follow the directions below.

#### PTP motion near the singularity

When jogging a robot from a point near the singularity to a point calculated by point operations such as P1+X(10), the robot may move to unintended direction because the arm orientation is not properly specified.

For example, when jogging from a point where the wrist is NoFlip to another point calculated by point operations, if the wrist keeps the NoFlip orientation while jogging, Joints #4 and #6 may rotate widely (by approx. 180 degrees).

In this case, switch to the Flip wrist orientation to jog smoothly through the wrist singularity. This phenomenon occurs not only with the point operations but also when creating points automatically with Pallet command or the result values that run from vision sequence.



However in the cases like this, it is difficult for users to specify the proper arm orientations by a program. For this LJM function is a useful command. LJM function switches the arm orientations to enable the least motion of the joints. For the details of LJM function, refer to *SPEL+ Language Reference manual*.

Also, AutoLJM command can automatically apply LJM function to the motion commands which are included in a particular section of the program without using LJM function. For details of AutoLJM command, refer to the *SPEL+ Language Reference*.

In addition, you can set AutoLJM function to be enabled at the controller start up by setting preferences of the controller. However, if Auto LJM is enabled in preferences, this function automatically adjusts the posture of the manipulator to reduce the motion distance, even when you intend to move the joint widely. Therefore, it is recommended to build a program using AutoLJM command or LJM function to operate the manipulator as you desired.

If you specify all points by teaching, the arm orientations are also recorded. Therefore, the manipulator moves to the taught position without using LJM function or AutoLJM. Instead, the manipulator may move differently from the taught position by the use of LJM and AutoLJM.

### LJM function for CP motion command

LJM function and AutoLJM command described above are also available for CP motion commands. However, since CP motion commands give priority to operate based on specified trajectories, the manipulator sometimes reach to the point with a different posture from the specified one. At this time, if CP motion command is used with CP On, an error from 4274 to 4278 will occur according to the mismatched point flag. To avoid the error, operate the manipulator with CP Off, or match the point flag of a target point and the one after motion completion. If operated with CP Off, the error does not occur and the manipulator can continue operation from the point where the mismatch happened.

Also, you can set the controller's preference so that the mismatches of flags are not considered as an error at the controller startup. However, path motions which use CP On will be disabled.

### CP motion near the singularity (singularity avoiding function in CP motion)

When executing Move or CP motion near the singularity, joint speed may increase rapidly. The over speed error will occur and the joints will move widely and interfere with peripherals. Particularly the position of Joint #1 changes greatly near the hand singularity and the position of Joints #2 - #6 near the wrist singularity.

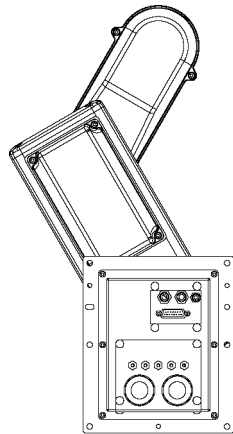
RC+6.0 has a singularity avoiding function to prevent acceleration errors during the execution of CP motion commands that pass the wrist singularity described above. With this function, the manipulator takes a detour to avoid an acceleration error by passing a different trajectory and returns to the original trajectory after passing the singularity. For details of the singularity avoiding function, refer to *AvoidSingularity* in the *SPEL+ Language Reference*.

Singularity avoiding function is enabled as default. If you want to avoid the error by reducing the motion speed in order to maintain the trajectory accuracy, you can disable the function temporarily by setting "0" to *AvoidSingularity*.

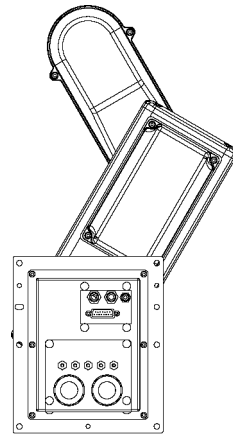
If you cannot avoid errors even if you use the singularity avoiding function, use PTP motion to enable the least motion of the joints or arrange the manipulator installation position and hand offset volume to prevent the CP motion near the singularity.

### 6.17.3 RS series arm orientations

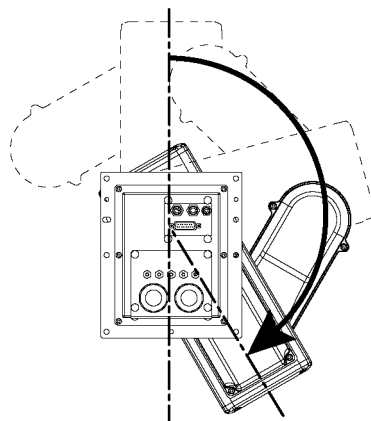
The RS series can be operated in various arm orientations within a given work envelope as shown below:



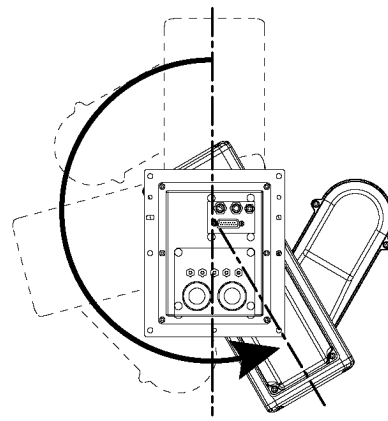
Lefty arm orientation



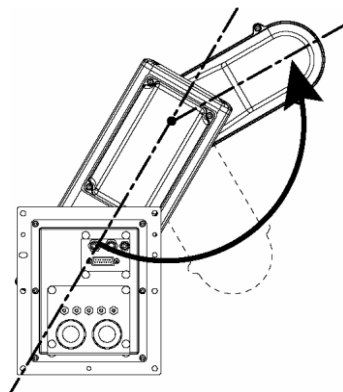
Righty arm orientation



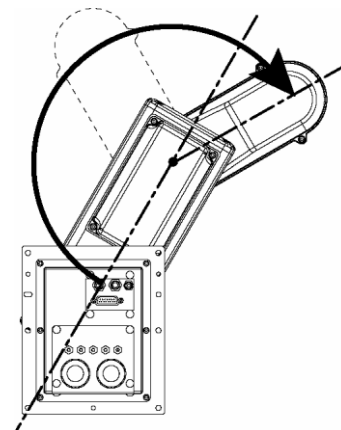
J1 F0 arm orientation



J1 F1 arm orientation



J2 F0 arm orientation



J2 F1 arm orientation

To specify orientation for the RS series, add a forward slash (/) followed by L (for Lefty hand orientation) or R (Righty hand orientation), J1F0 or J1F1, J2F0 or J2F1.

The RS series can have the same position and orientation even if the first joint or second joint are rotated 360 degrees. To distinguish these points, the J1Flag and J2Flag point attributes are provided.

To specify the J1Flag, add a forward slash (/) followed by J1F0 ( $-90 < \text{the first joint angle} \leq 270$ ) or J1F1 ( $-270 < \text{the first joint angle} \leq -90$  or  $270 < \text{the first joint angle} \leq 450$ ).

To specify the J2Flag, add a forward slash (/) followed by J2F0 ( $-180 < \text{the second joint angle} \leq 180$ ) or J2F1 ( $-360 < \text{the second joint angle} \leq -180$  or  $180 < \text{the second joint angle} \leq 360$ ).

There are eight available orientations as shown below. However, the RS series cannot be operated in all of the orientations depending on point.

	Available Orientation
1	/R /J1F0 /J2F0
2	/L /J1F0 /J2F0
3	/R /J1F1 /J2F0
4	/L /J1F1 /J2F0
5	/R /J1F0 / J2F1
6	/L /J1F0 / J2F1
7	/R /J1F1 / J2F1
8	/L /J1F1 / J2F1

## 6.18 Robot Motion Commands

SPEL<sup>+</sup> includes several commands for controlling the robot from your programs.

### 6.18.1 Homing the robot

The Home command moves the robot to a user defined "park" or "idle" position. This command works for all robots. It is mainly used for absolute encoder robots that normally do not need to be mechanically homed. Use the HomeSet command to set the home position and the Hordr command to set the home order.

### 6.18.2 Point to point motion

Point to point (PTP) commands move the robot from its current position to a specified point. Motion may not be in a straight line.

To set the speed for point to point commands, use the Speed command. To set acceleration and deceleration, use the Accel command.

Command	Description
<b>Go</b>	Move directly to a point using point to point motion.
<b>Jump</b>	Jump to a point. First move up to the current LimZ setting, the move over the destination point, then move to the point. The Arch table settings determine the Jump profile.
<b>Jump3</b>	Jump to a point in 3 dimensions.
<b>Pass</b>	Move near one or more points.
<b>TGo</b>	Move directly to a point in a tool coordinate system.

### 6.18.3 Linear motion

Linear motion commands move the robot from its current position to a specified point in a *straight line*. Linear motion is a CP (Continuous Path) motion.

To set velocity (speed) for straight motion, use the SpeedS command. To set acceleration and deceleration, use the AccelS command.

Command	Description
<b>Move</b>	Move in a straight line to the specified point.
<b>TMove</b>	Move in a straight line to the specified point in a tool coordinate system.
<b>Jump3CP</b>	Jump to a point in 3 dimensions using CP motion.

### 6.18.4 Curves

Curves commands move the robot in a circular arc. Curves is a CP (Continuous Path) motion.

To set velocity (speed) for Curves, use the SpeedS command. To set acceleration and deceleration, use the AccelS command.

Command	Description
<b>Arc</b>	Move the robot through one point to another point using circular interpolation.
<b>Arc3</b>	Move the robot in 3D using circular interpolation.
<b>Curve</b>	Creates a file containing a path specification.
<b>CVMove</b>	Executes a path specified by Curve.



### 6.18.5 Joint motion

Command	Description
<b>JTran</b>	The JTran command can be used to move one joint of the robot to a position specified in degrees or millimeters, depending on the joint type. The speed and acceleration are the same as for point to point motion commands -- i.e., specified with Speed or Accel commands.
<b>PTran</b>	The PTran command can be used to move one joint of the robot to an encoder pulse position. The speed and acceleration are the same as for point to point motion commands -- i.e., specified with Speed or Accel commands.
<b>Pulse</b>	The Pulse command can be used to move all joints of the robot to encoder pulse positions. The speed and acceleration are the same as for point to point motion commands -- i.e., specified with Speed or Accel commands.
<b>PG_Scan</b>	The PG_Scan command can be used to rotate a pulse generator axis of a Joint-type single axis PG robot continuously in CW/CCW directions. (To rotate it continuously, you need to enable the continuous rotation parameter.) The speed and acceleration are the same as for point to point motion commands -- i.e., specified with Speed or Accel commands.

### 6.18.6 Controlling position accuracy

Use the Fine command to adjust position accuracy for the end of a motion command. Fine specifies, for each joint, the allowable positioning error for detecting completion of any given move. The lower the Fine settings, the more accurate the final position of the joint, which can cause slower performance. Conversely, large Fine settings can speed up motion commands, but position accuracy will decrease. For many applications, the default settings can be used.

### 6.18.7 CP Motion Speed / Acceleration and Tool Orientation

When you attempt to change only the tool orientation while keeping the tool tip of the robot arm at the specified coordinate point or when the tool orientation variation is larger than the travel distance of the tool tip, moving the arm by normal CP motion commands will cause an increase in the variation of speed, acceleration and deceleration of tool orientation. In some cases, an error will occur.

To prevent these situations, add the ROT parameter to the CP motion commands. The arm will be moved based on the specified angular velocity and acceleration/deceleration of the main axis regarding the orientation variation.

The angular velocity and acceleration/deceleration of the main axis regarding the orientation variation should be specified with the SpeedR and AccelR commands in advance.

For example:

```
SpeedR 50 ' degree/sec
AccelR 200, 200 ' degree/sec2
Move P1 ROT
```



The tool orientation variation is normally comprised of orientation variations of more than one rotation axis.

The SpeedR and AccelR parameters specify the angular velocity and acceleration/deceleration of the main axis regarding the orientation variation. Therefore, actual angular velocity and acceleration/deceleration of the orientation variation are different from the parameters except for the case where the rotation axis of the orientation is only one.

While the motion command with the ROT parameter is executed, the specified SpeedS and AccelS parameters are invalid.

The ROT parameter can be used with the following motion commands:

```
Move    BMove
Arc      TMove
Arc3     Jump3CP
```

### 6.18.8 PTP Speed / Acceleration for Small Distances

You can change the speed and acceleration for small distances using PTPBoost and PTPBoostOK. Normally, PTPBoost is not required. In certain cases, you may want to shorten the cycle time even if vibration becomes larger, or conversely you may want to reduce vibration even if cycle time becomes longer. PTPBoost is a robot parameter with values from 0 – 100 that affects the speed and acceleration for small distances. Normally, for small distance motion, the desired speed cannot be attained using the current acceleration. By increasing PTPBoost, acceleration, deceleration, and speed are increased for small distance motion. To check if a motion command will be affected by PTPBoost, use the PTPBoostOK function. See PTPBoost and PTPBoostOK in the *SPEL+ Language Reference* manual for more details.

## 6.19 Working with Robot Points

A robot point is a set of coordinates that define a position in the robot work envelope. For SCARA and Cartesian robots, a point is defined by the position data (X, Y, Z) within the reference rectangular coordinate space and the orientation data (U) which is the rotation about the Z axis of the rectangular coordinate.

For 6-axis robots, a point is defined by the position and orientation of the tool coordinate system with respect to a reference rectangular coordinate system. The point is specified by the position data (X, Y, Z) and the orientation is specified by the orientation data (U, V, W) which correspond with *roll* (rotation about the Z axis), *pitch* (rotation about the Y axis), and *yaw* (rotation about the X axis).

When the additional ST axis is installed, the point is specified by the position data of each additional axis (S, T).

The X, Y, and Z coordinates of a point are specified in millimeters. The U, V, and W coordinates are specified in degrees.

The S and T coordinates of a point are specified in millimeters or degrees, according to the type of axis.

Points are referenced using the letter P followed by an integer number or integer expression or by a label defined in the point file editor or Robot Manager Jog & Teach page.

### 6.19.1 Defining points

You can define points in a program statement, points editor window, Robot Manager Jog and Teach page, or at the Command window.

In a program statement or at the Command window, you can assign coordinates to a point, or define a point that is the current robot arm position.

```
P1 = XY(200, 100, -25, 0)      ' Assign coordinates to point P1
Pick = XY(300, 200, -45, 0)   ' Assign coordinates to point pick
P10 = Here                     ' Assign a point to current position
```

### 6.19.2 Referencing points by point label

You can assign names to point numbers so you can refer to points by name in a program. Assign names from the point editor (see Editing Points) or the Robot Manager Jog and Teach page. Names must be unique for each point number when used in the same point file.

Point labels can include up to 16 alphanumeric, Japanese, and the underscore characters. Characters can be upper case or lower case. Only alphabets and Japanese can be used for the first letter.

```
For i = 0 To 10
  Go pick
  Jump place
Next i
```

### 6.19.3 Referencing points with variables

Use the letter P followed by a variable name within parentheses that represents the point number you are referencing.

```
For i = 0 To 10
  Go P(i)
Next i
```



Although you can define points at the Command window for test purposes, it is recommended that all points be defined in a program, point editor, or with the Robot Manager Jog and Teach page. Points defined at the Command window will be cleared from memory when you build a project or run a program unless you execute SavePoints.

### 6.19.4 Using points in a program

When starting programs, the default point file for the robot is loaded. You can also load other points in the program using the LoadPoints statement.

```
Function main
  Integer i

  LoadPoints "modell.pts"
  For i = 0 To 10
    Jump pick
    Jump place
  Next i
Fend
```

### 6.19.5 Importing points into program

You can import points into the current project while the program is running using the ImportPoints statement.

```
Function main
  Integer i

  ImportPoints "c:\models\modell.pnt", "robot1.pnt"
  LoadPoints "robot1.pnt"
  For i = 0 To 10
    Jump pick
    Jump place
  Next i
Fend
```

### 6.19.6 Saving and loading points

Use LoadPoints to load a point file in the current project. You can optionally specify the **Merge** parameter to combine points in a file with points that have already been loaded.

Use SavePoints to save the points in a point file. If the point file is in the current project, it will be updated on the PC when it is connected and the same project is open.

If the point file is not the current project, it will not be automatically updated on the PC. Use Project Synchronize to copy the file to the PC if desired.

### 6.19.7 Point attributes

Each point definition can optionally specify a local number and various arm orientations, depending on the robot type. You can specify point attributes in point assignment statements or use individual statements and functions to change the attributes of a previously defined point.

#### Local point attribute

To specify a local coordinate system number for a point in an assignment statement, add a forward slash (/) followed by the local number after the coordinates of the point.

```
P1 = XY(300, -125.54, -42.3, 0) /1 ' P1 is in local 1
```

The local number can also be an expression enclosed in parentheses.

```
P2 = P3 /(mylocal)
```

Use the PLocal function and statement to read and set the local attribute of a point.

#### Hand point attribute

To specify orientation for the SCARA or 6-axis robot, add a forward slash (/) followed by L (for Lefty hand orientation) or R (for Righty hand orientation).

```
P2 = XY(200, 100, -20, -45) /L ' Hand orientation is Lefty
```

```
P3 = XY(50, 0, 0, 0) /2 /R ' Righty in Local 2
```

You can read and set point hand orientation using the Hand statement and function.

```
Hand P1, Righty
```

#### Elbow point attribute

To specify elbow orientation for the 6-axis robot in a point assignment statement, add a forward slash (/) followed by A (Above elbow orientation) or B (Below elbow orientation),

Elbow orientation is Below.

```
P1 = XY(0, 600, 400, 90, 0, 180) /B
```

You can read and set point elbow orientation using the Elbow statement and function.

#### Wrist point attribute

To specify wrist orientation for the 6-axis robot in a point assignment statement, add a forward slash (/) followed by NF (NoFlip wrist orientation) or F (Flip wrist orientation).

Wrist orientation is Flip.

```
P2 = XY(0, 600, 400, 90, 0, 180) /F
```

You can read and set point wrist orientation using the Wrist statement and function.

#### J4Flag and J6Flag point attributes

At some points in the work envelope, the 6-axis robot can have the same position and orientation even if the fourth joint or the sixth joint is rotated 360 degrees. To distinguish these points, the J4Flag and J6Flag point attributes are provided. These flags allow you to specify a position range for joint 4 and joint 6 for a given point.

To specify the J4Flag in a point assignment statement, add a forward slash (/) followed by J4F0 ( $-180 < \text{the forth joint angle} \leq 180$ ) or J4F1 ( $\text{the forth joint angle} \leq -180$  or  $180 < \text{the forth joint angle}$ ).

```
P2 = XY(0, 600, 400, 90, 0, 180) /J4F1
```

To specify the J6Flag in a point assignment statement, add a forward slash (/) followed by J6F0 ( $-180 < \text{the sixth joint angle} \leq 180$ ), J6F1 ( $-360 < \text{the sixth joint angle} \leq -180$  or  $180 < \text{the sixth joint angle} \leq 360$ ), or J6Fn ( $-180*(n+1) < \text{the sixth joint angle} \leq 180*n$  or  $180*n < \text{the sixth joint angle} \leq 180*(n+1)$ ).

```
P2 = XY(50, 400, 400, 90, 0, 180) /J6F2
```

**J1Flag and J2Flag point attributes**

At some points in the work envelope, the RS series can have the same position and orientation even if the first joint or the second joint is rotated 360 degrees. To distinguish these points, the J1Flag and J2Flag point attributes are provided. These flags allow you to specify a position range for joint 1 and joint 2 for a given point.

To specify the J1Flag in a point assignment statement, add a forward slash (/) followed by J1F0 ( $-90 < \text{the first joint angle} \leq 270$ ) or J1F1 ( $-270 \leq \text{the first joint angle} \leq -90$  or  $270 < \text{the first joint angle} \leq 450$ ).

```
P2 = XY(-175, -175, 0, 90) /J1F1
```

To specify the J2Flag in a point assignment statement, add a forward slash (/) followed by J2F0 ( $-180 < \text{the second joint angle} \leq 180$ ), J2F1 ( $-360 < \text{the second joint angle} \leq -180$  or  $180 < \text{the second joint angle} \leq 360$ ).

```
P2 = XY(300, 175, 40, 90) /J2F1
```

**J1Ang and J2Flag point attributes**

At the origin of the robot coordinate system, the RS series can have the same position and orientation even if the first joint is rotated. To distinguish these points, the J1Ang point attributes are provided.

**6.19.8 Extracting and setting point coordinates**

Use the CX, CY, CZ, CU, CV, CW, CS, and CT commands to get a coordinate of a point or set it.

```
xcoord = CX(P1)
P2 = XY(xcoord, 200, -20, 0)
ycoord = CY(P*)           ' Gets current Y position coordinate

CX(pick) = 25.5
CY(pick) = CY(pick) + 2.3
```

**6.19.9 Alteration of points**

There are several ways of modifying a point without re-teaching it. You can change one or more coordinate values with relative offsets or absolute values.

To set an absolute value for a coordinate, use a colon followed by the axis letter and the value.

To add a relative offset to a coordinate, use an axis letter followed by the offset value or expression in parentheses. If the offset is negative, then precede the axis letter with the minus sign. If parentheses are omitted, they will be automatically added.

Go P1 -Z(20)	Move to P1 with a z offset of -20mm
Go P1 :Z(-25)	Move to P1 with a z absolute position of -25mm
Go P1 -X(20) +Y(50) :Z(-25)	Move to P1 with offsets for X and Y relative offsets and an absolute position for Z

## 6.20 Input and output control

### 6.20.1 Hardware I/O

There are 24 DC inputs and 16 DC outputs on a standard controller. By purchasing I/O boards, you can add additional 128 inputs and 128 outputs. You can also expand the I/O by using the Fieldbus I/O master option and Fieldbus I/O slave option. Refer to *10. Inputs and Outputs* for details.

### 6.20.2 Memory I/O

There are 128 bytes (1024 bits) of memory I/O. Memory I/O is especially useful for synchronizing multi-tasking. Each memory bit can be treated as both an input and an output.

Use the commands with the "Mem" prefix for memory I/O.

### 6.20.3 I/O Commands

Command	Description
<b>In</b>	Reads one byte (eight bits) of input data.
<b>InW</b>	Reads one word (sixteen bits) of input data.
<b>MemIn</b>	Reads one byte (eight bits) of Memory I/O.
<b>MemInW</b>	Reads one word (sixteen bits) of Memory I/O.
<b>MemOff</b>	Turns off one Memory I/O bit.
<b>MemOn</b>	Turns on one Memory I/O bit.
<b>MemSw</b>	Read status of one bit of memory I/O.
<b>Off</b>	Turns off one output bit.
<b>On</b>	Turns on one output bit.
<b>Out</b>	Sets/reads one byte (eight bits) of output data.
<b>OutW</b>	Sets/reads one word (sixteen bits) of output data.
<b>Oport</b>	Reads the status of one output bit.
<b>InBCD</b>	Reads one byte of input data in BCD (binary coded decimal) format.
<b>OpBCD</b>	Outputs one byte of output data in BCD format.
<b>Sw</b>	Read status of one bit of hardware inputs or memory inputs.

## 6.21 Using Traps

Traps enable a program to jump to a label or enable a function to be called when a certain event occurs.

Traps are divided into the following two types:

- 4 Traps are fired by user defined input
- 7 Traps are fired by system

You should keep trap functions short and avoid continuous loops. According to the type, some Traps must be re-armed. Also, some motion commands are limited to execute in trap functions.

For details on Trap statement, see the *SPEL+ Language Reference manual*.

Here is a simple example for a trap. In this example, when input 1 turns on, it executes the SwlTrap function.

```
Function main
  ' Arm the trap
  Trap 1 Sw(1) = On Xqt SwlTrap
Do
  RunCycle
Loop
Fend
Function SwlTrap
  ' Turn on output 1 for 2 seconds
  On 1, 2
  ' Wait for trap condition to clear
  Wait Sw(1) = Off
  ' Re-arm the trap
  Trap 1 Sw(1) = On Xqt SwlTrap
Fend
```

Trap	Description
<b>Trap 1 – 4 Goto</b>	Triggered by an input condition specified by the user.
<b>Trap 1 – 4 Call</b>	User traps can use GoTo, Call, or Xqt.
<b>Trap 1 – 4 Xqt</b>	
<b>Trap Emergency Xqt</b>	When Emergency Stop occurs, a specified function is executed.
<b>Trap Error Xqt</b>	When an error occurs, a specified function is executed.
<b>Trap SgOpen Xqt</b>	When the Safeguard circuit is open, a specified function is executed.
<b>Trap SgClose Xqt</b>	When the Safeguard circuit is closed, a specified function is executed.
<b>Trap Pause Xqt</b>	When the system enters the Pause state, a specified function is executed.
<b>Trap Abort Xqt</b>	When all tasks (except background tasks) have been stopped by user or system, such as when a command corresponding to Abort All is executed, a specified function is executed.
<b>Trap Finish Xqt</b>	When all tasks (except background tasks) have been finished, a specified function is executed. However, the function will not be executed under the condition that executes Trap Abort.



## 6.21.1 Cautions of Trap when it triggers the system condition

**Forced Flag**

Specify Forced flag in the I/O output commands such as On/Off command to enable On/ Off of the I/O outputs during Emergency Stop, Safety Door open, Teach mode, and error condition.

Do not connect external equipment that operates mechanically such as actuator to the I/O output that specifies Forced flag. Otherwise, the external equipment may move during Emergency Stop, Safety Door Open, Teach mode, or error condition and this will cause serious safety problems.

Forced flag is designed to be specified for I/O outputs connected to external equipment without mechanical motion such as status display LEDs.

**Outputs off during Emergency Stop**

Uncheck **Outputs off during Emergency Stop** in the Preferences page of System Configuration SPEL Controller Board to execute I/O On/Off using the Trap Emergency Xqt task after Emergency Stop. If this checkbox is checked, the execution order of turning Off by the controller and turning On using the task are not guaranteed.

## 6.22 Special Tasks

Each task of SPEL<sup>+</sup> pauses by Pause input or Safety Door open and stops by Emergency Stop or Error. Therefore you cannot create a system that monitors the whole system.

To enable the RC620 controller to monitor the whole system, the following special tasks are provided:

**NoPause/NoEmgAbort task**

You can create a task that continues a processing even when the Pause is input or safeguard is open by specifying NoPause or NoEmgAbort as a task type when creating Xqt data task.

**Background task**

You can create a task that starts as the controller power is turned ON and continues a processing even when the Pause is input or safeguard is open.

These special tasks are useful tasks but may reduce the safety of the system by using them improperly.

Be sure to understand the following items when using these tasks.

## 6.22.1 Precautions to Use the Special Tasks

**Forced Flag**

Specify Forced flag in the I/O output commands such as On/Off command to enable On/ Off of the I/O outputs during Emergency Stop, Safety Door open, and error.

Do not connect external equipment that operates mechanically such as actuator to the I/O output that specifies Forced flag. Connecting external equipment may cause serious safety problems and operate the external equipment during Emergency Stop, Safety Door Open, or error occurrence.

Forced flag is designed to be specified for I/O outputs connected to external equipment without mechanical motion such as status display LEDs.

**NoEmgAbort Task**

When Emergency Stop or errors occur, finish the task promptly after completing the error handling.

If you do not complete the NoEmgAbort task, the controller does not change to Ready status and you cannot cancel the Emergency Stop or the error. You cannot execute Reset command from the NoEmgAbort task to cancel the Emergency Stop or the error automatically.

NoEmgAbort task is designed for I/O process without motion and communication with external device using the Ethernet. Therefore there are commands such as robot motion commands that cannot be executed in the NoEmgAbort task. An error occurs if you use these commands. The list of these commands is in the next section.

For details, refer to EPSON RC+ 6.0 *Online Help* or *Xqt* in *SPEL+ Language Reference*.

**NoPause Task**

NoPause task continues the operation during the Pause or Safety Door open condition. However, when a robot is operating NoPause task, the task pauses as the robot pauses.

**Background task**

Background task always exists while the controller is working, and it is designed for monitor of the entire system and communication with external device. Therefore there are commands such as robot motion commands that cannot be executed in the background task. An error occurs if you use these commands. The list of these commands is in the next section.

In addition, the background task continues processing even when Pause is input or safeguard is open, so it doesn't affect the controller state transition.

For details, refer to 6.23 *Background Task*.

**Outputs off during Emergency Stop**

Uncheck this preference to execute I/O On/Off using the NoEmgAbort task or background task after Emergency Stop. If this check box is checked, the execution order of turn Off by the controller and turn On using the task are not guaranteed.

**Setting of Safeguard open stops all tasks**

When this preference is checked, NoPause task stops by Safety Door open. NoEmgAbort task or background task continues the task.

**Setting of [Enable the Background task]**

Set this preference when you use the background task.

**Setting of [Initialize global variables as the MainXX starts]**

Uncheck this preference when you use the global variables from the background task. When this check box is checked, the controller will initialize the variables and the variable-access conflict from tasks will occur.



CAUTION

**Setting of [Enable advanced task commands]**

Check this preference when you execute the commands below from a background task.

StartMain, Cont, Recover, Reset Error

When you execute these commands from a task, you should understand each command specification and verify that the system has the appropriate conditions.

Improper use, such as executing commands continuously in a loop, can reduce the security of system.

## 6.22.2 NoPause/NoEmgAbort task specification

## Status by Event and Task

Event	Task Type		
	Normal	NoPause	NoEmgAbort
Pause Statement Pause Input Pause Button	Pause	Continue *1	Continue
Safety Door Open	Pause *2	Continue *1 *2	Continue
Error during Auto Mode	Stop	Stop	Continue
Error during Program Mode	Pause	Pause	Continue
Emergency Stop	Stop	Stop	Continue
Stop Button Stop Input	Stop	Stop	Stop
Halt Statement Halt Button	Pause	Pause	Pause
Brake Point	Pause	Pause	Pause
Switching to Teach Mode	Stop	Stop	Stop

\*1 When the robot is operating, the task pauses as the robot pauses.

\*2 When **Outputs off during Emergency Stop** is checked in the Preferences page of Setup Controller, normal tasks and NoPause tasks stop by Safety Door open.

## Task Execution

Normal	Omit the task type in Xqt statement, or specify Normal for the task type. Xqt NormalTask Xqt NormalTask, Normal
NoPause	Specify NoPause in Xqt statement. Xqt NoPauseTask, NoPause
NoEmgAbort	Specify NoEmgAbort in Xqt statement. Xqt NoEmgAbortTask, NoEmgAbort

You cannot change the task type after executing a task.

main to main63 that are executed at the beginning of the program are executed as normal tasks.

Type of a task executed in Trap Xqt is determined by the event type.

For details, refer to EPSON RC+ 6.0 *Online Help* or *Trap* in *SPEL<sup>+</sup> Language Reference*.

## Restricted Commands by Task Types

Normal	No restriction
NoPause	No restriction
NoEmgAbort	Cannot execute the following commands. Command for robot motion Commands for vision Reset, Xqt, Trap, etc. For details, refer to EPSON RC+ 6.0 <i>Online Help</i> or <i>Xqt</i> in <i>SPEL<sup>+</sup> Language Reference</i> .

### 6.22.3 NoPause/NoEmgAbort task example

The following example shows a program that monitors the error of the controller and switches the I/O On/Off when error occurs according to the error number.

The program example of ErrOn, EStopOn, SafetyOn are indicated in the *EPSON RC+ 6.0 SPEL+ Language Reference*.

```
Function main
    Xqt ErrorMonitor, NoEmgAbort
    :
    :
Fend

Function ErrorMonitor
    Wait ErrorOn
    If 4000 < SysErr And Syserr < 5999 Then
        Print "Mortion Error = ", SysErr
        Off 10, Forced
        On 12, Forced
    Else
        Print "Other Error = ", SysErr
        Off 11, Forced
        On 13, Forced
    EndIf
Fend
```

## 6.23 Background Task

### 6.23.1 Primary features of background task

The purpose of the background task is to monitor the status of the cell as a whole and to communicate with external devices.

Function BgMain, a function specified as the “Background task” will be automatically activated as task 65 when the controller starts and loads the project.

If another task is created within the background task using the XQT command, that created task will be assigned to task No.65 (and onward in the ascending order) and will also function as a background task. In addition, specifying a task type for an XQT command in a background task has no meaning.

An operator is not necessarily aware of the operating Background task which does not stop at the input of emergency stop or safeguard signal. The Background task will not stop when an operator inputs “PAUSE” or “ABORT”.

In this sense, the background task functions for the application program to work as a part of the system.

On the other hand, the execution commands to operate the Manipulator, set-up commands for the Manipulator or the commands for image processing cannot be executed within the background task.



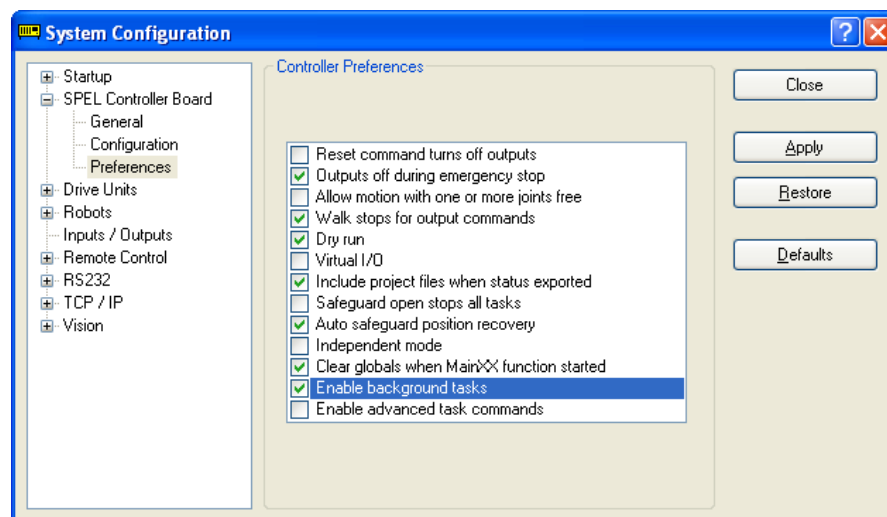
- Specify Forced flag in the I/O output commands operated from the background tasks to enable On/ Off of the I/O outputs during Emergency Stop, Safety Door open, and error.

Do not connect external equipment that operates mechanically such as actuators to the I/O output that specifies the Forced flag. Connecting external equipment may cause serious safety problems and operate the external equipment during Emergency Stop, Safety Door Open, or error occurrence.

Forced flag is designed to be specified for I/O outputs connected to external equipment without mechanical motion such as status display LEDs.

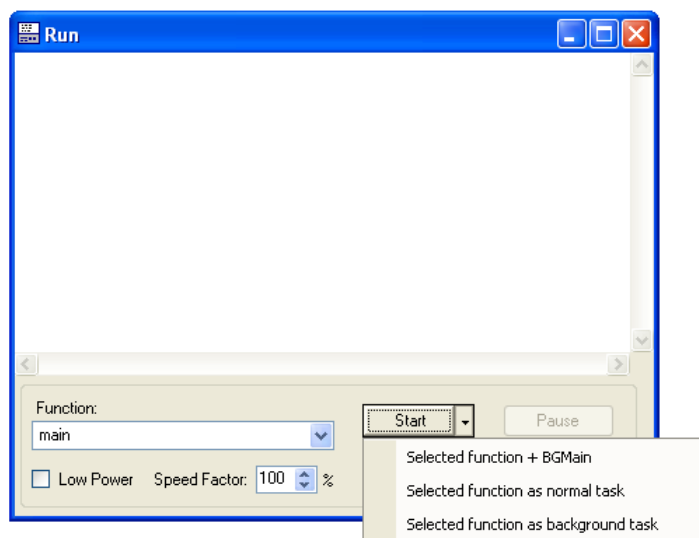
### 6.23.2 Setup and start the background task

When you use the background task, first of all you need to check the [Enable background tasks] in the Preferences page of Setup | System Configuration | SPEL Controller Board.



When you have already checked the box above and the Function BgMain exists in your program, it will automatically start as Task 65 as the controller starts and loads the project, it executes as a “Background task”.

However in PROGRAM mode, the Function BgMain will not start automatically. You need to start it using the **Start** button in the [Run] window. This is because the PROGRAM mode is for creating programs and debugging and it may be more efficient when it doesn't start the Function BgMain.

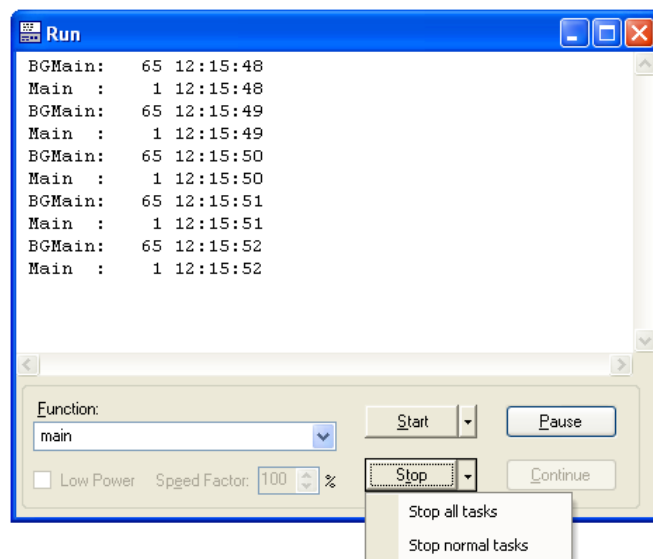


When the controller operating mode shifts from PROGRAM to AUTO mode, the Function BgMain will start automatically.

### 6.23.3 Holding background task (from being activated)

The purpose of the background task is to monitor the status of the cell as a whole and to communicate with external devices. It is activated before a non-background task is activated and continues to function when the non-background task either generates an error or is aborted by an operator. In this sense, the background task can be a program that never stops functioning.

The background task can be debugged in PROGRAM mode. Click the **Stop** button dropdown menu in the [Run] window and you can select the background task is to be aborted as well or not.



In the [Task Manager] window, the background tasks can be managed in the same way as the non-background tasks except for the <Pause/Cont> button. You can set a break point in a background task and step through the code.

As a rule, the background task cannot be controlled in AUTO mode. It is by design that any error that occurs in the background task cannot be recovered in AUTO mode. Therefore, thorough debugging in PROGRAM mode is recommended. Be particularly careful that the communication errors are handled properly without fail before using the background task in AUTO mode.

The following tables show how the background will (or will not) be affected by operation from the console.

#### Operator Window

Button	Background task
START	It will not be affected.
Abort	It will not be affected.
Pause	It will not be affected.
Continue	It will not be affected.

#### Remote Input

Button	Background task
Start / Stop	It will not be affected.
Pause / Continue	It will not be affected.
Reset	It will not be affected.
Shutdown	It will be stopped.

#### Run Window (PROGRAM mode)

Button	Background task
Start	You can select how to start the task.
Abort	You can select how to abort the task: abort only non-background task or abort all tasks including the background task.
Pause	It will not be affected.
Continue	It will not be affected.

#### Task Manager (PROGRAM mode)

Button	Background task
Halt / Resume	When the background task is selected, you cannot execute Halt/Resume.
Quit	When the background task is selected, you can execute Quit.
Pause/Cont	It will not be affected.
Stop	All tasks including the background task will stop.

#### Break point (PROGRAM mode)

Switch name	Background task
Set a break point	You can set a breakpoint to the background task. It will pause at the break point.
Step Into	Available
Step Over	Available
Continue	Available
Walk	Available, but the motion commands are not available to execute from the background task.

### 6.23.4 Commands that will cause error in background task

The following commands are prohibited in background tasks and execution will result in error:

- commands that relate to the Manipulator operation or operation settings
- commands that relate to the Vision relation instruction
- TRAP commands

If a program that is to be executed as the background task includes any of the following commands, it will result in error when executed.

However, using the command related to the Manipulator operation settings or the Manipulator settings to gain the current setting values or refer to them will not result in error:

Commands that will cause error are almost the same as with NoEmgAbort, but there are some commands such as Xqt that can be executed in a background task.

For details, refer to EPSON RC+ 6.0 *Online Help* or *Xqt* in *SPEL+ Language Reference*.

### 6.23.5 Background task and Remote control

No matter whether the background task is being executed or not, it doesn't affect the remote I/O outputs Ready, Running, and Pause. For example, even if the background task is being executed, when no non-background tasks (Task No. 1 ~ 32) are being executed, the READY output will be ON.



## 6.24 Predefined Constants

There are several predefined constants for use in SPEL<sup>+</sup> program. A project build time, the values for these constants are substituted for the constant name.

Constant name	Value	Use
TRUE	-1	Boolean expression
FALSE	0	Boolean expression
High	1	
Low	0	
Off	0	
On	1	
Above	1	
Below	2	
NoFlip	1	
Flip	2	
Righty	1	
Lefty	2	
J1	1	
J2	2	
J3	4	
J4	8	
J5	16	
J6	32	
J7	64	
MB_OK	0	MsgBox flags
MB_OKCANCEL	1	MsgBox flags
MB_ABORTRETRYIGNORE	2	MsgBox flags
MB_YESNOCANCEL	3	MsgBox flags
MB_YESNO	4	MsgBox flags
MB_RETRYCANCEL	5	MsgBox flags
MB_ICONSTOP	16	MsgBox flags
MB_ICONQUESTION	32	MsgBox flags
MB_ICONEXCLAMATION	48	MsgBox flags
MB_ICONINFORMATION	64	MsgBox flags
MB_DEFBUTTON1	0	MsgBox flags
MB_DEFBUTTON2	256	MsgBox flags
IDOK	1	MsgBox return
IDCANCEL	2	MsgBox return
IDABORT	3	MsgBox return
IDRETRY	4	MsgBox return
IDIGNORE	5	MsgBox return
IDYES	6	MsgBox return
IDNO	7	MsgBox return
BACKCOLORMODE_VISUALSTYLE	0	For GUI Builder
BACKCOLORMODE_USER	1	For GUI Builder
BORDERSTYLE_NONE	0	For GUI Builder
BORDERSTYLE_FIXEDSINGLE	1	For GUI Builder
BORDERSTYLE_FIXED3D	2	For GUI Builder
CNV_QUELEN_ALL	0	Cnv_QueLen
CNV_QUELEN_UPSTREAM	1	Cnv_QueLen
CNV_QUELEN_PICKUPAREA	2	Cnv_QueLen
CNV_QUELEN_DOWNSTREAM	3	Cnv_QueLen
DEVID_OP	23	CLS
DEVID_PC	21	CLS
DEVID_TP	24	CLS
DLG_JOG	100	ShowDialog

Constant name	Value	Use
DLG_IOMON	102	ShowDialog
DLG_ROBOTMNG	100	ShowDialog
DLG_ROBOTPANEL	100	ShowDialog
DLG_VGUIDE	110	ShowDialog
DROPDOWNSTYLE_SIMPLE	0	For GUI Builder
DROPDOWNSTYLE_DROPDOWN	1	For GUI Builder
DROPDOWNSTYLE_DROPDOWNLIST	2	For GUI Builder
ERROR_DOINGMOTION	2999	For GUI Builder
ERROR_NOMOTION	2998	For GUI Builder
EVENTTASKTYPE_NORMAL	0	For GUI Builder
EVENTTASKTYPE_NOPAUSE	1	For GUI Builder
EVENTTASKTYPE_NOEMGABORT	2	For GUI Builder
FORMBORDERSTYLE_NONE	0	For GUI Builder
FORMBORDERSTYLE_FIXEDSINGLE	1	For GUI Builder
FORMBORDERSTYLE_FIXED3D	2	For GUI Builder
FORMBORDERSTYLE_FIXEDDIALOG	3	For GUI Builder
FORMBORDERSTYLE_SIZABLE	4	For GUI Builder
IMAGEALIGN_TOLEFT	1	For GUI Builder
IMAGEALIGN_TOPCENTER	2	For GUI Builder
IMAGEALIGN_TOPRIGHT	3	For GUI Builder
IMAGEALIGN_MIDDLELEFT	4	For GUI Builder
IMAGEALIGN_MIDDLECENTER	5	For GUI Builder
IMAGEALIGN_MIDDLERIGHT	6	For GUI Builder
IMAGEALIGN_BOTTOMLEFT	7	For GUI Builder
IMAGEALIGN_BOTTOMCENTER	8	For GUI Builder
IMAGEALIGN_BOTTOMRIGHT	9	For GUI Builder
IOTYPE_INPUT	0	For GUI Builder
IOTYPE_OUTPUT	1	For GUI Builder
IOTYPE_MEMORY	2	For GUI Builder
IOSIZE_BIT	1	IOLabel function
IOSIZE_BYTE	8	IOLabel function
IOSIZE_WORD	16	
LANGID_ENGLISH	0	ErrMsg\$
LANGID_JAPANESE	1	ErrMsg\$
LANGID_GERMAN	2	ErrMsg\$
LANGID_FRENCH	3	ErrMsg\$
SCROLLBARS_NONE	0	For GUI Builder
SCROLLBARS_HORIZ	1	For GUI Builder
SCROLLBARS_VERT	2	For GUI Builder
SCROLLBARS_BOTH	3	For GUI Builder
SHUTDOWN_ALL	0	Shutdown
SHUTDOWN_RESTART	1	Shutdown
SHUTDOWN_EPSONRC	2	Shutdown
SING_NONE	0	AvoidSingularity
SING_THRU	1	AvoidSingularity
SING_THRUROT	2	AvoidSingularity
SING_VSD	3	AvoidSingularity
SING_AUTO	4	AvoidSingularity
SIZEMODE_NORMAL	0	For GUI Builder
SIZEMODE_STRETCHIMAGE	1	For GUI Builder
SIZEMODE_AUTOSIZE	2	For GUI Builder
SIZEMODE_CENTERIMAGE	3	For GUI Builder
SIZEMODE_ZOOM	4	For GUI Builder
STARTPOSITION_MANUAL	0	For GUI Builder
STARTPOSITION_CENTERSCREEN	1	For GUI Builder

Constant name	Value	Use
STARTPOSITION_CENTERPARENT	2	For GUI Builder
TEXTALIGN_LEFT	1	For GUI Builder
TEXTALIGN_CENTER	2	For GUI Builder
TEXTALIGN_RIGHT	3	For GUI Builder
TEXTALIGN_Topleft	1	For GUI Builder
TEXTALIGN_TOPCENTER	2	For GUI Builder
TEXTALIGN_TOPRIGHT	3	For GUI Builder
TEXTALIGN_MIDDLELEFT	4	For GUI Builder
TEXTALIGN_MIDDLECENTER	5	For GUI Builder
TEXTALIGN_MIDDLERIGHT	6	For GUI Builder
TEXTALIGN_BOTTOMLEFT	7	For GUI Builder
TEXTALIGN_BOTTOMCENTER	8	For GUI Builder
TEXTALIGN_BOTTOMRIGHT	9	For GUI Builder
VISION_SORT_NONE	0	For Vision Guide
VISION_SORT_PIXELX	1	For Vision Guide
VISION_SORT_PIXELY	2	For Vision Guide
VISION_SORT_PIXELXY	3	For Vision Guide
VISION_SORT_CAMERAX	4	For Vision Guide
VISION_SORT_CAMERAY	5	For Vision Guide
VISION_SORT_CAMERAXY	6	For Vision Guide
VISION_SORT_ROBOTX	7	For Vision Guide
VISION_SORT_ROBOTY	8	For Vision Guide
VISION_SORT_ROBOTXY	9	For Vision Guide
VISION_SIZETOFind_ANY	0	For Vision Guide
VISION_SIZETOFind_LARGEST	1	For Vision Guide
VISION_SIZETOFind_SMALLEST	2	For Vision Guide
VISION_BACKCOLOR_NONE	0	For Vision Guide
VISION_BACKCOLOR_BLACK	1	For Vision Guide
VISION_BACKCOLOR_WHITE	2	For Vision Guide
VISION_CAMORIENT_STANDALONE	1	For Vision Guide
VISION_CAMORIENT_FIXEDDOWN	2	For Vision Guide
VISION_CAMORIENT_FIXEDUP	3	For Vision Guide
VISION_CAMORIENT_MOBILEJ2	4	For Vision Guide
VISION_CAMORIENT_MOBILEJ4	5	For Vision Guide
VISION_CAMORIENT_MOBILEJ5	6	For Vision Guide
VISION_CAMORIENT_MOBILEJ6	7	For Vision Guide
VISION_FOUNDCOLOR_LIGHTGREEN	1	For Vision Guide
VISION_FOUNDCOLOR_DARKGREEN	2	For Vision Guide
VISION_GRAPHICS_ALL	1	For Vision Guide
VISION_GRAPHICS_POSONLY	2	For Vision Guide
VISION_GRAPHICS_NONE	3	For Vision Guide
VISION_OPERATION_OPEN	1	For Vision Guide
VISION_OPERATION_CLOSE	2	For Vision Guide
VISION_OPERATION_ERODE	3	For Vision Guide
VISION_OPERATION_DILATE	4	For Vision Guide
VISION_OPERATION_SMOOTH	5	For Vision Guide
VISION_OPERATION_SHARPEN1	6	For Vision Guide
VISION_OPERATION_SHARPEN2	7	For Vision Guide
VISION_OPERATION_HORIZEDGE	8	For Vision Guide
VISION_OPERATION_VERTEDGE	9	For Vision Guide
VISION_OPERATION_EDGEDETECT1	10	For Vision Guide
VISION_OPERATION_EDGEDETECT2	11	For Vision Guide
VISION_OPERATION_LAPLACE1	12	For Vision Guide
VISION_OPERATION_LAPLACE2	13	For Vision Guide
VISION_OPERATION_THIN	14	For Vision Guide

Constant name	Value	Use
VISION_OPERATION_THICKEN	15	For Vision Guide
VISION_OPERATION_BINARIZE	16	For Vision Guide
VISION_OPERATION_ROTATE	17	For Vision Guide
VISION_OPERATION_FLIPHORIZ	18	For Vision Guide
VISION_OPERATION_FLIPVERT	19	For Vision Guide
VISION_OPERATION_FLIPBOTH	20	For Vision Guide
VISION_ACQUIRE_NONE	0	For Vision Guide
VISION_ACQUIRE_STATIONARY	1	For Vision Guide
VISION_ACQUIRE_STROBED	2	For Vision Guide
VISION_TRIGGERMODE_LEADINGEDGE	1	For Vision Guide
VISION_TRIGGERMODE_TRAILINGEDGE	2	For Vision Guide
VISION_THRESHCOLOR_BLACK	1	For Vision Guide
VISION_THRESHCOLOR_WHITE	2	For Vision Guide
VISION_OBJTYPE_CORRELATIO	1	For Vision Guide
VISION_OBJTYPE_BLOB	2	For Vision Guide
VISION_OBJTYPE_EDGE	3	For Vision Guide
VISION_OBJTYPE_POLAR	4	For Vision Guide
VISION_OBJTYPE_LINE	5	For Vision Guide
VISION_OBJTYPE_POINT	6	For Vision Guide
VISION_OBJTYPE_FRAME	7	For Vision Guide
VISION_OBJTYPE_IMAGEOP	8	For Vision Guide
VISION_OBJTYPE_OCR	9	For Vision Guide
VISION_OBJTYPE_CODEREADER	10	For Vision Guide
VISION_OBJTYPE_GEOMETRIC	11	For Vision Guide
VISION_ROBUSTNESS_VERYHIGH	1	For Vision Guide
VISION_ROBUSTNESS_HIGH	2	For Vision Guide
VISION_ROBUSTNESS_MEDIUM	3	For Vision Guide
VISION_ROBUSTNESS_LOW	4	For Vision Guide
VISION_ROBUSTNESS_VERYLOW	5	For Vision Guide
VISION_IMAGESOURCE_CAMERA	1	For Vision Guide
VISION_IMAGESOURCE_FILE	2	For Vision Guide
WINDOWSTATE_NORMAL	0	WindowsStatus
WINDOWSTATE_MINIMIZED	1	WindowsStatus
WINDOWSTATE_MAXIMIZED	2	WindowsStatus
WithMove	0	Recover
WithoutMove	1	Recover

## 6.25 Calling Native Functions in Dynamic Link Libraries

EPSON RC+ 6.0 allows you to call native functions in Dynamic Link Libraries (DLLs).

This is used for complicated arithmetic processing and call for a native function of an external device.

To call the native DLL function, use a Declare statement which is a function definition command from the SPEL<sup>+</sup> program and write a function call as normal.

For details, refer to the *Declare* in the *EPSON RC+ 6.0 SPEL<sup>+</sup> Language Reference*.

### Sample of calling a native DLL

By using a development tool such as Microsoft Visual Studio 2008, you can create a native DLL that can be called from SPEL<sup>+</sup>. Here, it uses Visual Studio 2008 as a sample to create a function that executes the arithmetic operator.

### Step 1: Decide on variable type for a native DLL

You need to plan the data type to use for transferring with the native DLL in the EPSON RC+ 6.0.

Correspondence table for the EPSON RC+ 6.0 data type and the C/C++ variable type is shown below.

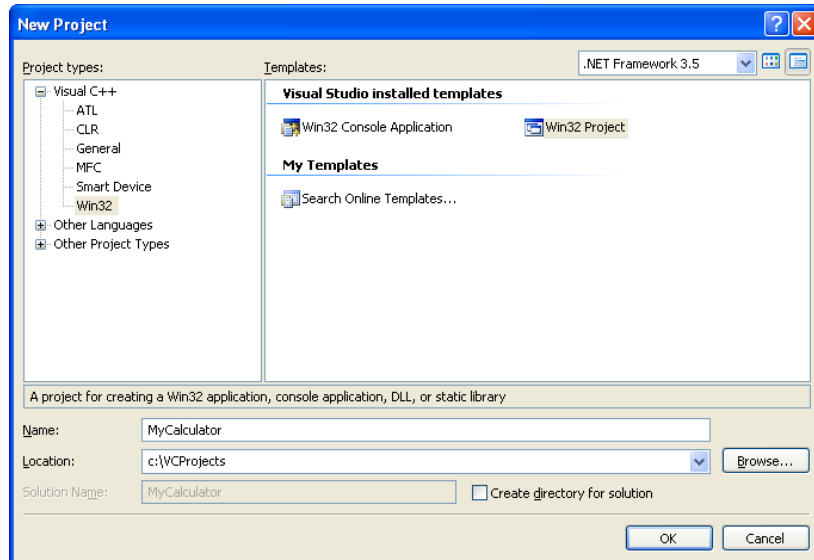
You cannot use the C/C++ byte type and structure because the EPSON RC+ 6.0 has no correspond data for them.

Data correspondence

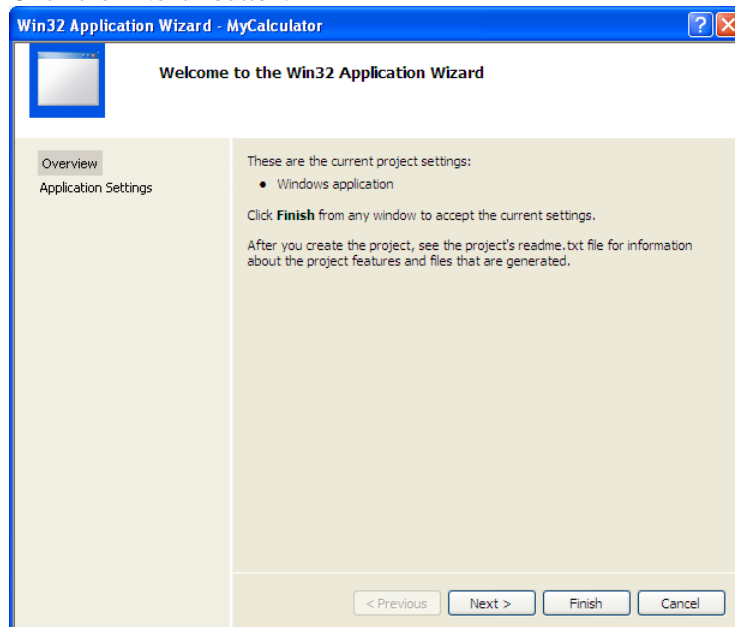
EPSON RC+ 6.0	C/C++
Boolean	short
Byte	short
Short	short
Integer	short
Long	int
Real	float
Double	double
String	char [256] * Null included

## Step 2: Create a native DLL

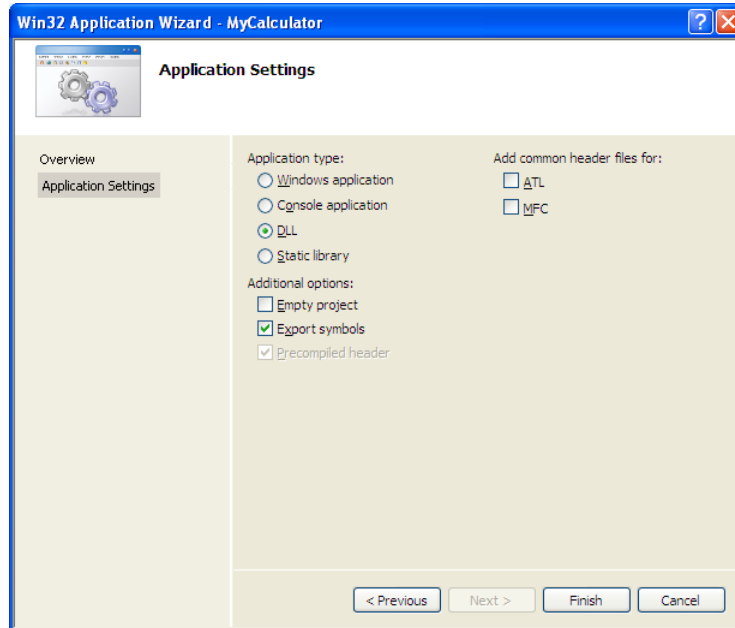
- (1) Start Visual Studio 2008.
- (2) Select the File | New | Project from the Visual Studio 2008 menu.  
Select the Win32 in the [Project type (P):].  
Select the Win32 project in the [Template (T):].  
Type in a project name in the [Project (N):]. (Here types in MyCalculator.)  
Click the <OK> button.



- (3) Start the Win32 application wizard.  
Click the <Next> button.



- (4) Select the <DLL> option button in the [Application type:].  
Check the [Export symbols] box in the [Additional options:].  
Click the **Finish** button.



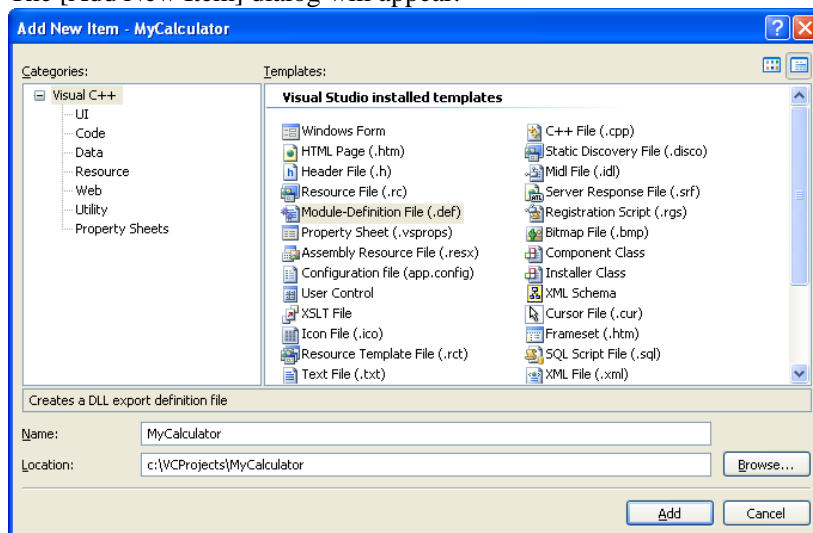
- (5) A simple example of function `fnMyCalculator` will be created in `MyCalculato.cpp`.  
Add a function `MyArithmetic` which executes the arithmetic operator to this file.

```
MYCALCULATOR_API float MyArithmetic(short value1, short
value2, char * kind )
{
    if ( !strcmp(kind, "add") )
    {
        return (float)(value1 + value2);
    }
    else if ( !strcmp(kind, "sub") )
    {
        return (float)(value1 - value2);
    }
    else if ( !strcmp(kind, "mul") )
    {
        return (float)(value1 * value2);
    }
    else if ( !strcmp(kind, "div") )
    {
        return (float)(value1) / (float)(value2);
    }
    else
    {
        strcat_s(kind, 10, " NG");
        return 0;
    }
}
```

- (6) Export a function to enable it to be called from SPEL<sup>+</sup>.

Select the [Add New Item] from the Project menu.

The [Add New Item] dialog will appear.



Select the module definition file (def) in the [Templates:].

Type in a file name in the [Name:].

(Here sets MyCalculator as a file name.)

Click the <Add> button.

Register “fnMyCalculator function” and “MyArithmetic function” to the created “MyCalculator.def” file.

```
LIBRARY      "MyCalculator"

EXPORTS

    fnMyCalculator
    MyArithmetic
```

- (7) Build the project and create the DLL.

Select the Build | MyCalculator build from the Visual Studio 2008 menu.

DLL will be successfully created if any error is displayed.



**Step 3: Call the DLL function from SPEL<sup>+</sup>**

You can now try your DLL function from SPEL<sup>+</sup>.



Before you call your function from the EPSON RC+ 6.0, you must debug it and check thoroughly if it can work without errors.

In case that error occurs (such as system error) in the native function, the EPSON RC+ 6.0 will not work normally.

- (1) Copy the created MyCalculator.dll to the EPSON RC+6.0 project folder (e.g. C:\EpsonRC60\projects\dllcall).
- (2) Define a DLL function which executes the arithmetic operator in the SPEL<sup>+</sup> program and write a function call for MyArithmetic in Function main.

```
Declare MyArithmetic, "MyCalculator.dll"(value1 As Integer,
value2 As Integer, ByRef calc$ As String) As Real
```

```
Function main
```

```
    Real result;
```

```
    String calc$
```

```
    calc$ = "add"
```

```
    result = MyArithmetic(1, 2, ByRef calc$);
```

```
    Print "1+2=", Str$(result)
```

```
    calc$ = "sub"
```

```
    result = MyArithmetic(1, 2, ByRef calc$);
```

```
    Print "1-2=", Str$(result)
```

```
    calc$ = "mul"
```

```
    result = MyArithmetic(1, 2, ByRef calc$);
```

```
    Print "1*2=", Str$(result)
```

```
    calc$ = "div"
```

```
    result = MyArithmetic(1, 2, ByRef calc$);
```

```
    Print "1/2=", Str$(result)
```

```
End
```

- (3) Build and execute the project.

The following result will be displayed.

```
1+2=3
```

```
1-2=-1
```

```
1*2=2
```

```
1/2=0.5
```



Before you build the project, be sure to copy the native DLL to the project folder without fail. If you fail, a warning or error will occur.

## 7. Building SPEL<sup>+</sup> Applications

### 7.1 Designing Applications

#### 7.1.1 Creating the simplest application

The simplest SPEL<sup>+</sup> application has one program and one point file. This is what is automatically defined for you when you create a new project. A blank program named “Main.prg” and a blank point file named “Points.pts” are created.

#### To write and run a simple application

1. Select New Project from the Project Menu to create a new project.
2. Write your program source code in the file that was created for you called Main.prg.
3. Teach the robot points using the Robot Manager Jog and Teach page.
4. Run the program by selecting Run Window from the Run Menu or by pressing F5 (the shortcut key for the Start command).

#### 7.1.2 Application layout

Before writing your application, you need to decide what your application will accomplish and how the project will be structured. Here are some general guide lines.

#### Programs

Each project can contain up to 64 programs that can be started from the Operator Window, Remote Control, VB Guide, or GUI Builder. Each program has a start function, as shown in the table below.

Program #	Program Name	Start Function
0	main	main
1	main1	main1
2	main2	main2
3	main3	main3
4	main4	main4
5	main5	main5
6	main6	main6
7	main7	main7
...	...	...
63	main63	main63

Your project must always define function main so that the main program can be started. The other programs are optional. If you use the Operator Window for your operator interface, you can define meaningful names for each of the programs used in your project in Project | Properties | Operator setting | Operator Window.

#### Operator interface

##### Operator Window

Use the operator window provided with EPSON RC+ 6.0. You can configure EPSON RC+ 6.0 so that after Windows starts, EPSON RC+ 6.0 will start in Auto mode, which will automatically open the Operator Window.

Operators can select up to 64 programs. They can also optionally use the Pause/Continue buttons, I/O Monitor, Robot Manager, and System History viewer.

To use the Operator Window to allow programs to be started and stopped, the Control Device must be set to Self from Setup | System Configuration | Controller | Configuration.

For details on configuring EPSON RC+ 6.0 for auto start, see section *Start Mode* in the *Operation* chapter.

**Remote Control**

Use remote control to turn motors on/off, home the robot, start programs, etc. A simple push button box can be used, or a PLC can be connected.

When using Remote Control, the Control Device must be set to Remote from Setup | System Configuration | SPEL Controller Board | Configuration.

**Windows Applications using VB Guide**

Use the VB Guide Option along with a Windows development tool such as Visual Basic, Visual C#, or Visual C++. See the VB Guide Manual for more information.

**GUI Builder**

To use the GUI Builder option, refer to the *GUI Builder manual*.

**Safety interface**

Use guard doors, safety mats, light curtains, etc. to protect the operator from injury.

**Robot Points, Pallets, Tools, Locals**

Decide on which points you need for the work cell. In many cases you will only need one point file per robot.

Take advantage of Pallets, Tools, and Locals. Time spent on using these can save hours later on the production line. For example, if your cell has many points that take a lot of time to train, consider using Locals so that if the end effector is damaged or replaced, you only need to redefine the Locals, not retrain all of the points.

Try to design in automatic or semi-automatic procedures for calibrating tools and locals. Even if you define them manually, write instructions on how to define them so the process can be repeated easily.

**Inputs and outputs**

Layout your I/O early in the design stages. Use I/O labels in your programs. You must purchase additional I/O boards if you need more than 24 inputs or 16 outputs. You can also use the Fieldbus option so the controller can be a Fieldbus slave.

**Peripherals**

RC620 Controller has one RS-232C port as standard (two ports depending on the type of controller). You can also add up to 9 ports by installing an optional RS232C expansion board. Refer to *12. RS-232 Communications* for details.

You can use TCP/IP to connect peripheral equipment. For details, refer to *13. TCP/IP Communications*.

**7.1.3 Auto start at power up**

Your application can automatically log in a Windows user and start your SPEL<sup>+</sup> project after Windows boots.

Refer to *4.3.7 Auto Start*.

### 7.2 Managing Projects

#### 7.2.1 Overview

##### **What is an EPSON RC+ 6.0 Project?**

An EPSON RC+ 6.0 project is a collection of SPEL<sup>+</sup> program files, include files, robot point files, I/O labels, user errors, Vision settings, and conveyor settings used to run a SPEL<sup>+</sup> application.

##### **Why do you need projects?**

Projects are a safe and convenient way to manage your SPEL<sup>+</sup> applications. All the information for each application is kept in one project. By keeping all of your application code and point definitions in one project, it's easy to open a project and begin running or editing. Also, it's easy to create new versions of an application and run older versions.

Projects make it easier to maintain your application code with less chance of a program being lost.

There are also functions for copying and renaming projects, making it easy to create new projects from previous versions and for backing up projects to an external media such as a USB memory key.

##### **What's in an EPSON RC+ 6.0 project?**

Each project is stored in the \EpsonRC60\Projects directory.

The following paragraphs describe the components of a project.

##### **Project file**

This file contains all of the information that describes the project. This file is automatically created by EPSON RC+ 6.0. You should never edit this file. Doing so may cause errors when you open the project. The file extension is SPRJ.

##### **Program files**

A program file is an ASCII text file that contains one or more SPEL<sup>+</sup> functions. Each function in SPEL<sup>+</sup> can be run as a separate task (thread) in the controller or called from other functions.

Include files can also be used. These contain macro definitions and must be included in a program file using the #include statement. The file extension is PRG.

##### **Point files**

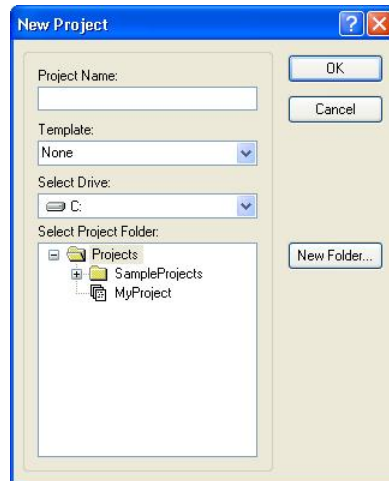
A point file contains a list of robot points. The file extension is PTS.

##### **Include files**

In the include file, you can declare variables and macros. The file extension is INC.

### 7.2.2 Creating a new project

Projects always reside in specific drive, \EpsonRC60\Projects folder. Also you can create a sub-folder to systematize the projects of different types.



To create a new project

1. Select New Project from the Project Menu. The [New Project] dialog box will appear.
2. Select the disk drive where you want the project to be stored on.
3. Select the project folder or create a new folder by clicking the **New Folder** button after selecting the parent folder.
4. Type in the name for the new project.
5. Optionally, select a template to base the project on.
6. Choose **OK** to create the project.

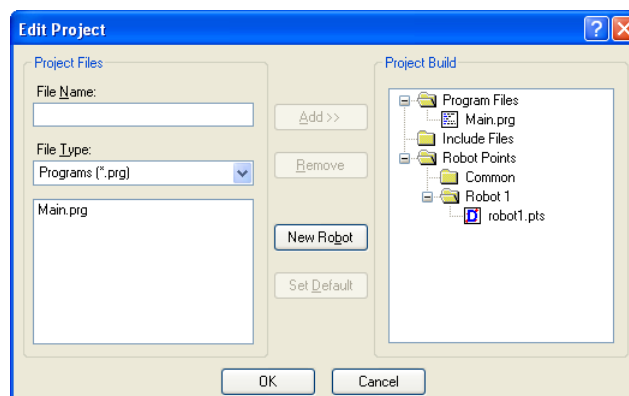
### 7.2.3 Configuring a project

Each application project you create must be configured properly before you can run the program.

There are two commands in the Project Menu that allow you to configure a project: Edit and Properties.

#### Editing a project

Select Edit from the Project Menu to open the [Edit Project] dialog. From this dialog, you configure which program files, include files, and point files are used in the current project.



For details on Project | Edit, refer to 5.9.5 *Edit Command*.

### 7.2.4 Building a project

Before you can run any of the code in your application, you must build the project.


#### To build your application project

Select Build from the Project Menu or click on the Build button  on the toolbar.

Or

Select Rebuild from the Project Menu. This will rebuild the entire project.

Or

Select Run Window from the Run Menu or click the Run button  on the toolbar. The project will be built before the Run Window appears.

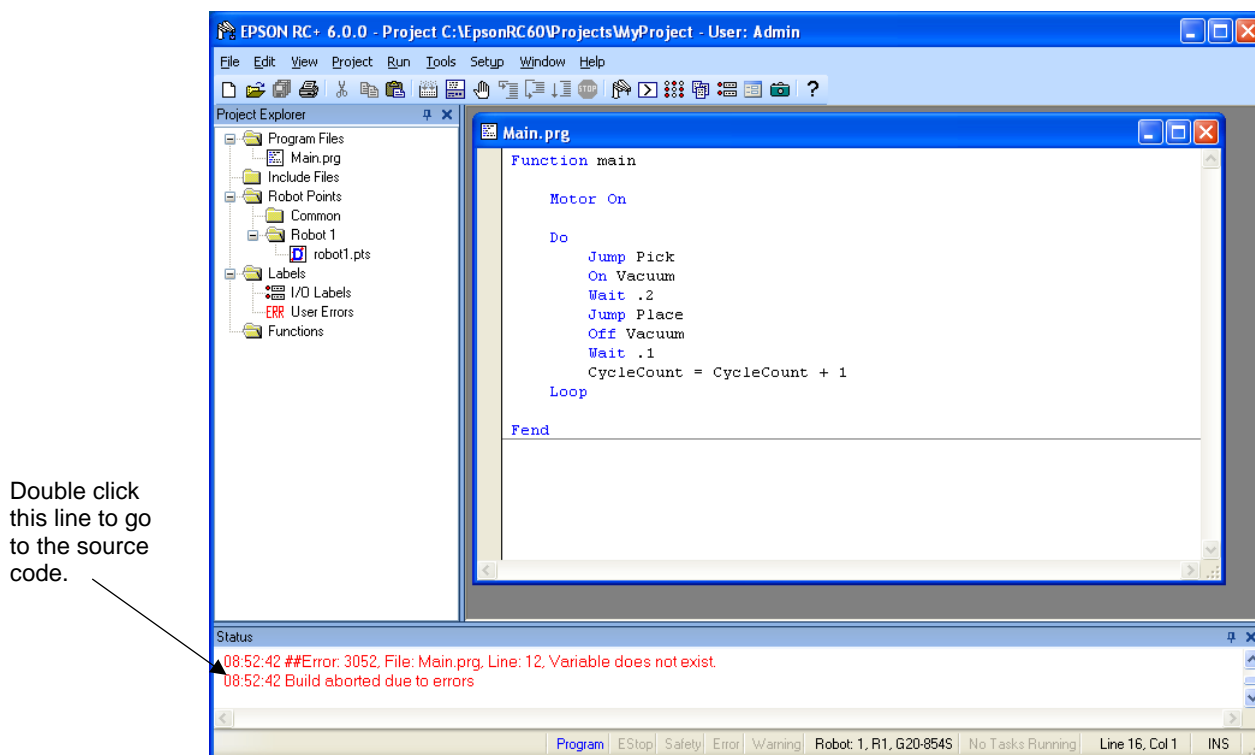
Or

Select Operator Window from the Run Menu. The project will be built before the Operator Window appears.

After the files have been compiled and linked, the project files are sent to the controller.

#### Status Pane

This window shows progress messages and error messages during project build.



When errors occur during the build process, a message is displayed that includes the error number, program file name, and line number. Double click on the line with the error to go directly to the source code that caused the error.

### 7.2.5 Backing up a project

To make a backup copy of the current project, use the Copy Project command in the Project Menu to copy the project to another disk drive or folder. You can also save the project under a different name.

This command is useful for transferring a project to an external media such as a USB memory.

## 7.3 Editing Programs

Before you can edit a program, it must be in the current project and opened in a program window.

### To open a program for editing

1. Select Open from the File Menu.
2. Select the file(s) you want to open.
3. Choose **OK** to open the file.

#### 7.3.1 Program rules

A program contains one or more SPEL<sup>+</sup> function definitions.

Lines can be blank. You can insert any number of blank lines to separate subroutines and functions, if desired.

The maximum length for each line is 512 characters, including the line number, if used.

#### 7.3.2 Typing in program code

You can enter program statements in upper or lower case. Whenever you leave a line that has been changed, the line will be formatted. SPEL<sup>+</sup> keywords are case formatted and spaces are inserted around operators and after semi-colons and commas.

Consider using mixed case or lower case for variables and function names instead of all CAPS. This will make your code easier to read.

Use indentation for statements within loops. The **Auto Indent** feature automatically moves the cursor under the start of the previous line. It also indents lines after If, Else, For, Select, Case, and Do statements.

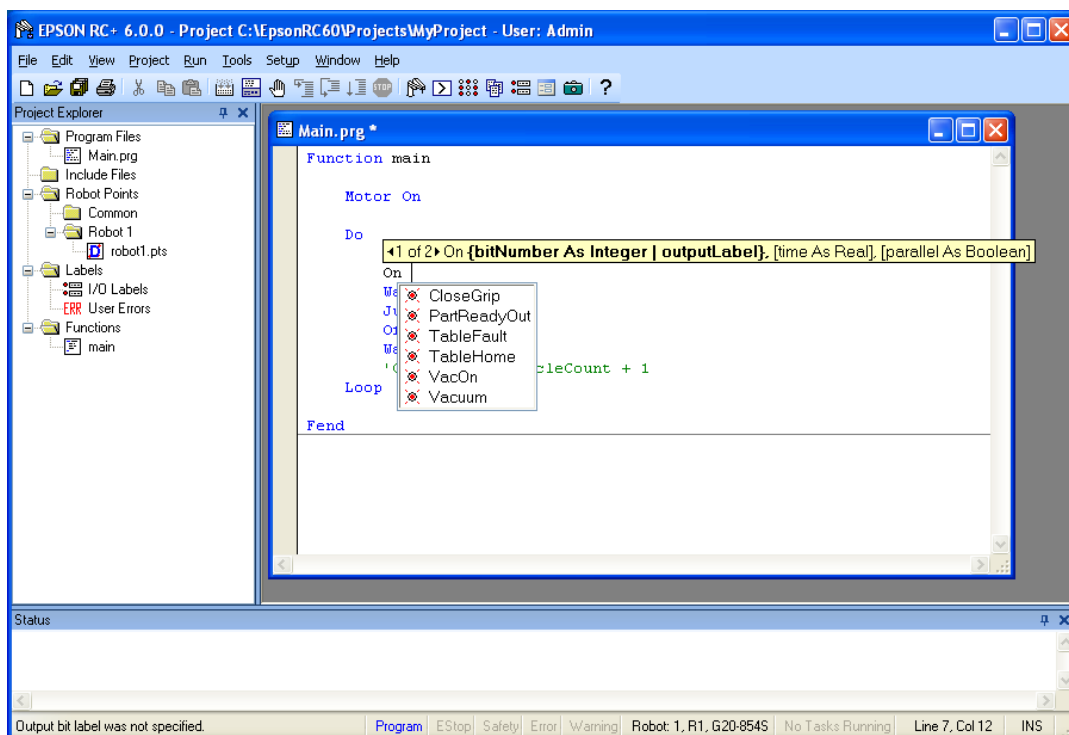
```
For i = 1 To 10
    Jump P(i)
    Jump P0
Next i
```

Use the **Auto End Construct** feature to automatically add the end construct statement. For example, when you enter a For statement and press Enter, a Next statement is automatically created with an indented blank line above it.

### 7.3.3 Syntax Help

When you type in a SPEL<sup>+</sup> keyword, the syntax help window will appear to show the syntax of the statement or function. After the statement is entered, the syntax helper will automatically close, or you can press the Esc key to close it. You can enable / disable Syntax Help from the Setup | Preferences | Editor tab.

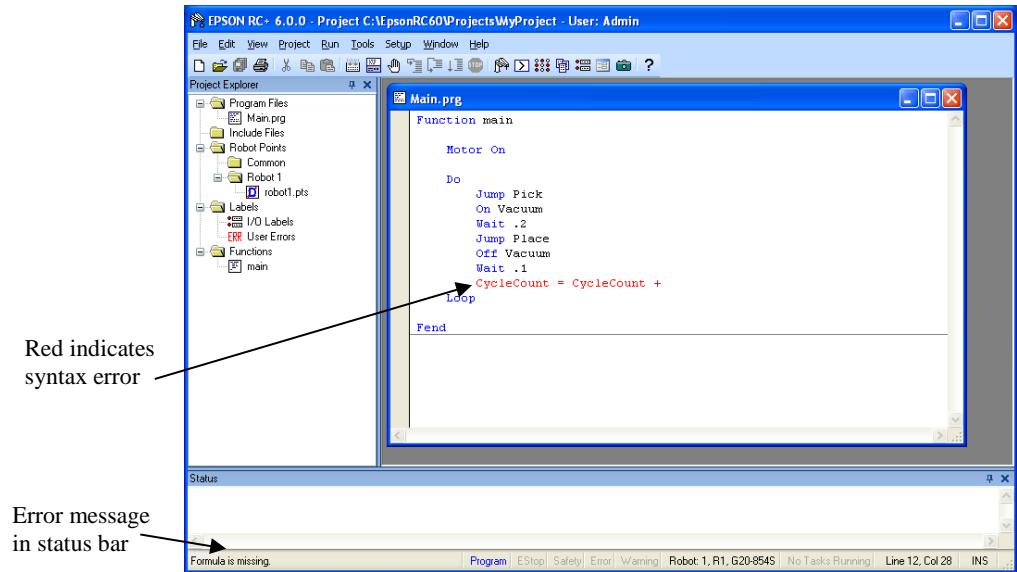
A list box will be displayed for some parameters as you type. To select a value in the list, use the up and down arrow keys, or type the first few characters, to highlight the desired item, then press Tab to select the item. You can also type in a value not shown in the list, such as a variable or literal constant. Press Esc to hide the list box. In addition to Tab, you can use a comma or period to select an item. In the example shown below, the first parameter of the On statement can be an output label, so a list of output labels in the current project is displayed.





### 7.3.4 Syntax Errors

When a syntax error is detected, the line with the error will be displayed in red. If the caret is placed on the line with the error, then a brief message will be displayed on the status bar. For example, in the program shown below, the message "Expression expected" is displayed on the status bar.



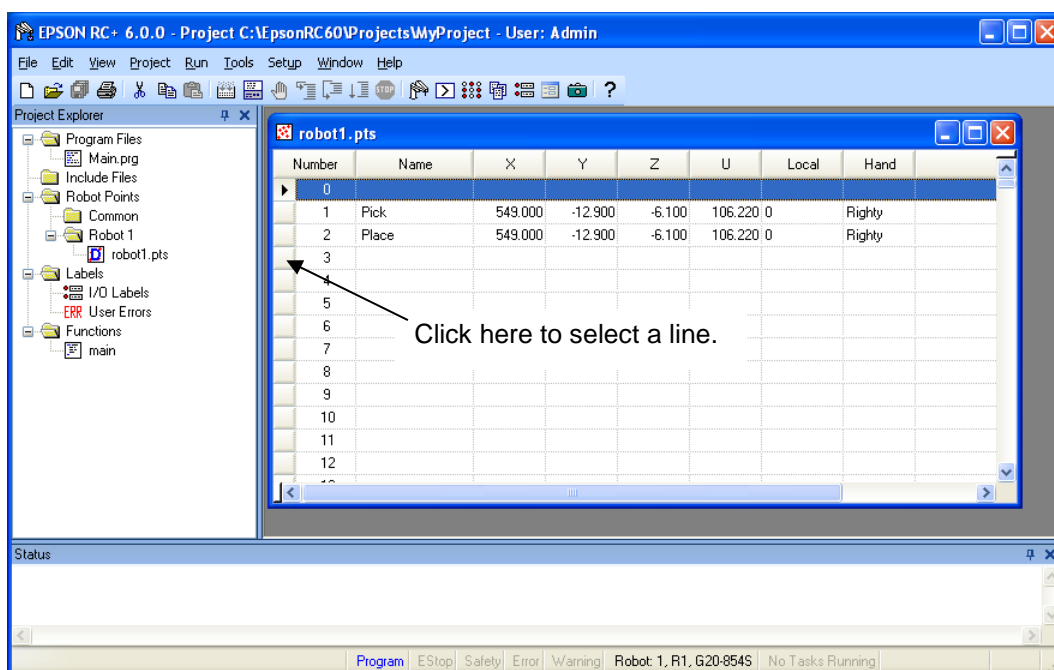
## 7.4 Editing Points

You can edit the robot points from the robot point file. You can define new points or cut, copy, and paste points from one point file to another, including between projects.

### To open a point file for editing

1. Select Open from the File Menu to show the Open dialog box.
2. Choose the **Points** option button. You will see a list of point file names in the files list box.
3. Select the point file you want to edit by clicking on the name.
4. Click **Open** to open the file. You will see a spread sheet window for the point file you selected.

### The robot points spread sheet window



The spreadsheet window contains one row for each point in the file. The spreadsheet always contains rows for all points, even if they are not defined. The cells for an undefined point are blank.

<b>Row select column</b>	This is the first column on the left. Click on this column to select a row.
<b>Number column</b>	Point number. Range is from 0 to a maximum point number.
<b>Name column</b>	Name of the point.
<b>Coordinate columns</b>	Coordinates in millimeters for X, Y, Z and degrees for U, V, W.
<b>Local number column</b>	Local number drop down list. Range is from 0 to 15.
<b>Hand column</b>	Drop down list with two values for robot hand orientation: Lefty and Righty.
<b>Elbow column</b>	Drop down list with two values for robot elbow orientation: Above and Below. This column is shown only for 6-axis robots.
<b>Wrist column</b>	Drop down list with two values for robot wrist orientation: Flip and NoFlip. This column is shown only for 6-axis robots.
<b>J4Flag column</b>	Drop down list with two values for robot J4Flag: 0 and 1. This column is shown only for 6-axis robots.
<b>J6Flag column</b>	Drop down list with values for robot J6Flag: 0 - 127. This column is shown only for 6-axis robots.
<b>J1Flag column</b>	Drop down list with values for robot J1Flag: 0 and 1. This column is shown only for RS series.
<b>J2Flag column</b>	Drop down list with values for robot J2Flag: 0 and 1. This column is shown only for RS series.
<b>J1Angle</b>	Coordinate in units of degrees. This column is shown only for RS series.

**To select one or more rows**

Click on the row select column (first column on the left) to select a row. To select more than one row, point to the row select column of the first row you want to select. Hold down the left mouse button and drag the mouse down or up to select more rows.

**To select all rows**

Execute Select All from the Edit Menu, or type Ctrl + A.

**To define a new point**

Using the mouse put the spreadsheet cursor anywhere on the row of the point you want to define and start entering information for the point. This automatically defines the point, which means it will be sent to the robot controller at the next project build or Jog and Teach command.

For example, put the cursor in the Name column and type in a name for the point. Press the TAB key to move to the X coordinate column. Type a coordinate value, then press Enter. You will see zeros automatically entered in the coordinates of Y ~ W and the Local column. This indicates that the point as been defined.

**To delete a point**

Select the row containing the point and cut it by selecting Cut from the Edit Menu or by typing Ctrl + X.

**To cut and paste points**

1. Select one or more rows and execute either the Cut or Copy command from the Edit Menu.
2. Select the row where you want to start the paste.
3. Execute the Paste command from the Edit Menu.

## 7.5 Running and Debugging Programs

You can run programs from the Run Window or from the Operator Window. The Run Window is used primarily for testing and debugging. The Operator Window is used as an operator interface for simple applications or demos. You can also run programs using the VB Guide option or GUI Builder option.


### To run a program

Select Run Window from the Run Menu. This command will build the project (if required) and open the Run Window. The Run Window allows you to choose which function to execute. Select a function, then click **Start**.

#### 7.5.1 The Run window

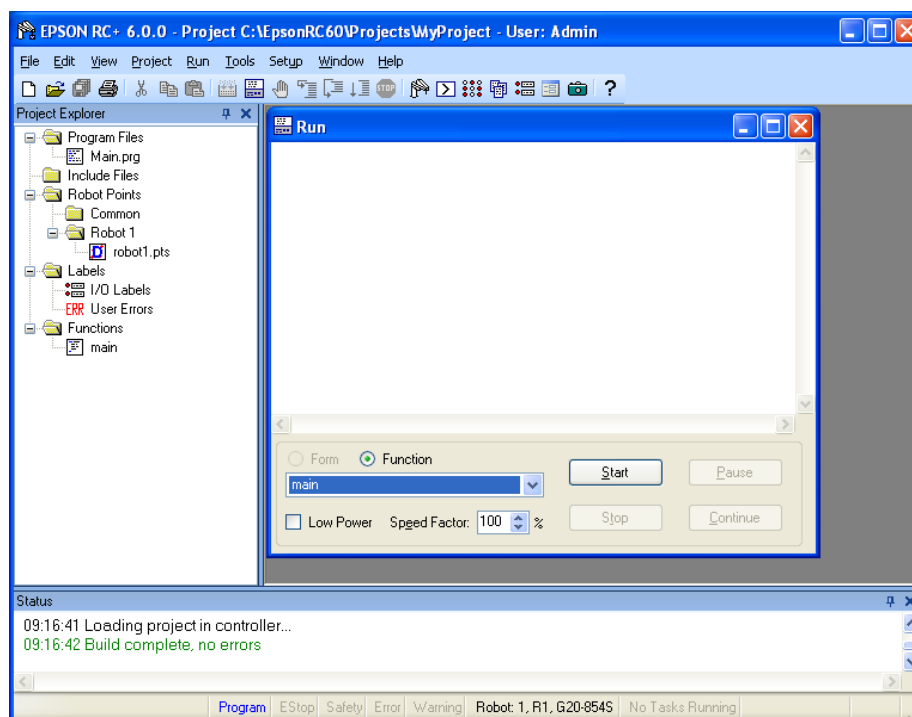
The Run window includes controls for running the programs in the current project.

### To open the Run window

Select Run Window from the Run Menu, or click on the Run button  on the toolbar. If necessary, all changed open files will be saved and the project will be built. If the build is successful, the Run window will appear.

### To close the Run window

Choose Close from the File Menu or click on the **X** button in the upper right hand corner of the window.



Item	Description
<b>Text area</b>	<p>This is the area that takes up most of the run window. Output from your programs is displayed here. When your program uses an Input statement, you can type in the requested input from this text box. You can use the scroll bars to view the entire text buffer.</p> <p>If an error occurs while running a program, the error number, program file name, line number and function name will be displayed in this text area. You can double click on the line where the error is displayed to directly go to the source line that caused the problem.</p>
<b>Function</b>	Select a function to start. Functions are sorted alphabetically. Function main is selected by default.
<b>Low Power</b>	When this box is checked, SPEL <sup>+</sup> ignores the Power High command. This allows you to run your program in low power mode to verify operation without having to change the program.
<b>Speed Factor</b>	Specifies the robot motion speed factor. The speed factor is a percentage of maximum point to point speed and linear interpolated speed. For example, if you program executes Speed 80 and the speed factor is 50%, the robot will move at speed 40.
<b>Start</b>	Starts the function shown in the function drop down list.
<b>Stop</b>	Stops all tasks. If the robot is executing a motion command when this button is pressed, the robot will decelerate to a stop.
<b>Pause</b>	Pauses all tasks with pause enabled. Activates the <b>Continue</b> button. If the robot is executing a motion command when this button is pressed, the robot will decelerate to a stop.
<b>Continue</b>	Continues paused tasks.
<b>CTRL+C</b>	Same as <b>Stop</b> button.

### 7.5.2 Debugging

EPSON RC+ 6.0 supports source level debugging. You can set breakpoints and step through your source code. You can also pause / continue a program or halt a task using the Task Manager.

#### Setting and clearing breakpoints

Open the program where you want to set a breakpoint, then click on the line where you want to stop. Use one of the following methods to set a breakpoint:

- If Margin Indicators are enabled, then click in the margin next to the line on the left. You will see a breakpoint symbol next to the line.

Or

- Type F9.

Or

- Select Toggle Breakpoint from the Run Menu.

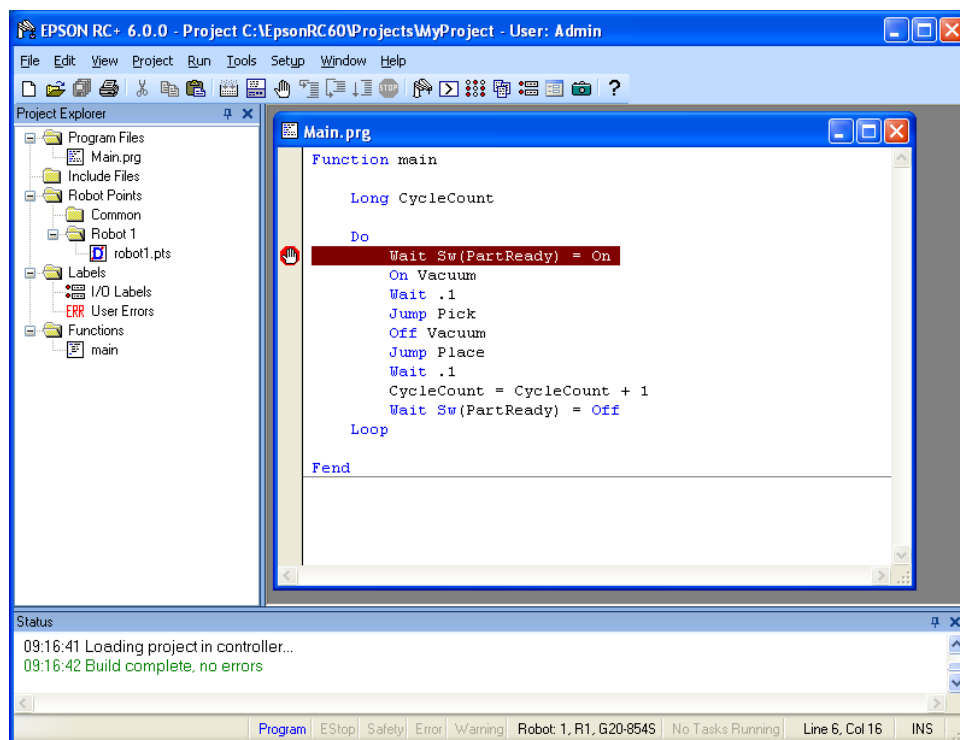
Execute one of the methods above to clear a breakpoint, or select Clear All Breakpoints from the Run Menu.

You cannot set a breakpoint on non-executing statements, such as #define, #include, or blank lines.

After setting a breakpoint, the task will halt when the execution line is reached the breakpoint. You can set or clear a breakpoint while a task is running.

When reached a breakpoint, the program window containing the program source line at the breakpoint is opened and the line is highlighted in yellow. The task number is shown in the title of the program window.

If more than one task reaches a breakpoint, then a program window will be opened for each task. This allows you to step through each task that reached the breakpoint.



## Stepping through a program

There are three commands on the Run Menu that are used for stepping through code.

**Step Into** steps through each line and also steps into functions when a step is executed on a Call statement.

**Step Over** steps through each line but when a Call statement is encountered, the function in the statement is executed completely.

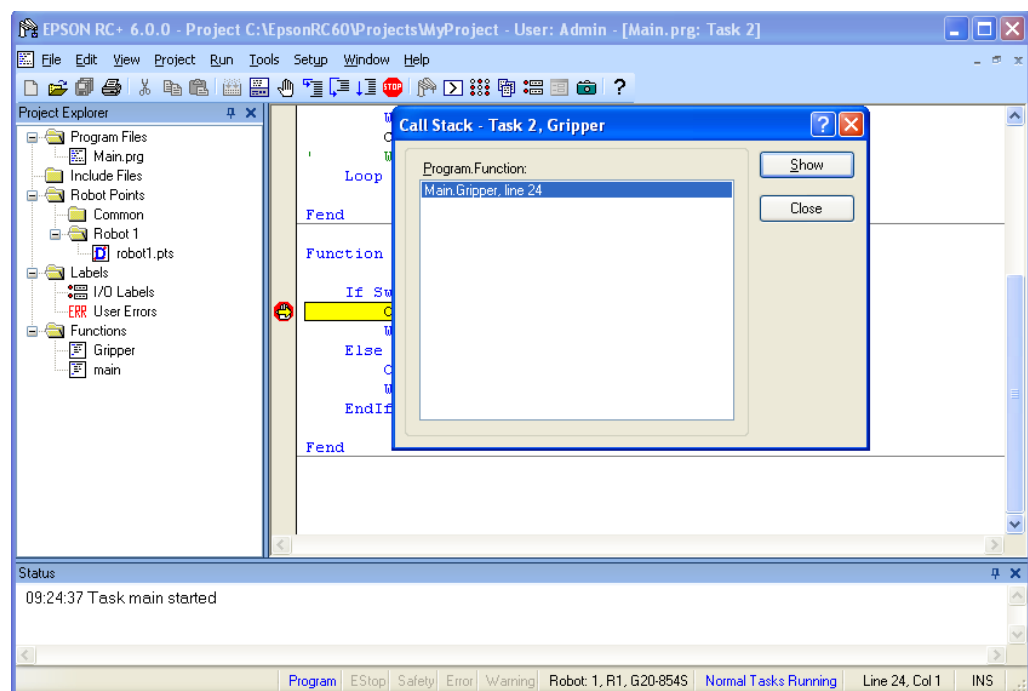
**Walk** executes lines until after the next motion command and then halts the task. It will halt after the next output command if the Setup | System Configuration | Controller | Preferences *Walk stops for outputs* checkbox is checked.

To step through code, you must either set a breakpoint and run until the breakpoint is reached, or suspend a task from the Task Manager using the Halt button.

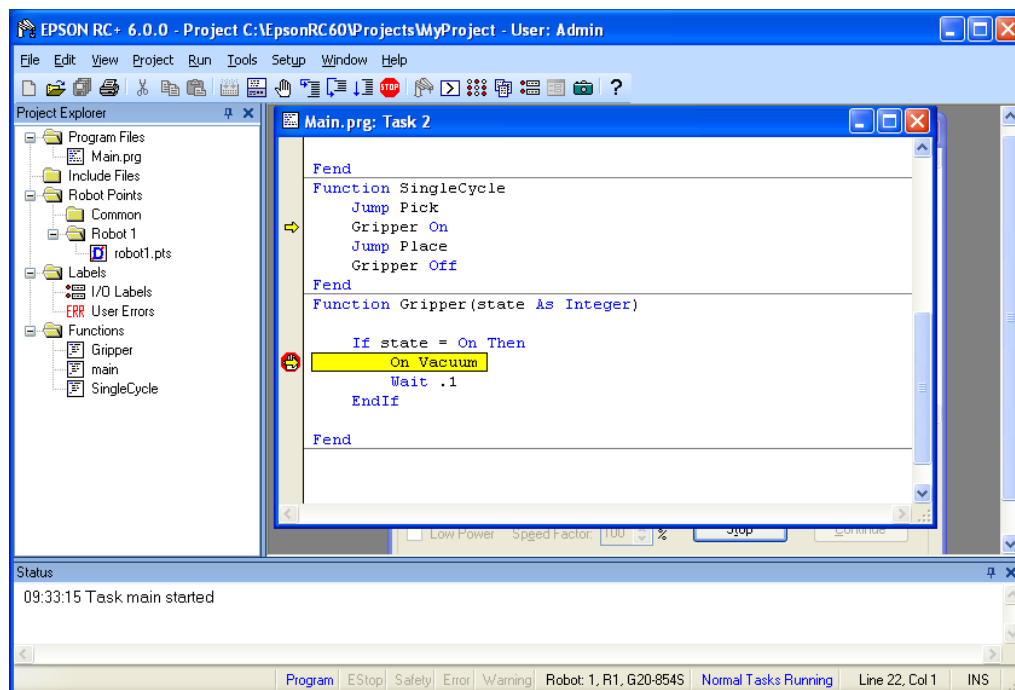
## Viewing the Call Stack

Sometimes you may want to examine the call stack for the current task after you halt the task from the task manager, or reach a breakpoint.

To view the call stack, select Call Stack from the Run Menu. The Call Stack list will be displayed, as shown below.



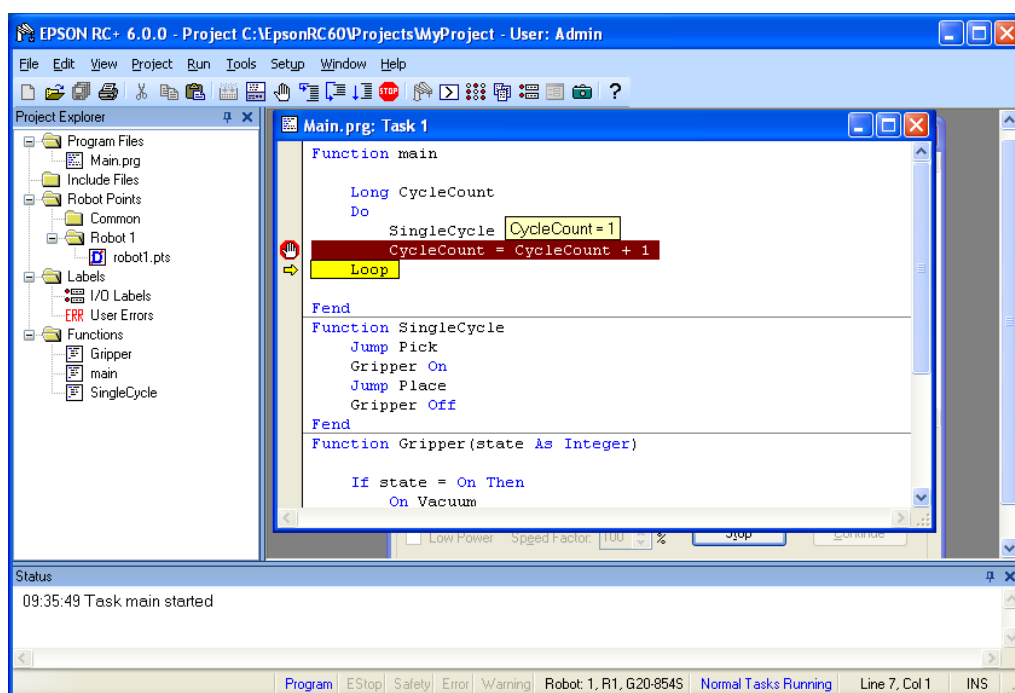
After you double click a function in the Call Stack list, the function will be displayed in a program window and an arrow in the left margin will point to the line where the next function in the call stack is being called. In the example below, the arrow in the SingleCycle function is pointing to the Gripper On statement to indicate that Gripper was called from SingleCycle.



### Displaying variables

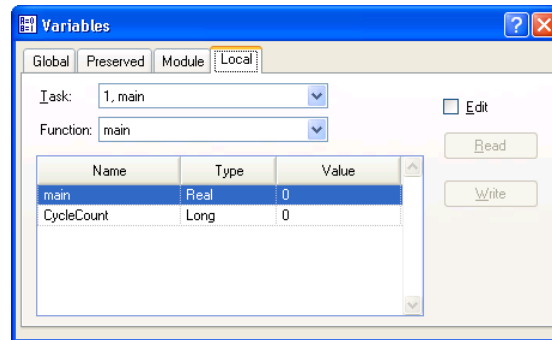
To view variable values, you can do one of the following:

1. When a task is halted by halt or breakpoint, you can view the value for a variable by moving the mouse cursor over the variable name. The value will be displayed in a tooltip type window beneath the variable name.





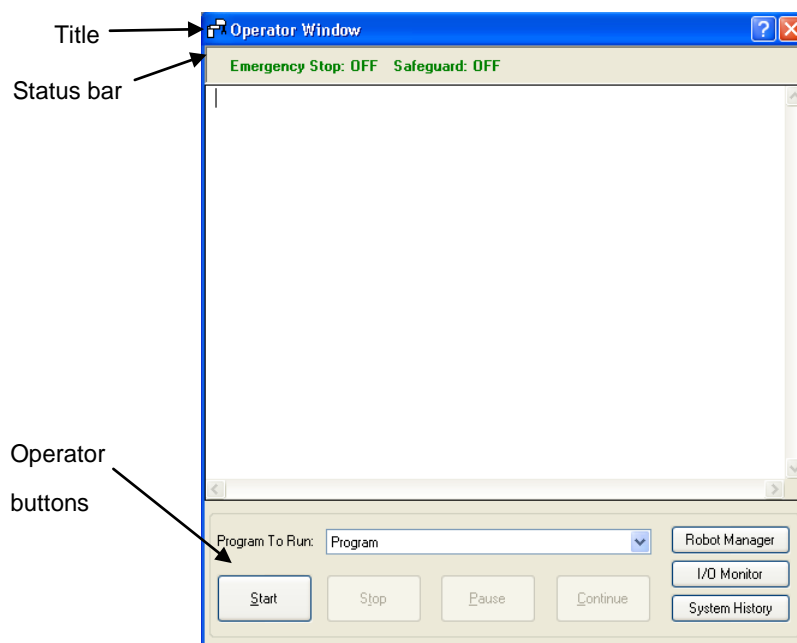
2. Select Display Variables from the Run Menu to display the variable display dialog. This dialog has three tabs for viewing Global, Module, and Local variables.



You can change the value of a variable by checking the Edit check box, then type in the new value in the value column. Next, click the Write button to change the variable. When the Edit box is checked, the variable values are not automatically updated. You can click the Read button to update all values.

## 7.6 The Operator Window

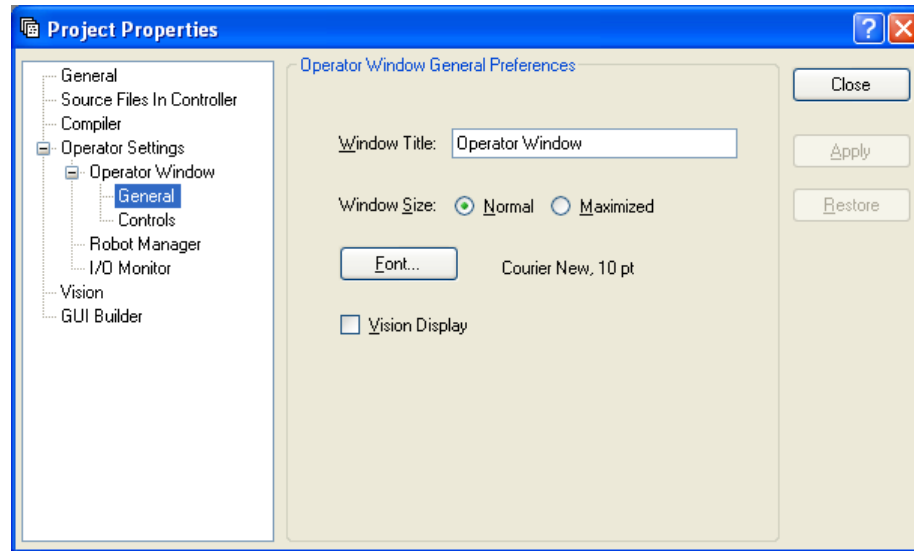
The Operator Window can be used as a simple interface for operators. You can configure EPSON RC+ 6.0 to open only the Operator Window when started. In addition, when Remote Control is being used, the Operator Window can be displayed for monitoring purposes.



Item	Description
<b>Program to Run</b>	Select a program to run.
<b>Start</b>	Starts the selected program.
<b>Stop</b>	Stops all tasks.
<b>Pause</b>	Pauses all tasks that are enabled for pause.
<b>Continue</b>	Continues paused tasks.
<b>Robot Manager</b>	Opens the Robot Manager dialog in operator mode. It cannot be shown while the program is running.
<b>I/O Monitor</b>	Opens the I/O Monitor in operator mode. This window can remain open while programs are running.
<b>System History</b>	Opens the System History window. This window can remain open while programs are running.
<b>Status Bar</b>	The status bar is located at the top of the window and shows emergency stop and safeguard status. In addition, if a warning is detected from the controller (such as low encoder battery), a warning label will be displayed on the right side of the status bar. If the mouse is over this label, you can see the warning error message. When there is no warning, the warning label is hidden.

### 7.6.1 Operator window configuration

You can configure the Operator Window from the Operator Window pages in Project | Properties.

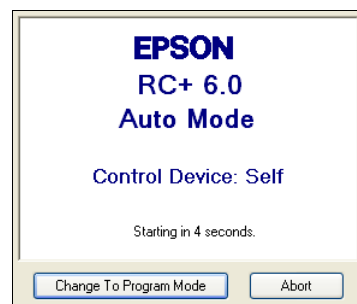


There are several settings for operator Robot Manager and I/O Monitor.

For details, refer to *5.9.14 Properties Command*.

### 7.6.2 Auto start configuration

You can configure the system to let it log into Windows automatically. Also you can configure a program to start automatically from the [Operator] window. For details, refer to *4.3.7 Auto Start*.



## 7.7 Using Remote Control

You can design your application to be run from external equipment using hardware I/O control. This includes push button boxes, PLCs, and other PC systems.

Refer to *11. Remote Control* for details.

## 7.8 Using Encrypt Files

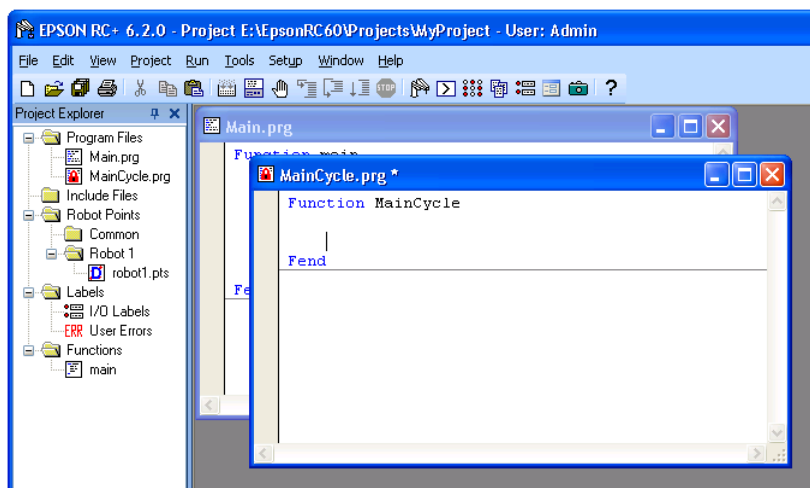
Encrypted files allow you to prevent end users from viewing your source code. When a file is encrypted, you must supply a password to open the file. Other users cannot view the file contents, even with an external editor, such as Notepad.

Each encrypted file can have its own password, or you can choose to encrypt multiple files with one password. You can encrypt program files, include files, Vision Guide, and GUI Builder.

If an encrypted file is imported from another project, it will remain encrypted in the current project.

As an example, assume you have some special SPEL+ programming code that you do not want your end users to view. But you want to allow end users to change some of the code in the project. To do this, put all of the functions you want to be hidden in one or more encrypted program and include files. When you go to the customer site, you can view your encrypted code by supplying the password(s) to open the encrypted files.

When files are encrypted, their icons are shown with a lock in the Project Explorer and also in the title bar of the program window. In the screenshot below, the file `MainCycle.prg` is encrypted, so its icons include a lock image.



When you open an encrypted file, you will be prompted for the password.



CAUTION

### USE EXTREME CAUTION!

Keep a record of the password(s) used for encryption in a safe place. Once a file is encrypted, it can only be opened with the password you enter. If you forget the password, the file contents **CANNOT BE RECOVERED**.

To configure encrypted files in your project, select Properties from the Project menu, then select Encrypted Files in the tree on the left. Refer to section 5.9.14 *Properties Command (Project Menu)* for details.

## 8. Motion System

EPSON RC+ supports the motion systems listed below.

- Standard Motion System
- PG Motion System

### 8.1 Standard Motion System

The standard motion system is built into one RC620 Control Unit and one or two external Drive Unit(s) as an option.

You can connect up to two robots to a Control Unit. For details on the RC620 controller hardware and maintenance, refer to the *RC620 Robot Controller manual*.

To setup a Drive Unit, open the [System Configuration]-[Drive Unit] page and add the Drive Unit.

### 8.2 RC620 Drive Module Software Configuration

The RC620 drive module is configured at the factory before shipment. It is automatically recognized by the controller and you do not have to configure the settings.

Also, you do not have to configure the settings for the drive module in the Drive Unit which is automatically recognized.

### 8.3 PG Motion System

The PG (Pulse Generator) Motion System is an option.

When a PG board is installed in the controller, it is automatically recognized. You can select it in the robot configuration dialog.

For instructions on using the PG Motion System, refer to the *RC620 Robot Controller option PG Motion System manual*.

## 9. Robot Configuration

This chapter contains information for adding robots and configuring additional axes.

- Robot Configuration

Adding a standard robot

- Additional axes Configuration

Adding a robot with additional axes

Robots are configured from the Robots folder on the Setup | Controller dialog tree.

### 9.1 Setting the Robot Model

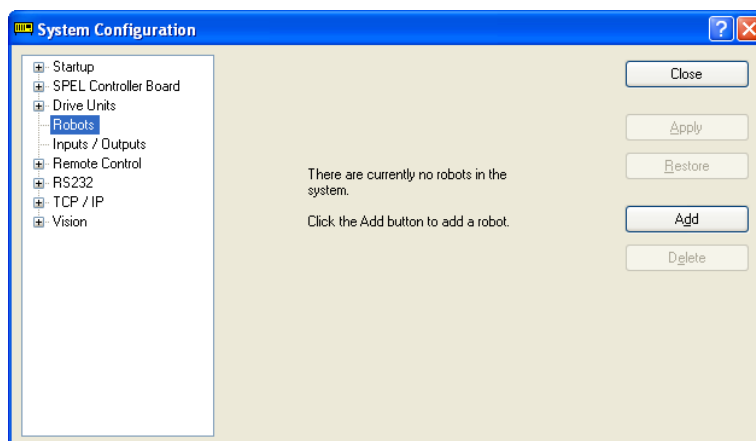


- We have configured each manipulator before shipment so normally you don't need to change the settings. If you change the settings, it may cause the robot to malfunction or perform unusual motion. This is extremely hazardous and you should be careful.

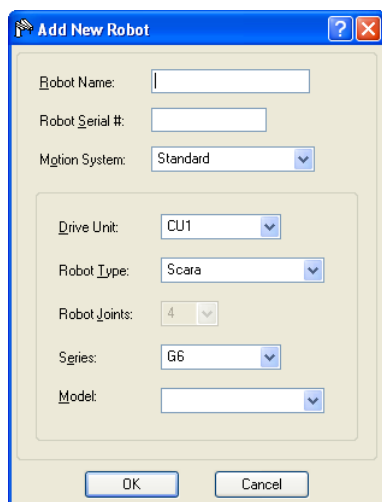
#### 9.1.1 Adding a standard robot

If you have purchased the PG motion system Option, you can add user defined robots. Refer to the *EPSON RC+6.0 Option manual: PG Motion System*.

1. From the Setup Menu, select System Configuration.
2. Click on **Robots** in the tree on the left.



3. Click **Add** and the next dialog will appear.



4. Type in a name for the new robot and enter the serial number from the robot's nameplate. Any serial number can be used, but it is recommended that you use the number that is stamped on the robot.
5. Select a motion system to use from the [Motion System] dropdown list. If there are no other motion systems installed, then "Standard" will already be selected.
6. Select a Drive Unit for your robot from the [Drive Unit] dropdown list.
7. Select a robot type from the [Robot type] box.
8. If you are adding an EZ module, select the number of robot axes from the [Robot Joints] dropdown list.
9. Select a robot series from the [Series] dropdown list.
10. Select a robot model from the [Model] dropdown list.  
After you select a robot model, all robots available for the type of motor driver currently installed in the controller will be shown. If you use [Dry run], all robots selected in step 9 will be shown.
11. Click **OK** and the controller will be rebooted.

### 9.1.2 Calibrating a standard robot

The calibration method differs according to the robot model.

For details, refer to the *Manipulator manual: Maintenance section: Calibration*.

### 9.1.3 Changing robot system parameters

The following system parameters for the robot can be changed from EPSON RC+ 6.0:

#### - **Enable/Disable Joints**

You can disable one or more joints from Setup | System Configuration | Robots | Robot\*\* | Configuration. On robots with a ball screw Z axis, you must disable both joints 3 and 4 together.

#### - **Hofs**

Hofs are the joint home offsets. You can view and edit the values from System Configuration | Robots | Robot\*\* | Calibration. However, it is recommended that you use the Robot Calibration wizard to set these values. These values are unique for each robot and are supplied from the factory. Hofs are especially important for SCARA robots because the values determine that both lefty and righty hand orientation will position the robot at the same point.

#### - **CalPIs**

CalPIs values are joint calibration offsets. You can view and edit the values from System Configuration | Robots | Robot\*\* | Calibration. However, it is recommended that you use the Robot Calibration wizard to set these values. These values are unique for each robot and are supplied from the factory. CalPIs values are used to calibrate joint position after replacing a motor or encoder.

These are one-time settings for each robot. Additional robot parameters can be set from the Robot Manager.

To change robot parameters, follow these steps

1. Select System Configuration from the Setup Menu.
2. Under the Robot folder in the tree on the left, select Robot\*\* | Calibration.
3. Execute the calibration wizard or change values for Hofs or CalPIs.
4. Click **Apply** to make the changes permanent.

## Saving robot calibration data

You can save and load individual robot calibration files. This is useful for moving a robot from one controller to another. When you save calibration data, a file is created with an MPD file extension. This file contains Hofs and CalPIs values.

To save robot calibration data

1. Select System Configuration from the Setup Menu.
2. Under the Robot folder in the tree on the left, select Robot\*\* | Calibration.
3. Ensure that the robot serial number is correct. The serial number will be used to create the default file name. It is recommended that the serial number be used.
4. Click the **Save Cal** button. Browse to a destination directory and click Save.

## Loading robot calibration data

To load robot calibration data

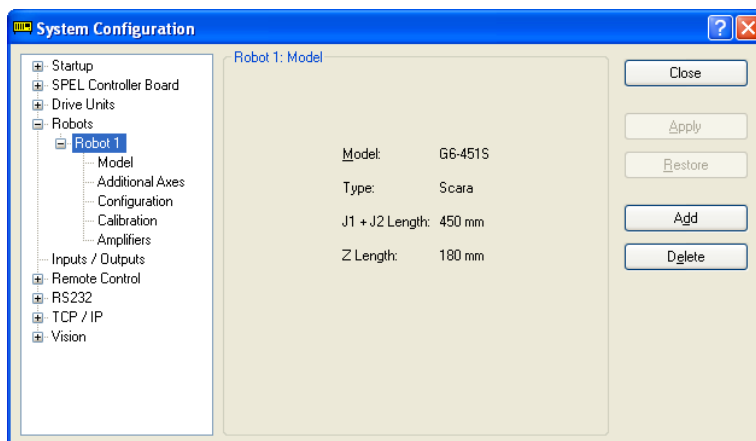
1. Select System Configuration from the Setup Menu.
2. Under the Robot folder in the tree on the left, select Robot\*\* | Calibration.
3. Click the **Load Cal** button. Browse to the desired MPD file and click Open.

### 9.1.4 Deleting a standard robot

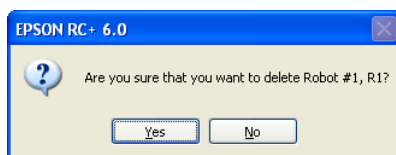
1. Select System Configuration from the Setup Menu.
2. Under the Robot folder in the tree on the left, select Robot\*\*.



You can only delete the last robot.



3. Click **Delete** and the next dialog will appear.



4. Click **Yes** and the controller will be rebooted.

If you delete only an additional axis from its installed robot, refer to 9.2.6 *Deleting the additional axes*.



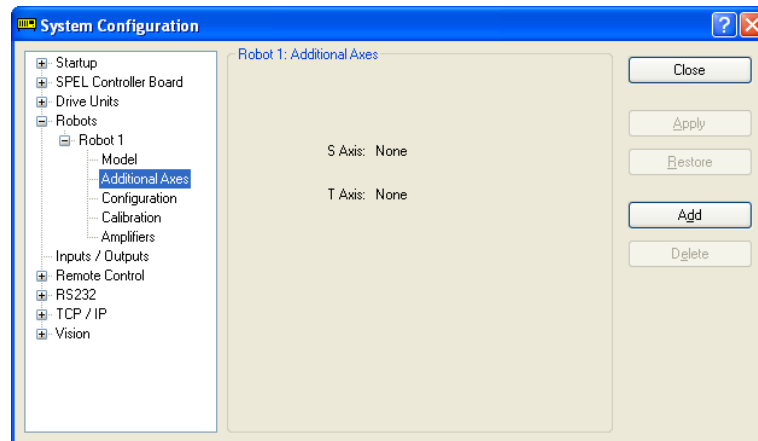
## 9.2 Configuration of Additional Axes

Using the additional axes feature, you can configure the axes that move with the manipulator.

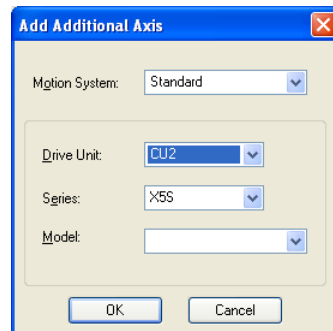
You can configure up to two additional axes (S and T).

### 9.2.1 Adding the additional S axis

1. Select System Configuration from the Setup menu.
2. Under the Robot folder in the tree on the left, select Robot\*\* | Additional Axes



3. Click **Add** and the next dialog will appear.



4. Select a motion system from the [Motion System] dropdown list. If there are no other motion systems installed, then “Standard” will already be selected.
5. Select a Drive Unit from the [Drive Unit] dropdown list. Here, Robot 1 uses “CU1”, so “CU2” will be selected.
6. Select a robot series from the [Series] dropdown list.
7. Select a robot model from the [Model] dropdown list. After you select a robot model, all robots available for the type of motor driver currently installed to the controller will be shown. If you use [Dry run], all robots selected in step 6 will be shown.
8. Click **OK** and the controller will be rebooted.



NOTE

When you configure additional axes, you should perform the calibrations for the added axes.

Refer 9.2.3 *Calibrating the additional axes* and perform the calibration.

### 9.2.2 Adding the additional T axis



After the additional S axis has been added to the robot, you can add the additional T axis. The procedure is the same as for the S axis. Refer to *9.2.1 Adding the additional S axis*.

### 9.2.3 Calibrating the additional axes

Note that the additional axes have not been calibrated before shipment unlike the manipulator.

Before use, perform the calibration.

1. Move the additional axes to the home position.
2. Create a data for the calibration

Type the following into the [Command Window] and execute it.

```
> Calpls 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

3. Execute the calibration.

Type the following into the [Command Window].

Additional S axis : >Calib 8

Additional T axis : >Calib 9

Additional S and T axes: >Calib 8, 9



- If you execute an incorrect calibration, it can cause the manipulator to perform unusual motion and cause serious safety problems. This is extremely hazardous and you should be careful.

### 9.2.4 Changing the parameters of robot with additional axes installed

For details, refer to *9.1.3 Changing robot system parameters*.

### 9.2.5 Differences of the standard robot and robot with additional axes

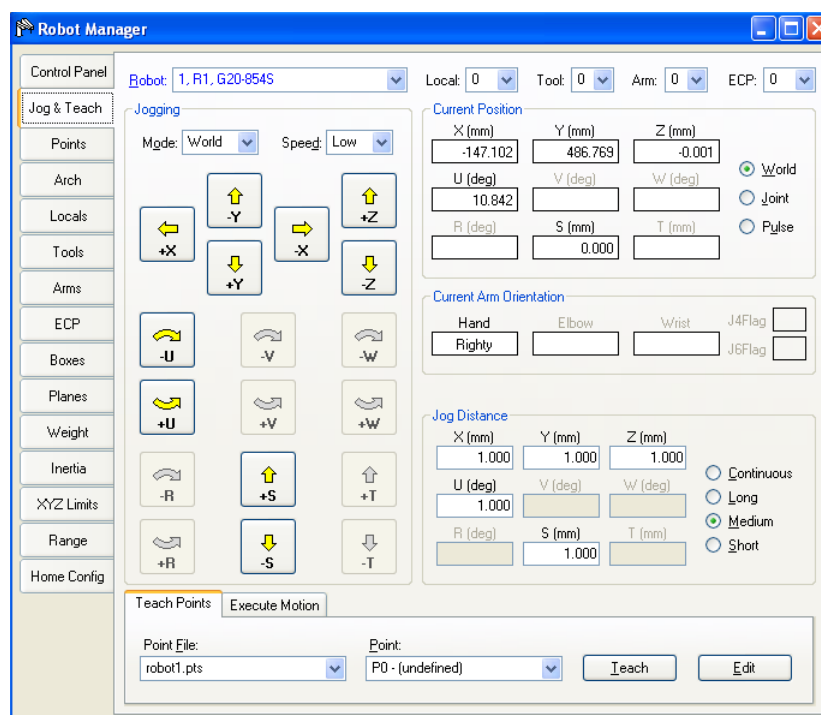
The robot with additional axes installed has some parts which are different from the standard robot when using GUI and SPEL<sup>+</sup> commands.

For the SPEL<sup>+</sup> commands, refer to the *SPEL<sup>+</sup> Language Reference manual*.

The main differences in the EPSON RC+ 6.0 GUI are as below.

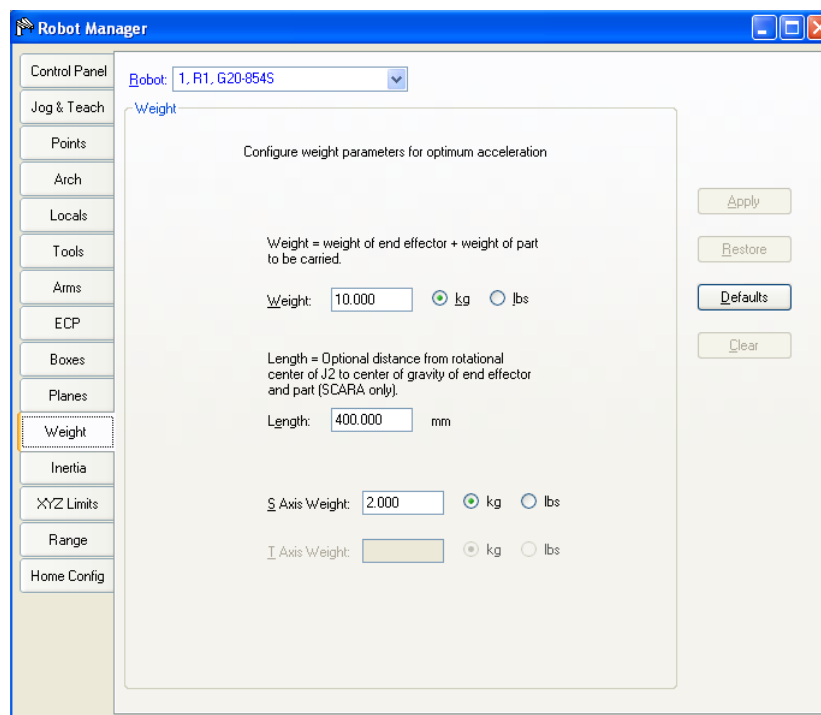
## Tools: Robot Manager: Jog & Teach Page

You can jog the additional S and T axes. When the additional T axis is not installed, the jog buttons will be dimmed.



## Tools: Robot manager: Weight Page

This page is for changing the Weight parameters for the robot. When the additional T axis is not installed, the corresponding weight setting will be dimmed.



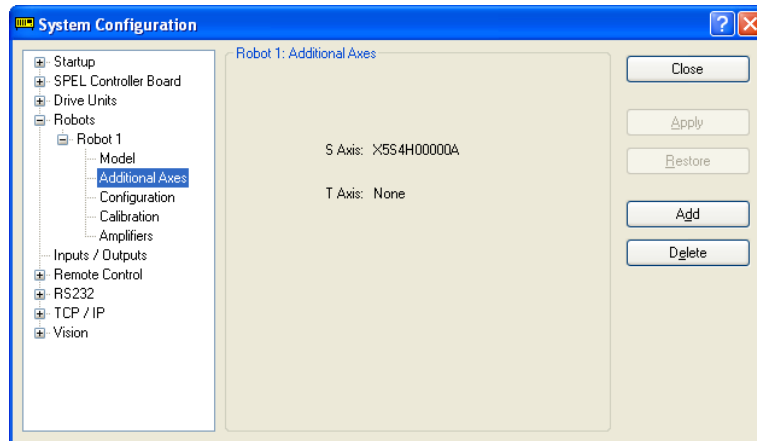
### 9.2.6 Deleting the additional axes



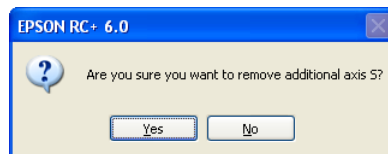
When the additional T axis is installed, delete it first.

When only the additional S axis is installed, delete it.

1. Select System Configuration from the Setup Menu.
2. Under the Robot folder in the tree on the left, select Robot\*\* | Additional Axes.



3. Click **Delete** and the next dialog will appear.



4. Click **Yes** and the controller will be rebooted.

## 10. Inputs and Outputs

### 10.1 Overview

The RC620 controller I/O has the following types of I/O:

**Standard I/O** This digital I/O comes standard with the controller.

**Expansion I/O** This is optional digital I/O that can be added to the controller to expand standard I/O. Up to four boards can be added, each with 32 inputs and 32 outputs.

**Fieldbus master I/O**

An optional board for the controller to expand the standard I/O. You can add one of the following boards which support the fieldbus master board: DeviceNet, EtherNet/IP, PROFIBUS-DP

**Fieldbus slave I/O**

An optional board for the controller to expand the standard I/O. You can add one of the following boards which support the Fieldbus slave mode: DeviceNet, EtherNet/IP, PROFIBUS-DP, CC-Link

**Memory I/O** This is built-in memory bits that can be used for inter-task communications.

For Standard, Expansion, Fieldbus master, and Fieldbus slave I/O, there are input bits numbered starting with 0, and output bits numbered starting with 0.

For memory I/O, each memory bit is both an input and an output.

For specifications and instructions on wiring I/O, refer to the *Robot Controller RC620 manual*.

### 10.2 I/O Commands

The SPEL+ language has several commands for inputs and outputs listed below. For details on each command, see the SPEL+ Language Reference.

#### Input Commands

In Reads one byte of input bits.

InBCD Reads one byte of input bits in Binary Coded Decimal format.

InW Reads one word of input bits.

Oport Reads one output bit.

Sw Reads one input bit.

#### Output Commands

Off Turns off one output bit with optional time.

On Turns on one output bit with optional time.

OpBCD Sets one byte of output bits in Binary Coded Decimal format.

Out Sets / reads one byte of output bits.

OutW Sets / reads one word of output bits.

#### Memory I/O Commands

MemOff Turns off one memory bit.

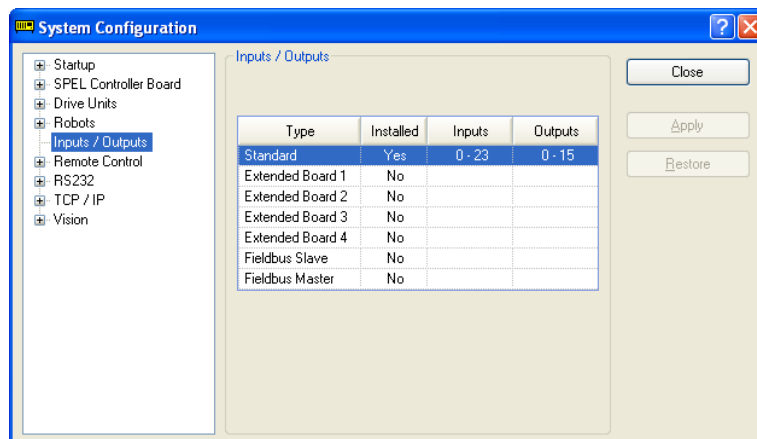
MemOn Turns on one memory bit.

MemOut Sets / reads one byte of memory bits.

MemSw Reads one bit of memory.

### 10.3 I/O Configuration

To view the current I/O configuration, select Setup | System Configuration | Inputs and Outputs. This will show you what I/O is installed on the controller.



#### Standard and expansion I/O

The board is automatically configured by the controller. To add expansion I/O boards, refer to *Robot Controller RC620 manual*.

The standard I/O in the Drive Unit automatically increases depending on the number of Drive Unit.

#### Fieldbus master I/O / Fieldbus slave I/O

For details on how to configure, add, check the boards, refer to the *RC620 Robot Controller option, Fieldbus I/O manual*.

### 10.4 Monitoring I/O

To monitor I/O, use the I/O Monitor tool by selecting Tools | I/O Monitor. From the I/O monitor, you can view inputs and outputs or memory I/O in bit, byte, and word formats.

For details on how to use the I/O Monitor tool, see 5.11.3 *I/O Monitor Command*.

### 10.5 Virtual I/O

The RC620 controller supports virtual I/O. When enabled, virtual I/O allows you to simulate your hardwired I/O. You can turn on / off any input bit or output bit. Normally this is used when the controller is in Dry Run mode with no robot or I/O connected.

#### Virtual I/O Commands

SetIn	Set the value of an 8 bit input port.
SetInW	Set the value of a 16 bit input port.
SetSw	Set the value of one input bit.

### 10.6 Fieldbus Master I/O

The Fieldbus master I/O is an option.

For details on how to use, refer to the *RC620 Robot Controller option, Fieldbus I/O manual*.

### 10.7 Fieldbus Slave I/O

The Fieldbus slave I/O is an option.

For details on how to use, refer to the *RC620 Robot Controller option, Fieldbus I/O manual*.

# 11. Remote Control

## 11.1 Overview

By using Input/Output, the controller can control robots from an external device. The external device can execute several commands, including Motor On/Off, Start, Pause, Continue, and Stop.

There are three basic steps required for remote control configuration:

1. Configure Remote Control inputs and outputs using the Remote Control tab on the Setup | System Configuration | Remote Control page.  
Nothing is initially assigned to remote functions.
2. Set the control device to remote on the Setup | System Configuration | Configuration page.  
To enable external remote inputs, assign the remote functions and also set the control device to remote. When control device setting is remote, the controller is only controllable from the remote device.
- 3 Set the controller startup mode to the Cooperative mode or Independent mode.  
In the Cooperative mode, the controller is in Ready status after RC+ becomes ready.

Remote control function can be used in the following systems.

Example: Control the robot from a PLC

Use remote control to control the robot (controller) from a PLC.

When using a PLC, you will need to be familiar with the handshake required to use remote inputs. See details below.

Example: Control the robot using a push button box with buttons and lights

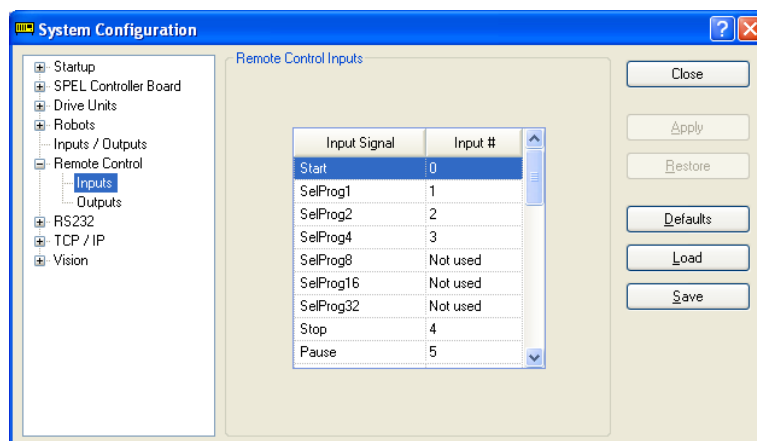
The lights are connected to remote control outputs on the controller to indicate status, such as AutoMode, MotorOn, Error, etc. The buttons are connected to remote inputs to control motor power and start programs.

For details of each I/O connection, refer to the *RC620 Robot Controller manual* or *Fieldbus I/O option manual*.

## 11.2 Remote Control Input Output Configuration

The following is the procedure to assign the remote control functions to the I/O system.

1. Select System Configuration from the Setup Menu and select the **Remote Control Inputs** or **Remote Control Outputs** page.
2. For each input or output you want to use for remote control, click on the Input # or Output # cell for the desired signal, then click the dropdown arrow and select a bit number in the list.
3. Click **OK** to save the new settings.



For details using this dialog, refer to 5.12.1 System Configuration Command (Setup Menu).

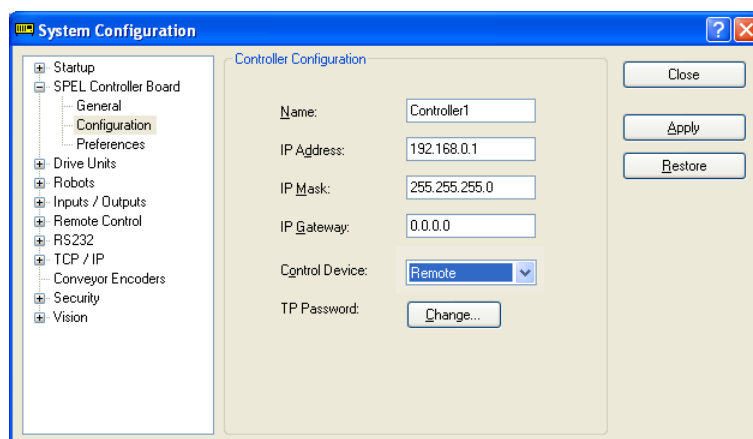
## 11.3 Control Device Configuration

The following is the procedure to set the control device to “Remote I/O”.

1. Select System Configuration from the Setup Menu, and click on SPEL Controller Board | Configuration in the tree on the left.

Select **Remote** in the [Control Device] box.

2. Click **Apply** to save the new setting and the click **Close**.



For details on using this dialog, refer to 5.12.1 System Configuration Command (Setup Menu) – Setup / System Configuration / Controller / Configuration.



## 11.4 Auto Mode with Remote Control

To run in auto cycle with remote control

1. The host device (e.g. PLC) should wait for the AutoMode or Ready remote output to turn on before issuing remote commands.
2. Now the remote input commands will be accepted.

To monitor remote operation from the EPSON RC+ 6.0 Operator Window

1. Set the EPSON RC+ 6.0 Start Up Mode to **Auto**. For details, refer to 4.2.3 *Start Up Mode*.
2. The PC should also be configured to auto log into Windows and start EPSON RC+ 6.0 during Windows start. Refer to 4.2.7 *Auto Start*.

## 11.5 Teach Mode with Remote Control

When using Teach Mode with remote control, an Teach mode is ON, no remote input commands can be used. Remote status outputs will still operate.



- Remote status outputs (such as MotorOn, Home, etc.) will operate when Teach Mode is ON, even when the enable switch (dead man) is disengaged. Therefore, **DO NOT** use remote status outputs to drive any devices that cause motion or any other safety hazard.

You can monitor teach mode status using the TeachMode remote output.

## 11.6 Debugging Remote Control

You can debug programs using Remote Control from the EPSON RC+ 6.0 development environment.

To run programs by remote control for debugging:

1. Create a program (in the same manner as usual).
2. Open the Run Window and click Enable Remote I/O.
3. Now the remote commands will be accepted.

You can set breakpoints and print messages to the run window.



If you cannot wire the I/O, use virtual I/O mode for debugging. Remote function is also available when virtual I/O is enabled.

## 11.7 Remote Inputs

Remote inputs are used to control the Manipulator(s) and start programs. Certain conditions must be met before inputs are enabled, as shown in the table below.

To accept external remote inputs, assign the remote function and set remote to the control device. When external remote input is available, "AutoMode output" turns ON.

Except SelProg and SelRobot, the signals execute each function when the signal starts in input acceptance condition. The function executes automatically. Therefore, no special programming is needed.

### NOTE



- No remote signals are initially configured at shipment time.
- When an error occurs, you must execute a "Reset" to clear the error condition before any other remote input commands can be executed. Use the "Error output" and "Reset input" to monitor the error status and clear error conditions from the remote device.
- If a remote input command cannot be accepted when you execute it, the CmdError output will turn ON. Check the input acceptance condition and re-execute the command.

Name	Description	Input Acceptance Condition (*1)
Start	Executes function selected at SelProg. (*2)	Ready output ON Error output OFF EStopOn output OFF SafeguardOn output OFF Pause input OFF Stop input OFF
SelProg1 SelProg2 SelProg4 SelProg8 SelProg16 SelProg32	Specifies the executing Main function number. (*2)	
Stop	All tasks and commands are stopped.	
Pause	All tasks are paused. (*3)	Running output ON
Continue	Continues the paused task.	Paused output ON Pause input OFF Stop input OFF
Reset	Resets emergency stop and error. (*4)	Ready output ON
Shutdown	Shutdown the system.	
ForcePowerLow (*8)	Stops all tasks and commands. Sets the motor power of all robots at Low. The status is Low power mode while the input is ON even executing Power High command.	Any time This input is accepted even when the AutoMode output is OFF.
SelRobot	Changes the output condition of MotorsOn, AtHome, PowerHigh, and MCalReqd. (*9)	
SelRobot1 SelRobot2 SelRobot4 SelRobot8 SelRobot16	Specify the number of robot which executes a command. (*5)	

Name	Description	Input Acceptance Condition (*1)
SetMotorsOn	Turn ON robot motors. (*5) (*6)	Ready output ON EStopOn output OFF SafeguardOn output OFF SetMotorsOff input OFF
SetMotorsOff	Turn OFF robot motors. (*5)	Ready output ON
SetPowerHigh	Set the robot power mode to High (*5)	Ready output ON EStopOn output OFF SafeguardOn output OFF SetPowerLow input OFF
SetPowerLow	Set the robot power mode to Low. (*5)	Ready output ON
Home	Move the Robot Arm to the home position defined by the user.	Ready output ON Error output OFF EStopOn output OFF SafeguardOn output OFF MotorsOn output ON Pause input OFF Stop input OFF
MCal	Execute MCal (*5) (*7)	Ready output ON Error output OFF EStopOn output OFF SafeguardOn output OFF MotorsOn output ON Pause input OFF Stop input OFF
Recover	After the safeguard is closed, recover to the position where the safeguard is open.	Paused output ON Error output OFF EStopOn output OFF SafeguardOn output OFF RecoverReqd output ON Pause input OFF Stop input OFF

(\*1) “AutoMode output” ON is omitted from the table. This is an input acceptance condition for all functions.

(\*2) “Start input” executes Function specified by the following six bits: SelProg 1, 2, 4, 8, 16, and 32.

Function name	SelProg1	SelProg2	SelProg4	SelProg8	SelProg16	SelProg32
Main	0	0	0	0	0	0
Main1	1	0	0	0	0	0
Main2	0	1	0	0	0	0
Main3	1	1	0	0	0	0
⋮						
Main60	0	0	1	1	1	1
Main61	1	0	1	1	1	1
Main62	0	1	1	1	1	1
Main63	1	1	1	1	1	1

0=OFF, 1=ON

(\*3) “NoPause task” and “NoEmgAbort task” do not pause.

For details, refer to *EPSON RC+ 6.0 Online Help* or *Pause* in *SPEL<sup>+</sup> Language Reference*.

(\*4) Turns OFF the I/O output and initializes the robot parameter.

For details, refer to *EPSON RC+ 6.0 Online Help* or *Reset* in *SPEL<sup>+</sup> Language Reference*.

(\*5) When specifying a robot, executes a function specified by the following bits: SelRobot 1, 2, 4, 8, and 16.

Robot number	SelRobot1	SelRobot2	SelRobot4	SelRobot8	SelRobot16
0 (All)	0	0	0	0	0
1	1	0	0	0	0
2	0	1	0	0	0
3	1	1	0	0	0
⋮					
13	1	0	1	1	0
14	0	1	1	1	0
15	1	1	1	1	0
16	0	0	0	0	1

0=OFF, 1=ON

(\*6) Initializes the robot parameter.

For details, refer to *EPSON RC+ 6.0 Online Help* or *Motor* in *SPEL<sup>+</sup> Language Reference*.

(\*7) For details, refer to *EPSON RC+ 6.0 Online Help* or *MCal* in *SPEL<sup>+</sup> Language Reference*.

(\*8) This is for experienced users only. Make sure that you fully understand the input specification before using.

CmdRunning output and CmdError output will not change for this input.

“NoEmgAbort task” will not stop by this input.

When the input changes from ON to OFF, all tasks and commands will stop.

(\*9) This function changes the output condition of MotorsOn, AtHome, PowerHigh, and MCalReqd.

By setting this signal with the condition selected using SelRobot1 - SelRobot16, you can switch the output condition.

Once you select the condition, it will be kept until you change it or turn off / restart the Controller. All manipulators are selected as default.

## 11.8 Remote Outputs

Remote outputs provide status for the Manipulator(s) and Controller.

Remote outputs provide the assigned function used with any control device. The outputs execute automatically. Therefore, no special programming is needed.

No remote control signals are configured at shipment time.

Name	Description
Ready	Turns ON when the controller startup completes and no task is running.
Running	Turns ON when task is running. However, turns OFF when "Paused output" is OFF.
Paused	Turns ON when pause task exists.
Error	Turns ON when an error occurs. Use "Reset input" to recover from the error.
EStopOn	Turns ON at Emergency Stop.
SafeguardOn	Turns ON when the safeguard is open.
SError	Turns ON when critical error occurs. When a critical error occurs, "Reset input" does not function. Reboot the controller to recover.
Warnig	Turns ON when warning occurs. The task runs as normal with the warning. However, be sure to eliminate the cause of the warning as soon as possible.
MotorsOn	Turns ON when the motor is ON. (*5)
AtHome	Turns ON when the robot is in the home position. (*5)
PowerHigh	Turns ON when the robot's power mode is High. (*5)
MCalReqd	Turns ON when the robot hasn't executed MCal. (*5)
RecoverReqd	Turns ON when at least one robot is waiting for Recover after the safeguard is closed.
RecoverInCycle	Turns ON when at least one robot is executing Recover.
WaitingRC	Turns ON when the controller is waiting to connect with the RC+.
CmdRunning	Turns ON when an input command is executing.
CmdError	Turns ON when an input command cannot be accepted.
CurrProg1 CurrProg2 CurrProg4 CurrProg8 CurrProg16 CurrProg32	Indicates the running or the last main function number (*1)
AutoMode	Turns ON in remote input acceptable status. (*2)
TeachMode	Turns ON in TEACH mode.
EnableOn	Turns ON when the Enable switch is ON.
ErrorCode1 ⋮ ErrorCode8192	Indicates the error number.
InsideBox1 ⋮ InsideBox15	Turns ON when the robot is in the approach check area. (*3)
InsidePlane1(*3) ⋮ InsidePlane15	Turns ON when a robot is on the approach plane area. (*4)

## 11. Remote Control

(\*1) Outputs the current or the last function number of CurrProg1, 2, 4, 8, 16, or 32.

Function name	CurrProg1	CurrProg2	CurrProg4	CurrProg8	CurrProg16	CurrProg32
Main	0	0	0	0	0	0
Main1	1	0	0	0	0	0
Main2	0	1	0	0	0	0
Main3	1	1	0	0	0	0
⋮						
Main60	0	0	1	1	1	1
Main61	1	0	1	1	1	1
Main62	0	1	1	1	1	1
Main63	1	1	1	1	1	1

0=OFF, 1=ON

(\*2) Remote function is available in the followings conditions.

- The setting is Auto mode and the control device is remote.
- The setting is Program mode and Remote I/O is enabled.

(\*3) For details, refer to EPSON RC+ 6.0 *Online Help* or *Box* in *SPEL<sup>+</sup> Language Reference*.

(\*4) For details, refer to EPSON RC+ 6.0 *Online Help* or *Plane* in *SPEL<sup>+</sup> Language Reference*.

(\*5) Manipulator status is output as follows, according to the condition selected in SelRobot.

Leave at least 40 ms to input the signal after changing the condition in SelRobot.

Name	(SelRobot1- SelRobot16) condition when inputting SelRobot	
	0: All robots are selected	1 - 16: Particular robot number is selected
MotorsOn	Turns ON when at least one motor is ON.	Turns ON when the motor of the selected robot is ON.
AtHome	Turns ON when all robots are in the home position.	Turns ON when the selected robot is in the home position.
PowerHigh	Turns ON when at least one robot's power mode is High.	Turns ON when the selected robot's power mode is High.
MCalReqd	Turns ON when at least one robot hasn't executed MCal.	Turns ON when the selected robot hasn't executed MCal.

## 11.9 Remote Input Handshake Timing

The following charts indicate the timing sequences for the primary operations of the Controller.

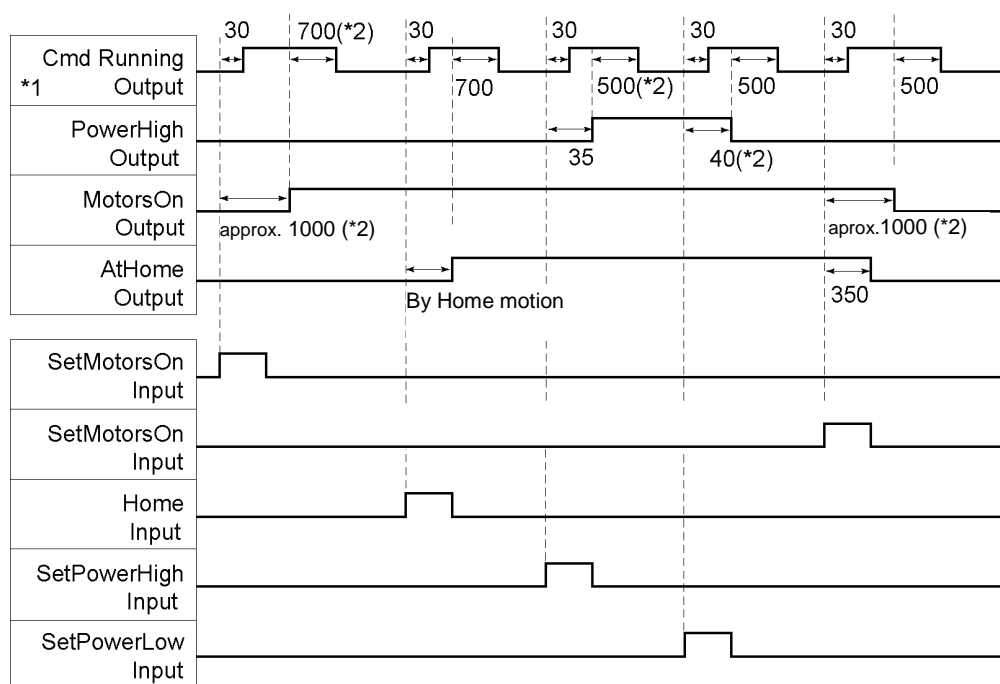
The indicated time lapses (time durations) should be referred to only as reference values since the actual timing values vary depending on some factors such as the numbers of manipulators and running tasks. Check carefully and refer to the following charts for the timing interrelation when you enter an input signal.

During system design, make sure that you actuate only one remote input operation at a time, otherwise an error will occur.

The pulse width of an input signal must be 25 or more milliseconds to be detected.

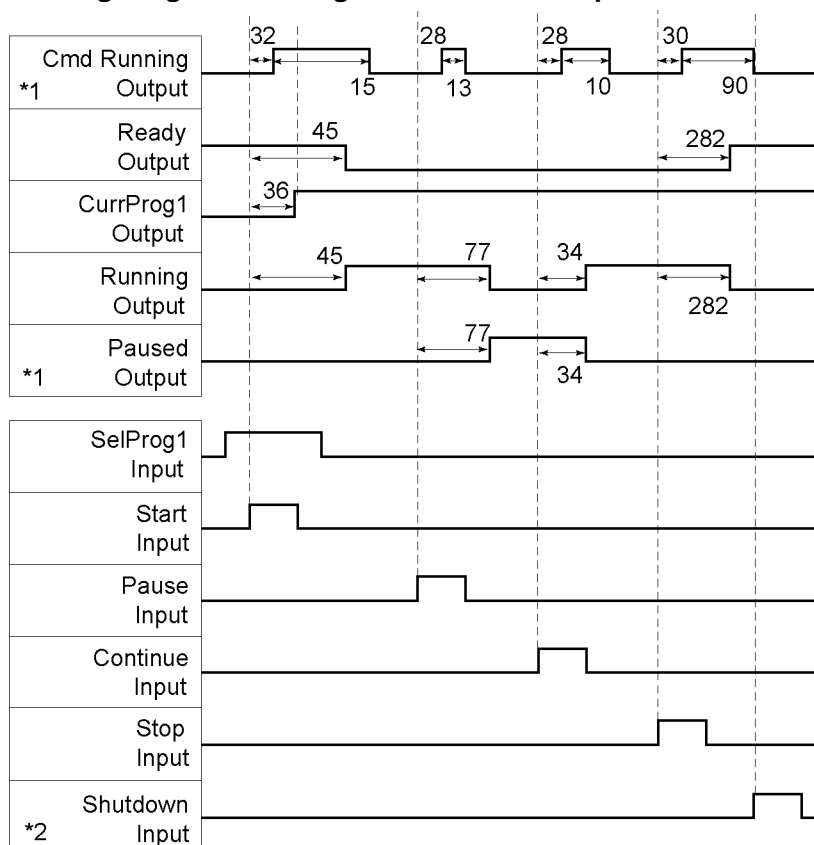
[Unit: msec]

**Timing Diagram for Operation Execution Sequence**



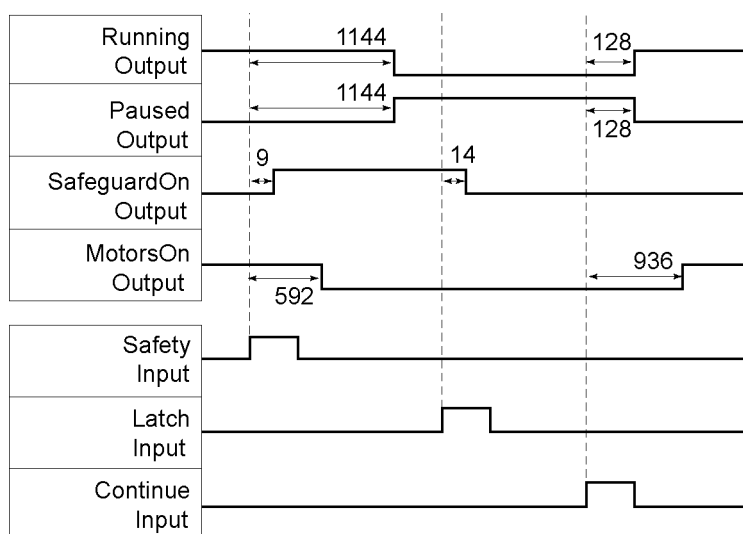
\*1 The motion of the CmdRunnig can be different from this figure according to the condition.

\*2 Refer to only as reference value for a robot. It can be different according to the number of robots.

**Timing Diagram for Program Execution Sequence**

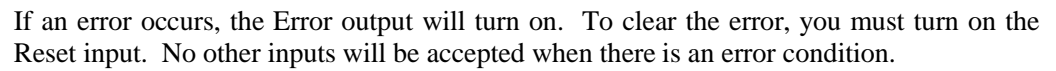
\*1 Differs according to the setting condition of Quick Pause and the program running condition when the PAUSE is input.

\*2 Shutdown input can be accepted when the Ready output is ON.

**Timing Diagram for Safety Door Input Sequence**



## 239



The diagram illustrates the timing sequence for the Recover function. It shows the states of various outputs and inputs over time. Vertical dashed lines mark specific time points. The sequence of events is as follows:

- Running Output** is initially high. It drops to low at time 1144. It remains low until time 31, when it returns to high.
- Paused Output** is initially high. It drops to low at time 1144. It remains low until time 31, when it returns to high.
- SafeguardOn Output** is initially low. It rises to high at time 9. It remains high until time 10, when it drops to low.
- RecoverReqd Output** is initially low. It rises to high at time 10. It remains high until time 133, when it drops to low.
- RecoverInCycle Output** is initially low. It rises to high at time 133. It remains high until time 133, when it drops to low. The duration of this pulse is labeled "By Recover motion".
- Safety Input** is initially low. It rises to high at time 1144. It remains high until time 10, when it drops to low.
- Latch Input** is initially low. It rises to high at time 10. It remains high until time 133, when it drops to low.
- Recover Input** is initially low. It rises to high at time 133. It remains high until time 133, when it drops to low.
- Continue Input** is initially low. It rises to high at time 133. It remains high until time 133, when it drops to low.

## 12. RS-232C Communications

The RC620 controller supports:

Windows part : Standard RS-232C port × 2  
(Standard: Port 1001 only, High-speed: Port 1001, 1002)

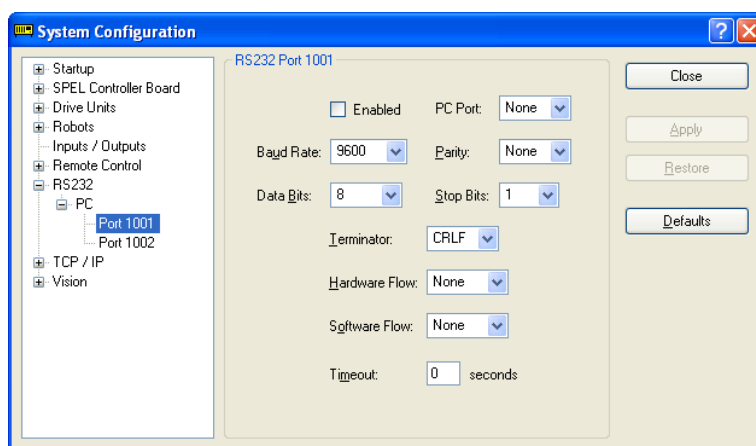
Expanded RS-232C : Option RS-232C port × 8 maximum (4 ports per board)

For instructions on how to install RS-232C boards, refer to the *RC620 Robot Controller manual*.

### 12.1 RS-232C Software Configuration

To configure a standard RS-232C port

1. Select **System Configuration** from the Setup Menu and open the dialog.



2. Select RS-232C | PC from the tree on the left.
3. Set the [Enabled] check box.
4. Change the settings as desired.
5. Click **Apply** to save the new settings.
6. Click **Close**.

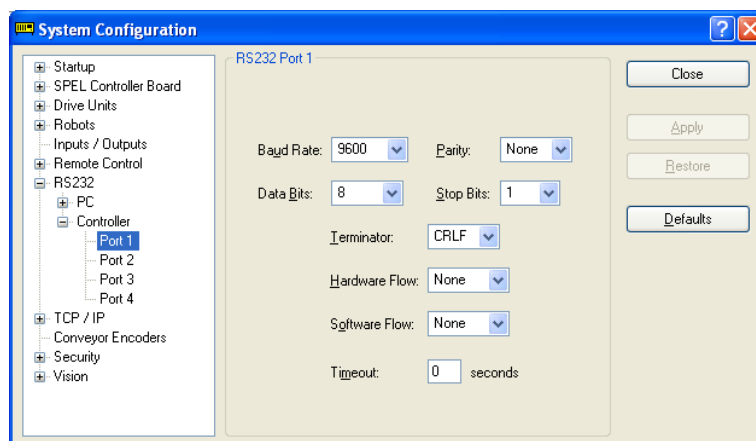


NOTE

If several ports are used for communication at one time with more than a 19200 baud rate, error 2929 or 2922 may occur. In this case, select a lower baud rate or avoid using several ports at one time.

To configure an option RS-232C port

1. Select **System Configuration** from the Setup Menu and open the dialog.



2. Select the RS-232C | Controller from the tree on the left.



If no RS-232C boards have been installed on the controller, you will not see the corresponding ports in the tree.

3. Select a port to configure.
4. Change the settings as desired.
5. Click **Apply** to save the new settings.
6. Click **Close**.

## 12.2 RS-232C Commands

The following is a list of all of the commands associated with RS-232C communications. For details, please see the online help or SPEL<sup>+</sup> Language Reference Manual.

<b>OpenCom</b>	Opens a communications port.
<b>ChkCom</b>	Returns port status: the number of bytes waiting to be read or error condition.
<b>CloseCom</b>	Closes a communications port.
<b>SetCom</b>	Sets communications port parameters at runtime or from the Command window.
<b>Print #</b>	Sends characters out of the port.
<b>Input #</b>	Receives characters from the port into one or more variables.
<b>Line Input #</b>	Receives one line characters from the port into one string variable.
<b>Read #</b>	Receives one or more characters from the port into one string variable.
<b>ReadBin #</b>	Receives one or more bytes from the port.
<b>Write #</b>	Sends characters out of the port.
<b>WriteBin #</b>	Sends one or more bytes out of the port.

## 13. TCP / IP Communications

EPSON RC+ 6.0 supports 16 TCP/IP ports that allow peer to peer communications. This chapter contains instructions on using TCP/IP, including IP addresses of LAN-1 port and Windows TCP/IP configuration.



- LAN-2 is not available for peer to peer communications of EPSON RC+ 6.0. For details, refer to *RC620 Controller manual: Setup & Operation 6. LAN (Ethernet Communication) Port*.

### 13.1 TCP/IP Setup

Before you can use TCP/IP communications between PCs and controllers, you must configure your network. The following sections describe basic network configuration.

#### 13.1.1 Ethernet Hardware

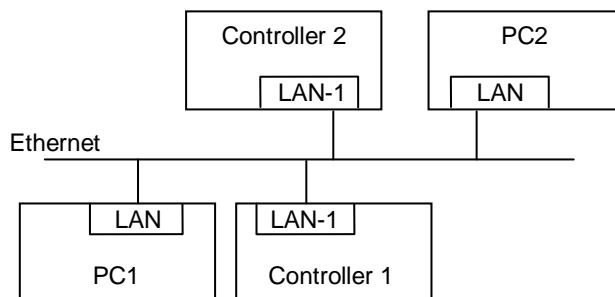
The RC620 controller includes a built in Ethernet interface with an RJ45 connector accessible from the controller rear panel. It supports 10BaseT (10 Mbps) and 10BaseTX (100 Mbps).

Your PC will need a 10BaseT 10/100 adapter to communicate with an RC620 controller via Ethernet.

#### 13.1.2 IP Addresses

The controller has a fixed IP address that you can configure from EPSON RC+ 6.0. To configure the IP address, mask, and gateway for the controller, refer to *5.12.1 System Configuration - System Configuration / Controller / Configuration*.

The following table shows a typical IP address configuration.



Host Name	IP Address	Subnet	Subnet Mask
PC1	192.168.0.1	192.168.0	255.255.255.0
PC2	192.168.0.2	192.168.0	255.255.255.0
Controller1	192.168.0.3	192.168.0	255.255.255.0
Controller2	192.168.0.4	192.168.0	255.255.255.0

In this example, the network address (subnet) is 192.168.0. With a subnet mask of 255.255.255.0, there can be 254 hosts on this subnet (0 and 255 cannot be used).

Refer to the Microsoft Windows operating system manual for instructions on setting the PC IP address.

### 13.1.3 IP Gateway

If you are connecting PCs and controllers on different networks, you will need to route traffic between the networks using one or more routers. Each device communicating via Ethernet will need to have their default gateway address set to the address of the router for its subnet.

To configure the gateway address for the controller, refer to *5.12.1 System Configuration - System Configuration / Controller / Configuration*.

### 13.1.4 Testing Windows TCP/IP setup

Use the ping command from a Command Window to test communications.

First, do a loopback test to check if you can ping your own address by using the local IP address:

```
C:\>ping 127.0.0.1
Pinging 127.0.0.1 with 32 bytes of data:
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
Reply from 127.0.0.1: bytes=32 time<10ms TTL=128
C:\>
```

Ping your PC's IP address:

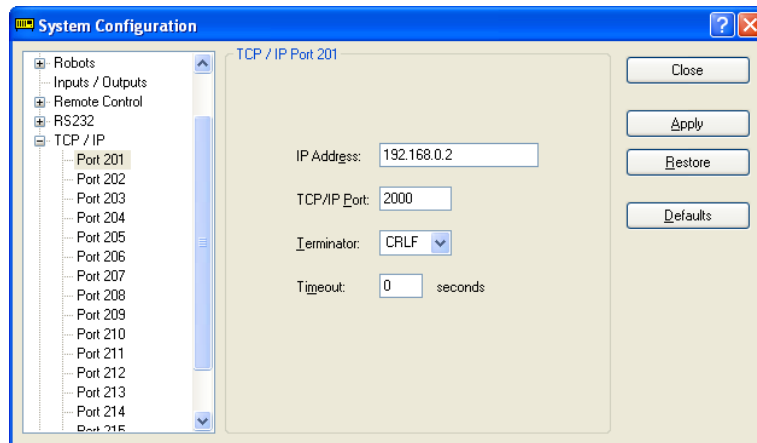
```
C:\>ping 192.168.0.1
Pinging 192.168.0.1 with 32 bytes of data:
Reply from 192.168.0.1: bytes=32 time<10ms TTL=128
Reply from 192.168.0.1: bytes=32 time<10ms TTL=128
Reply from 192.168.0.1: bytes=32 time<10ms TTL=128
Reply from 192.168.0.1: bytes=32 time<10ms TTL=128
C:\>
```

Now ping controller on the network. :

```
C:\>ping 192.168.0.3
Pinging pc2 [192.168.0.3] with 32 bytes of data:
Reply from 192.168.0.3: bytes=32 time<10ms TTL=128
Reply from 192.168.0.3: bytes=32 time<10ms TTL=128
Reply from 192.168.0.3: bytes=32 time<10ms TTL=128
Reply from 192.168.0.3: bytes=32 time<10ms TTL=128
C:\>
```

## 13.2 TCP/IP Software Configuration

You can configure TCP/IP settings for the controller in a SPEL<sup>+</sup> program using the SetNet command. You can also configure settings from the TCP/IP tab on the Setup | System Configuration dialog.



To configure a TCP/IP port

1. Select System Configuration from the Setup Menu and select the page for the TCP/IP port you want to configure.
2. Enter the IP address for the controller or PC that you want this controller to communicate with.

The controller does not support DNS, so you must specify an IP address for the host you are communicating with. You cannot specify a name for the host.

3. Enter the TCP/IP port number. This must be the same port number that is used on the host device. It must be different from any of the other TCP/IP port numbers used for the other TCP/IP ports.
4. Change the other settings as desired.
5. Click **Apply** to save the new settings and click **Close**.

## 13.3 TCP/IP Commands

Here is a list of all of the commands associated with TCP/IP communications. For details, please see the online help or SPEL<sup>+</sup> Language Reference Manual.

<b>OpenNet</b>	Opens a TCP/IP port.
<b>ChkNet</b>	Returns port status: the number of bytes waiting to be read or error condition.
<b>CloseNet</b>	Closes a TCP/IP port.
<b>SetNet</b>	Sets communications port parameters at runtime or from the Command window.
<b>Print #</b>	Sends characters out of the port.
<b>Input #</b>	Receives characters from the port into one or more variables.
<b>Line Input #</b>	Receives one line characters from the port into one string variable.
<b>Read #</b>	Receives one or more characters from the port into one string variable.
<b>ReadBin #</b>	Receives one or more bytes from the port.
<b>Write #</b>	Sends characters out of the port.
<b>WriteBin #</b>	Sends one or more bytes out of the port.

## 14. Security

### 14.1 Overview

The EPSON RC+ Security Option allows you to manage EPSON RC+ 6.0 users and also monitor usage.

When the Security Option is activated, administrators can add groups and users. Each group can have one or more rights associated with it. For example, you can create a group called Maintenance that has rights to edit robot points, use Jog & Teach, and enable you to use the Command Window. When a user attempts to do something that he/she does not have a right for, a message "Permission denied" will be displayed.

Each login session is recorded in a Microsoft Access compatible data base. Security Log Viewer is included that allows you to view each session's activity.

User can login to EPSON RC+ with a name and password. Optionally, EPSON RC+ can use the Windows user name to log in automatically.

### 14.2 Installation

Your RC620 controller must have the Security Option enabled. You can enable the option by calling your distributor and purchasing the Security Option. Refer to *Installing EPSON RC+ Options* for details.

Normally this is installed at the factory if the Security Option is purchased with the system.

### 14.3 Security Configuration

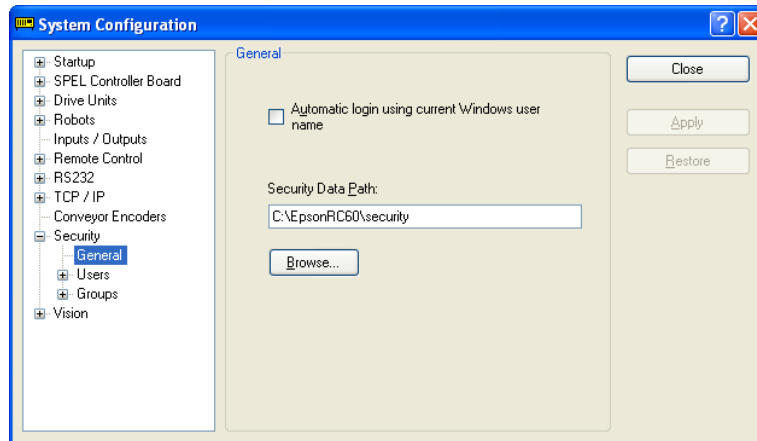
EPSON RC+ 6.0 requires a path for security files. If you have more than one system on a network, it is recommended that you setup the security files path for all systems to store the security logs in a server on the network.

To administer EPSON RC+ 6.0 security:

1. Start EPSON RC+ 6.0.
2. Select Setup | System Configuration.
3. Click on the Security tree.  
If the Security tree is not displayed, check your software key.
4. On the General tree, type in the path for your security files or click the Browse button.
5. Click on the Users tree.
6. For each user on your system, click **New** button.  
Each new user belongs to the Guest group by default. Click in the group field, then click the dropdown button to select the desired group.

## General Tab

This tab allows you to configure the general security settings.  
It will be disabled if the Security Option is not activated.



### Automatic log in using current Windows user name

Check this box if you want EPSON RC+ 6.0 to use the current Windows login ID. When the Security Option is active, you will not see a login dialog when EPSON RC+ starts, unless EPSON RC+ cannot find the user in the Security system.

### Security data path

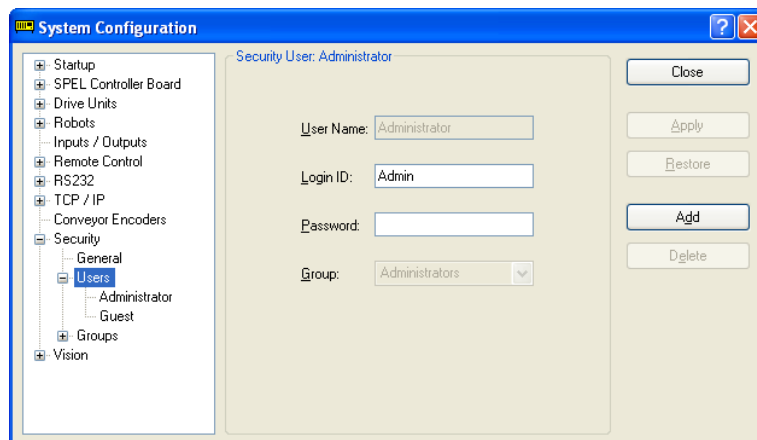
This is the path where security files will be stored.

This path should be protected with Windows security rights so that only Administrators can delete the files in this path. All other EPSON RC+ users should have only read rights to the files in this path.

## User Page

This page allows you to add and remove EPSON RC+ 6.0 users.

Two users are permanent: Administrator and Guest. Only the passwords can be changed for these users. You should always use a password for the Administrator, though no password is set at shipment time.



### To add a user

1. Click the **Add** button.
2. A new user will be added to the tree.
3. Click the Group tree for the new user.
4. Click the dropdown button and select the group for the user.



**To delete a user**

1. Click the User you want to delete in the tree.
2. Click the **Delete** button.
3. A confirmation message to delete the user will appear.

**To change a user's group**

1. Click the Group dropdown for the user you want to change.
2. Click a dropdown button in the field and select a new group.

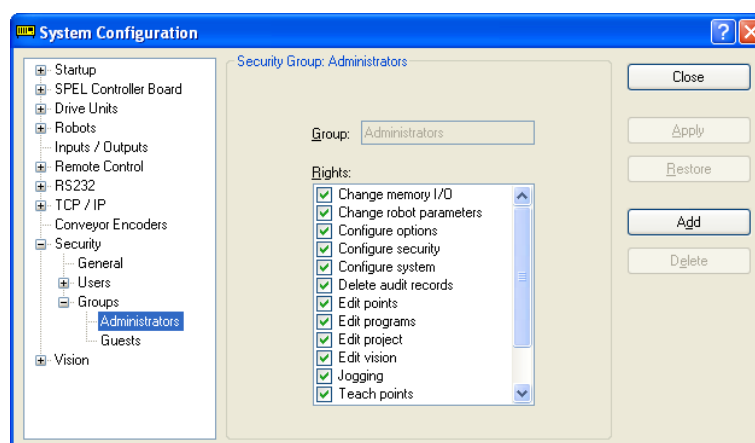
**Editing Name, Login ID, and Password**

1. Click a **User** you want to change.
2. Edit the field. All fields are not case sensitive.

**Group Page**

This page allows you to configure user groups. Every EPSON RC+ 6.0 user must belong to a group.

Two groups cannot be deleted or modified: Administrators and Guests. Administrators have full rights, and Guests have no rights.

**To add a group**

1. Click the **Add** button.
2. Type in a name for the group.
3. Click the **Apply** button.

**To delete a group**

1. Select the group you want to delete.
2. Click the **Delete** button.
3. A confirmation message to delete the group will appear.

**To change rights for groups**

1. Select the group you want to change rights for.  
Note that you cannot change rights for Administrators and Guests.
2. To add a right, set the checkbox(es) for the desired rights in the [Rights] checkbox list.
3. To remove rights, clear the checkbox(es) for the rights you want to remove in the [Right] checkbox list.

### Group Rights

The list below shows the rights that are available for user groups. Administrators have full rights, and Guests have no rights.

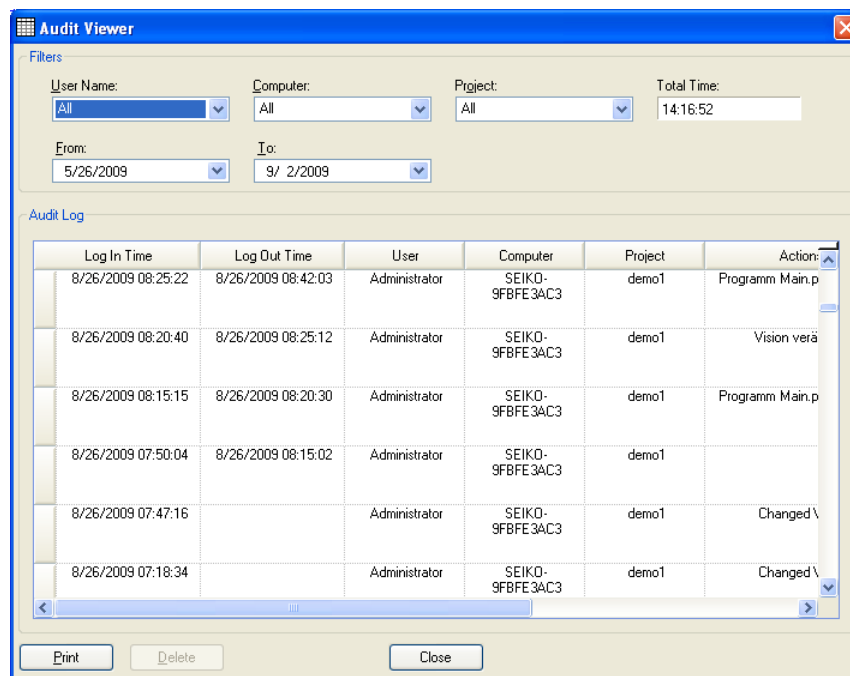
Right	Description
Configure options	Users can change option settings in Setup   Options.
Use Command Window	Users can open the command window and execute commands.
Configure system	Users can configure the entire EPSON RC+ system.
Jog	Users can open the Jog & Teach dialog and jog a robot.
Check security log	Users can see security logs.
Delete security log	Users can delete security logs in Tools   Audit Viewer.
Teach points	Users can teach points and delete points from the Jog & Teach dialog.
Edit vision	Users can edit vision parameters.
Edit programs	Users can edit program.
Edit projects	Users can edit projects.
Edit points	Users can change points.
Change memory I/O	Users can turn ON/OFF memory I/O bits.
Change robot parameter	Users can open the Robot Manager dialog and change the settings.
Output port ON	Users can turn ON/OFF outputs.

## 14.4 Security Audit Viewer

When the Security Option is enabled, EPSON RC+ 6.0 will keep track of who logs into the system and actions performed.

Activity is stored to the security data path in the Microsoft Access compatible data base format.

To view the security logs, select Audit Viewer from the Tools Menu.



## 14.5 SPEL<sup>+</sup> Security Command

Here are the SPEL<sup>+</sup> commands that are enabled with the Security Option. For details, please see the *EPSON RC+ 6.0 Online Help* or *SPEL+ Language Reference manual*.

Command	Description
<b>LogIn Function</b>	Logs in the application as another user at runtime.
<b>GetCurrentUser\$ Function</b>	Returns the login ID of the current user.

## 15. Conveyor Tracking

### 15.1 Overview

Conveyor Tracking is a process in which a robot picks up parts from a stationary or moving conveyor that are found by a vision system or sensor.

The EPSON RC+ 6.0 Conveyor Tracking option supports both tracking and indexed conveyor systems.

- **Tracking conveyor system**

Conveyor moves constantly. Vision system or sensor system finds the parts on it and robot picks them up as they move. During tracking, the robot can move along with the part as it picks up parts.

- **Indexed conveyor system**

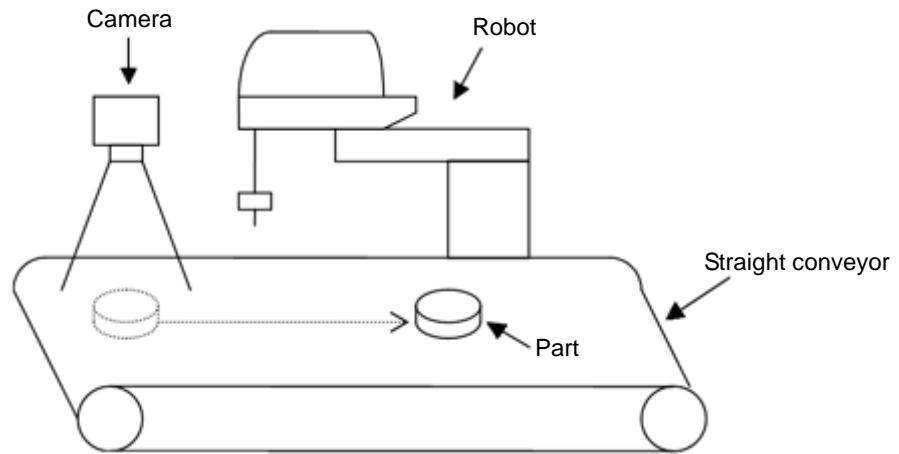
Conveyor moves a specified distance and then stops. The vision system finds the parts and robot picks up each part. After finding and picking up all parts, the conveyor moves again.

A total of 16 physical conveyors can be defined on each system. A physical conveyor has one encoder whose signals are received by an encoder board.

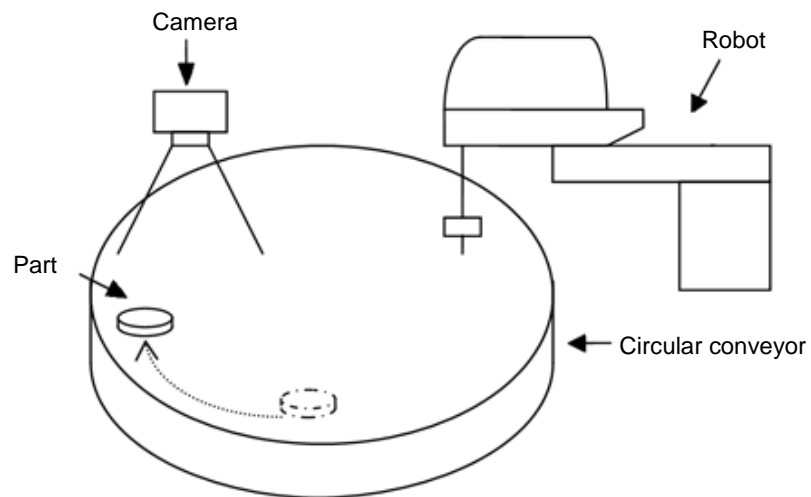
Up to 16 logical conveyors can be defined in each project. To define a logical conveyor, set a conveyor number, a robot number, encoder number and select vision or sensor.

Multiple robots and multiple conveyors are supported. For example, 2 robots can be used to pick parts from one physical conveyor.

The Conveyor Tracking option is available for straight conveyors and circular conveyors, as shown in the figures below. These conveyors have different calibration and programming methods. For details, refer to *15.11 Vision Conveyors* and *15.12 Sensor Conveyors*.



Straight conveyor tracking system



Circular conveyor tracking system

## 15.2 Conveyor Tracking Processes

### Tracking conveyor system

1. Vision system or sensor system finds the parts on a continuously moving conveyor.
2. Robot picks up the parts on the conveyor as they move.

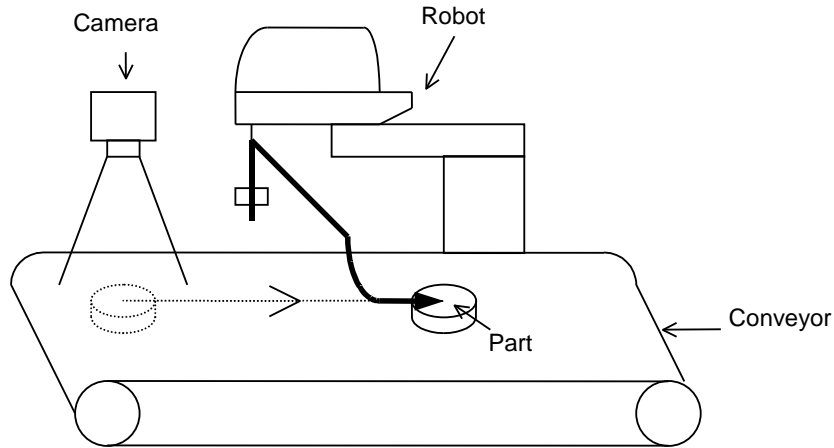


Fig 2.1 Tracking Conveyor System

### Indexed conveyor system

1. Conveyor moves a specified distance.
2. Vision system or sensor system finds the parts on the conveyor when it stops.
3. Robot picks up the parts found by vision system.
4. After finding and picking up all parts, conveyor moves by the specified distance again.

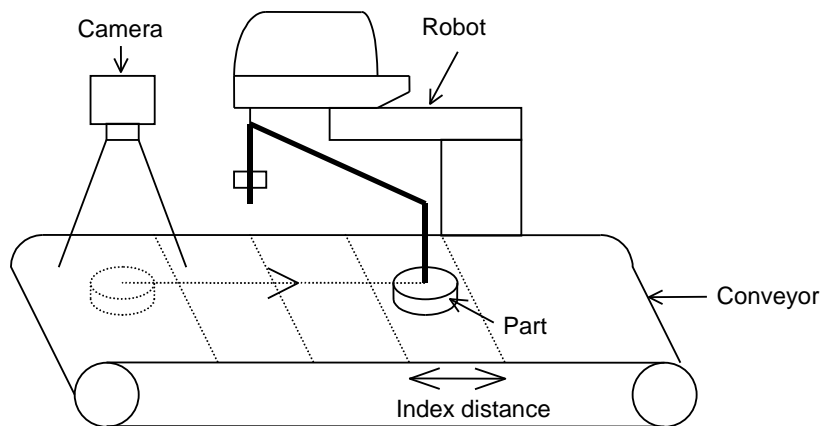


Fig 2.2 Indexed Conveyor System

## 15.3 Hardware Installation

To use conveyor tracking, you must install encoders for each physical conveyor on the system. Each encoder is wired to a single channel on a PG (Pulse Generator) board. Each board can accommodate up to 4 encoders. A trigger input is also provided for each encoder to latch position, such as when used with a strobed vision camera.

### PG board specifications

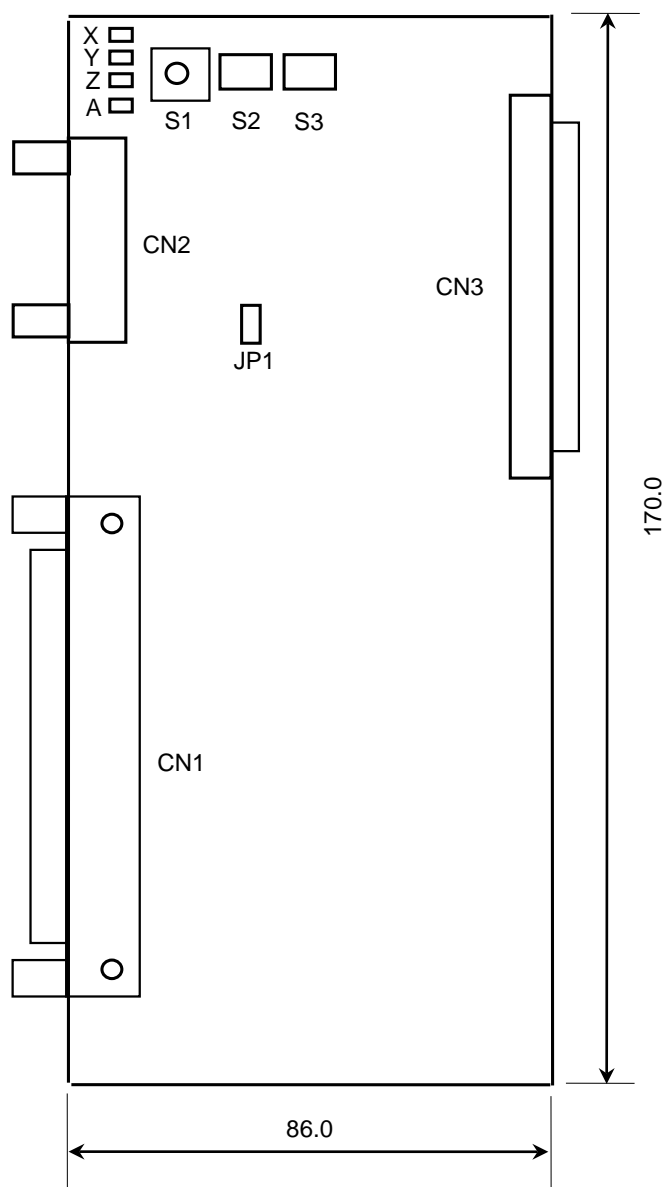
The table below shows the specification for the PG board.

<b>Board Name</b>	H745
<b>Compatible Controller</b>	RC620
<b>Board Extension Capability</b>	4 boards maximum
<b>Encoder channels</b>	4 channels / board
<b>Encoder Type</b>	ABZ phase differential input (RS-422 line receiver)
<b>Input Pulse Rate</b>	Max. 5 MPPS
<b>Input Signal</b>	Conveyor pulse latch input
<b>Board Address</b>	Set the DIP switch according the board number. (See DIP Switch Settings later in this chapter).
<b>connector</b>	DX10A - 100S (Hirose Electric Co.,Ltd.)
<b>Power Supply</b>	24V $\pm$ 2V 200mA or under

The following encoder models have been tested:

OMRON	E6B2-CWZ1X
TAMAGAWA	TS5312N512-2000C/T

The figure below shows the layout of the PG board.





### DIP switch settings

The board address is set by DIP switch (S2, S3) on the PG Board according to the board number, as shown in the following table.

Board #	Address	S2				S3			
		1 (A15)	2 (A14)	3 (A13)	4 (A12)	1 (A11)	2 (A10)	3 (A9)	4 (A8)
1	1000h	OFF	OFF	OFF	ON	OFF	OFF	OFF	ON
2	1100h	OFF	OFF	OFF	ON	OFF	OFF	ON	OFF
3	1200h	OFF	OFF	OFF	ON	OFF	OFF	ON	ON
4	1300h	OFF	OFF	OFF	ON	OFF	ON	OFF	OFF

If you purchased the PG board separately, place the attached Board No. Label sticker on the board panel prior to installation of the board in the Control Unit and keep a written record of the address setting and the board number.

If you have purchased the PG Board with the Control Unit, the board address has been set properly before shipment and further settings should not be necessary.

### Jumper settings

The jumpers are reserved and should not be changed.

### Rotary switch settings

The rotary switch S1 is reserved and should not be changed.

S1 : Position of 1

### Signal Connections

The table below lists the connectors on the PG board and the compatible connectors for wiring:

Receptacle on the Board		DXA10A-100S (manufacturer: Hirose Electric Co.,Ltd.)
Wiring Plug Connectors	Individually pressed-in type	DX30-100P (for AWG#30) DX30A-100P (for AWG#28)
	Pressed-in-as-a-whole type	DX31-100P (for AWG#30) DX31A-100P (for AWG#28)
	Soldered type	DX40-100P
Connector for Wiring to the Cover		DX-100-CV1

## Signal Assignments: PG board connector (DX10A-100S)

The signals on the PG board connector are assigned as shown in the table below.

Pin	Dir	Signal	Description	Pin	Dir	Signal	Description
1	-	-	Not used	51	-	-	Not used
2	-	-	Not used	52	-	-	Not used
3	-	-	Not used	53	-	-	Not used
4	-	-	Not used	54	-	-	Not used
5	-	-	Not used	55	-	-	Not used
6	-	-	Not used	56	-	-	Not used
7	-	-	Not used	57	-	-	Not used
8	-	-	Not used	58	-	-	Not used
9	-	-	Not used	59	-	-	Not used
10	In	TRG1	Trigger input for Counter1	60	-	-	Not used
11	In	TRG2	Trigger input for Counter2	61	-	-	Not used
12	In	TRG3	Trigger input for Counter3	62	-	-	Not used
13	In	TRG4	Trigger input for Counter4	63	-	-	Not used
14	In	EXTV	External power supply for Input circuit	64	In	EXTV GND	External power supply GND for Input circuit
15	In	EXTV	External power supply for Input circuit	65	In	EXTV GND	External power supply GND for Input circuit
16	-	-	Not used	66	-	-	Not used
17	-	-	Not used	67	-	-	Not used
18	-	-	Not used	68	-	-	Not used
19	-	-	Not used	69	-	-	Not used
20	-	-	Not used	70	-	-	Not used
21	-	-	Not used	71	-	-	Not used
22	-	-	Not used	72	-	-	Not used
23	-	-	Not used	73	-	-	Not used
24	-	-	Not used	74	-	-	Not used
25	In	+A1	Phase +A signal for Counter 1	75	In	+A3	Phase +A signal for Counter 3
26	In	-A1	Phase -A signal for Counter 1	76	In	-A3	Phase -A signal for Counter 3
27	In	+B1	Phase +B signal for Counter 1	77	In	+B3	Phase +B signal for Counter 3
28	In	-B1	Phase -B signal for Counter 1	78	In	-B3	Phase -B signal for Counter 3
29	In	+Z1	Phase +Z signal for Counter1	79	In	+Z3	Phase +Z signal for Counter 3
30	In	-Z1	Phase -Z signal for Counter 1	80	In	-Z3	Phase -Z signal for Counter 3
31	-	-	Not used	81	-	-	Not used
32	-	-	Not used	82	-	-	Not used
33	-	-	Not used	83	-	-	Not used
34	-	-	Not used	84	-	-	Not used
35	-	-	Not used	85	-	-	Not used
36	-	-	Not used	86	-	-	Not used
37	-	-	Not used	87	-	-	Not used
38	-	-	Not used	88	-	-	Not used
39	-	-	Not used	89	-	-	Not used
40	-	-	Not used	90	-	-	Not used
41	In	+A2	Phase +A signal for Counter 2	91	In	+A4	Phase +A signal for Counter 4
42	In	-A2	Phase -A signal for Counter 2	92	In	-A4	Phase -A signal for Counter 4
43	In	+B2	Phase +B signal for Counter 2	93	In	+B4	Phase +B signal for Counter 4

Pin	Dir	Signal	Description	Pin	Dir	Signal	Description
44	In	-B2	Phase -B signal for Counter 2	94	In	-B4	Phase -B signal for Counter4
45	In	+Z2	Phase +Z signal for Counter 2	95	In	+Z4	Phase +Z signal for Counter 4
46	In	-Z2	Phase -Z signal for Counter 2	96	In	-Z4	Phase -Z signal for Counter 4
47	-	-	Not used	97	-	-	Not used
48	-	-	Not used	98	-	-	Not used
49	-	-	Not used	99	-	-	Not used
50	-	GND	GND	100	-	GND	GND

**Pin # 25 ~ 30, 41 ~ 46, 75 ~ 80, 91 ~ 96**

Connect the pin numbers shown above with encoder output (+A, -A, +B, -B, +Z, -Z).

**Pins # 10 ~ 13**

When the conveyor pulse is latched by external signal, connect the pin numbers shown above with latch signal. Exactly when the signal is turned OFF to ON, the encoder pulse is latched.

**Pins # 14, 15, 64 and 65**

When using the pin # 10 ~ 13, connect external power with the pin numbers shown above.

When not using the pin # 10 ~ 13, it is not necessary to connect external power with the pin numbers shown above.

## Signal Assignments: PG board connector terminal block 1

The signals on the PG board connector terminal block #1 are assigned as shown in the table below. The pin numbers in parentheses are the pins on the PG board connector.

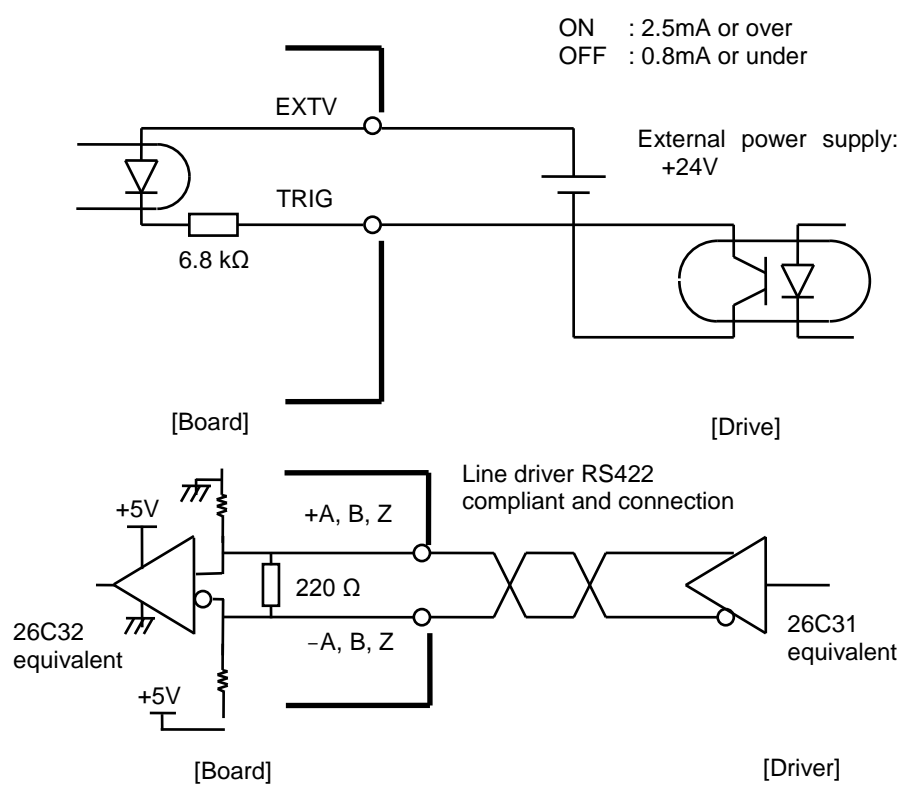
Pin	Signal	Description	Pin	Signal	Description
1 (16)	-	Not used	26 (32)	-	Not used
2 (17)	-	Not used	27 (33)	-	Not used
3 (18)	-	Not used	28 (34)	-	Not used
4 (19)	-	Not used	29 (35)	-	Not used
5 (20)	-	Not used	30 (36)	-	Not used
6 (21)	-	Not used	31 (37)	-	Not used
7 (22)	-	Not used	32 (38)	-	Not used
8 (23)	-	Not used	33 (39)	-	Not used
9 (24)	-	Not used	34 (40)	-	Not used
10 (25)	+A1	Phase +A signal for Counter 1	35 (41)	+A2	Phase +A signal for Counter 2
11 (26)	-A1	Phase -A signal for Counter 1	36 (42)	-A2	Phase -A signal for Counter 2
12 (27)	+B1	Phase +B signal for Counter 1	37 (43)	+B2	Phase +B signal for Counter 2
13 (28)	-B1	Phase -B signal for Counter 1	38 (44)	-B2	Phase -B signal for Counter 2
14 (29)	+Z1	Phase +Z signal for Counter 1	39 (45)	+Z2	Phase +Z signal for Counter 2
15 (30)	-Z1	Phase -Z signal for Counter 1	40 (46)	-Z2	Phase -Z signal for Counter 2
16 (31)	-	Not used	41 (47)	-	Not used
17 (48)	-	Not used	42 (49)	-	Not used
18 (9)	-	Not used	43 (50)	GND	Ground
19 (60)	-	Not used	44 (61)	-	Not used
20 (10)	TRG1	Trigger input for Counter 1	45 (11)	TRG2	Trigger input for Counter 2
21 (1)	-	Not used	46 (5)	-	Not used
22 (2)	-	Not used	47 (6)	-	Not used
23 (3)	-	Not used	48 (7)	-	Not used
24 (4)	-	Not used	49 (8)	-	Not used
25 (14)	EXTV	External power supply	50 (64)	EXTV GND	External power supply ground

### Signal Assignments: PG board connector terminal block 2

The signals on the PG board connector terminal block #2 are assigned as shown in the table below. The pin numbers in parentheses are the pins on the PG board connector.

Pin	Signal	Description	Pin	Signal	Description
1 (66)	-	Not used	26 (82)	-	Not used
2 (67)	-	Not used	27 (83)	-	Not used
3 (68)	-	Not used	28 (84)	-	Not used
4 (69)	-	Not used	29 (85)	-	Not used
5 (70)	-	Not used	30 (86)	-	Not used
6 (71)	-	Not used	31 (87)	-	Not used
7 (72)	-	Not used	32 (88)	-	Not used
8 (73)	-	Not used	33 (89)	-	Not used
9 (74)	-	Not used	34 (90)	-	Not used
10 (75)	+A3	Phase +A signal for Counter 3	35 (91)	+A4	Phase +A signal for Counter 4
11 (76)	-A3	Phase -A signal for Counter 3	36 (92)	-A4	Phase -A signal for Counter 4
12 (77)	+B3	Phase +B signal for Counter 3	37 (93)	+B4	Phase +B signal for Counter 4
13 (78)	-B3	Phase -B signal for Counter 3	38 (94)	-B4	Phase -B signal for Counter 4
14 (79)	+Z3	Phase +Z signal for Counter 3	39 (95)	+Z4	Phase +Z signal for Counter 4
15 (80)	-Z3	Phase -Z signal for Counter 3	40 (96)	-Z4	Phase -Z signal for Counter 4
16 (81)	-	Not used	41 (97)	-	Not used
17 (98)	-	Not used	42 (99)	-	Not used
18 (59)	-	Not used	43 (100)	GND	Ground
19 (62)	-	Not used	44 (63)	-	Not used
20 (12)	TRG3	Trigger input for Counter 3	45 (13)	TRG4	Trigger input for Counter 4
21 (51)	-	Not used	46 (55)	-	Not used
22 (52)	-	Not used	47 (56)	-	Not used
23 (53)	-	Not used	48 (57)	-	Not used
24 (54)	-	Not used	49 (58)	-	Not used
25 (15)	EXTV	External power supply	50 (65)	EXTV GND	External power supply ground

## Encoder Input Circuit



## 15.4 System Structure

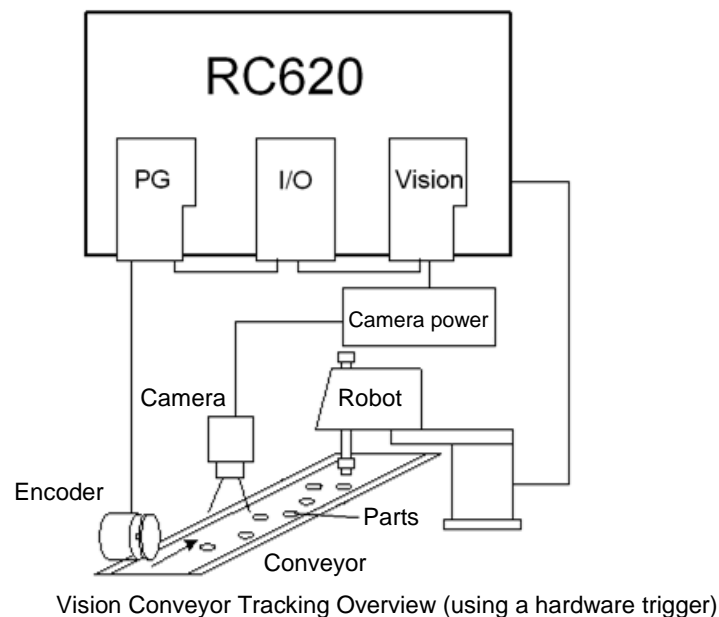
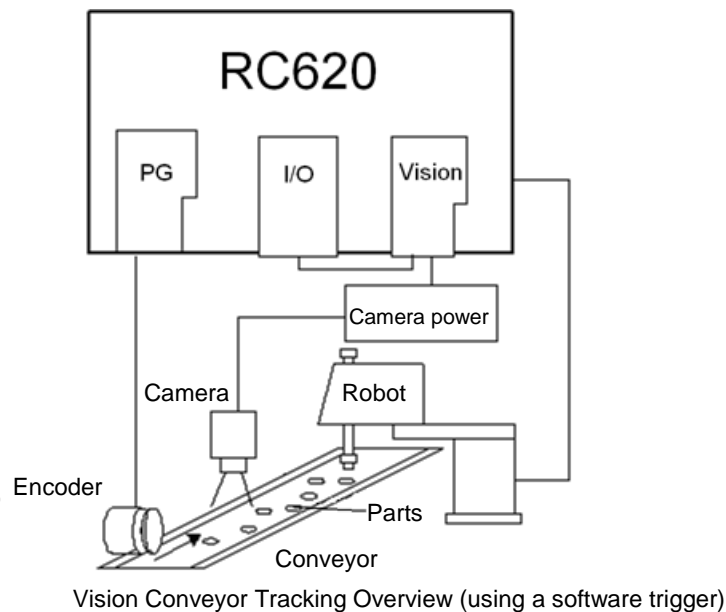
### Structure of Vision Conveyor Tracking System

The structure of a vision conveyor tracking system is shown in the figures below.

For this system, you need to set the same timing for the vision system to search for parts on the conveyor and for the encoder on the conveyor to latch position. To set the same timing, use the asynchronous reset mode in the vision system (if you don't use asynchronous reset mode, the timing of image acquisition is different from the encoder latch timing and the pickup precision will decrease).

Asynchronous reset mode allows the camera to acquire an image at the moment of trigger input and transfers the image to the vision sequence.

This section inputs the trigger using I/O and shows the wiring example using a frame grabber camera in the vision system.



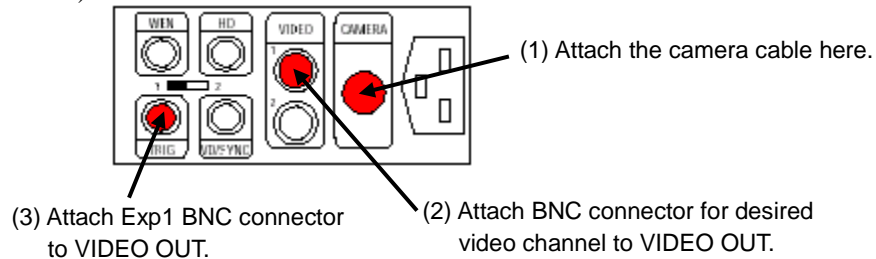
### Wiring Example of Vision System

The wiring of the vision system (using a frame grabber camera) is described below.

#### Vision System Wiring

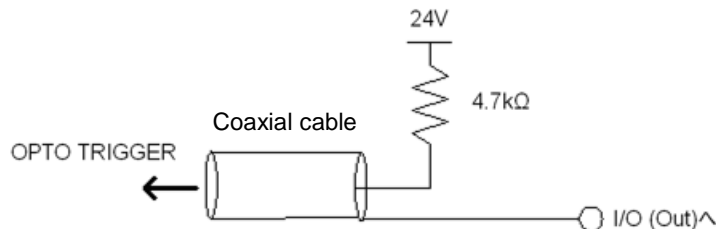
(Example: Sony XC-HR50)

- (1) Attach one end of the camera 1 cable to the camera to be used as camera 1 and the other end of the cable to the front of the Camera Power Junction box (see the figure below).



DC-700 Camera Power Junction Box (Rear View)

- (2) Pick up the BNC to D-Sub cable that is now attached to the frame grabber. Locate the cable to the Camera Power Junction box. Attach the Video 1 BNC connector cable to the VIDEO OUT BNC female connector on the Camera Power Junction box.
- (3) Attach the Exp 1 BNC connector cable to the TRIG connector on the Camera Power Junction box.
- (4) Connect the OPTO TRIGGER BNC connector cable and the coaxial cable as shown below.



- (5) Refer to the option manual, *Vision Guide 6.0 - Appendix A: Camera Interfaces* and set the camera external setting to asynchronous reset mode.

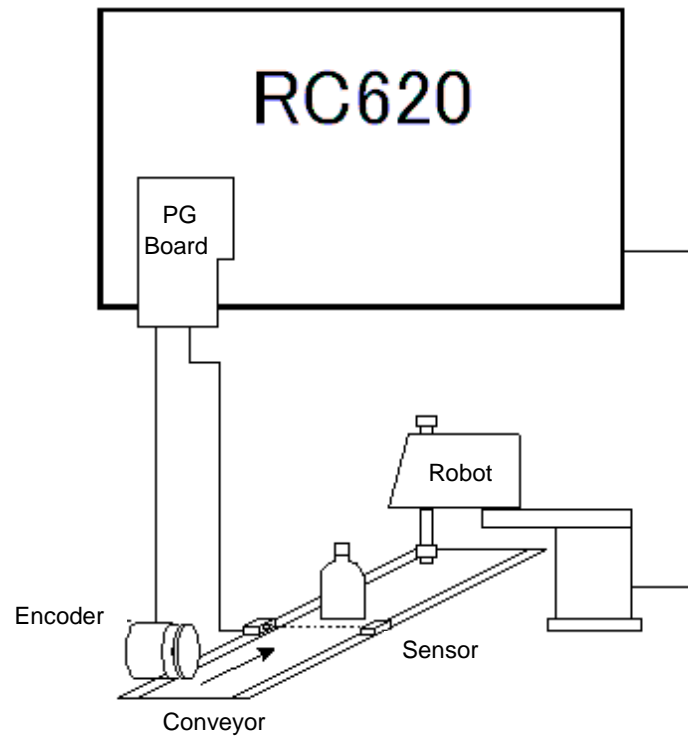


When you use the other cameras such as Smart Camera, refer to *Vision Guide 6.0 - 2. Installation and Appendix A: Camera Interface* to attach the cables.



### Structure of Sensor Conveyor Tracking System

The structure of Sensor Conveyor Tracking System is shown in the figure below. This system uses a hardware trigger. The hardware trigger signals the counter trigger input on the PG board and latches the encoder on the conveyor using the signals from the sensor or I/O.



Sensor Conveyor Tracking Overview

### Wiring of PG Board

The following describes the procedures to connect the encoder to the Axis #1 and to use the hardware trigger.

- (1) Encoder wiring  
Connect the encoder output +A, -A, +B, -B, +Z, -Z to pins 25 to 30.
- (2) Hardware trigger wiring  
Connect pin 14 and I/O (Out). For sensor conveyor tracking, connect pin 14 to the sensor trigger.  
Connect the 24V external power supply to the pin 14 and 64 .



- The pin number indicates the number on the PG board connector.
- The hardware trigger latches the encoder pulse when the signal turns from OFF to ON.
- When you use vision conveyor tracking, the software trigger is available instead of the hardware trigger.

When you use a software trigger, you need only the encoder wiring with the PG board and use the Cnv\_Trigger command in the SPEL+ program. For the command usage, refer to the sample program.

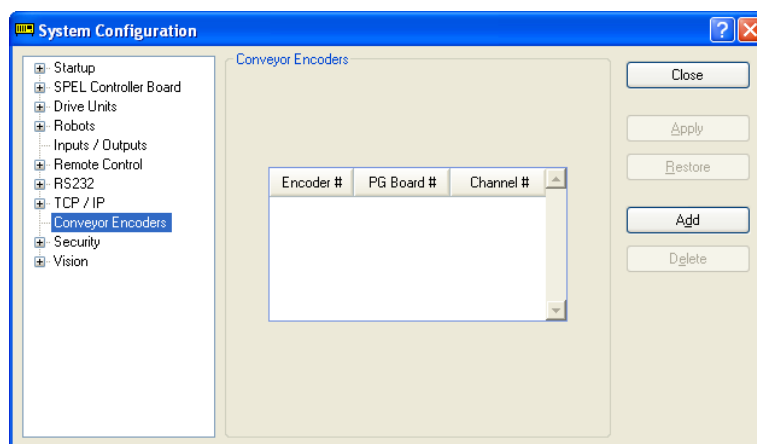
- The software trigger uses the Cnv\_Trigger command and latches the encoder on the conveyor.

## 15.5 Conveyor Encoder Configuration

Before you can create any conveyors in a project, you must first add conveyor encoders to the system. Each physical conveyor must have an encoder.

First, you must install one PG board for every four encoders in the PC Control Unit and wire the encoders to the board(s). Please refer to the Hardware Installation section of this chapter for details.

To define system encoders in EPSON RC+, select Setup | System Configuration and select the **Conveyor Encoders**.



Click the **Add** button to add an encoder. Encoders are added in the order of Axis number.

You can delete the last encoder in the list. Select it, then click the **Delete** button.

## 15.6 Verifying New Encoder Operation

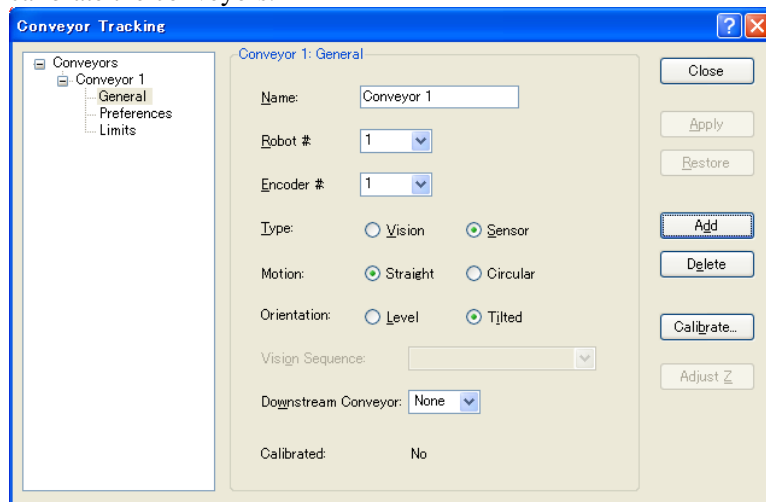
After wiring one or more new encoders and adding them to RC+ (as described in the previous section), follow these steps to verify operation.

1. Start RC+.
2. Create a new project called "TestCnv".
3. Create a conveyor by reference to the previous section.

Conveyor 1: Encoder

Type : Sensor

Make sure to perform the calibration, otherwise the conveyor tracking system cannot work properly. When you only check the encoder operation, it is not necessary to calibrate the conveyors.



4. Now you can use the Cnv\_Pulse function to read pulses from Encoder 1 from a program or from the monitor window.

For example, execute this print statement from the monitor window to read the pulses from encoder 1. Then move the conveyor and execute the command again.

```
>print cnv_pulse(1)
```

You can also use a simple program as shown below. Start the program and move the conveyor. When the conveyor starts moving, the value of Cnv\_Pulse will be changed.

```
Function main
Do
  Print Cnv_Pulse(1)
  Wait .5
Loop
Fend
```

## 15.7 Conveyor Tracking Commands

All Conveyor Tracking commands begin with the same prefix: "Cnv\_". Here is a list of all of the commands. For details, please see the *EPSON RC+ Online Help* or *SPEL<sup>+</sup> Language Reference manual*.

Command	Description / Usage
<b>Cnv_AbortTrack</b>	Aborts a motion command to a conveyor queue point.
<b>Cnv_DownStream</b>	Returns the downstream limit for the specified conveyor.
<b>Cnv_Fine Function</b>	Returns the setting of the range to judge if the tracking motion is completed or not for the specified conveyor.
<b>Cnv_Fine</b>	Sets / returns the value of Cnv_Fine for one conveyor.
<b>Cnv_LPulse Function</b>	Returns the pulse latched by a conveyor trigger.
<b>Cnv_Name\$ Function</b>	Returns the name of the specified conveyor.
<b>Cnv_Number Function</b>	Returns the number of a conveyor specified by name.
<b>Cnv_OffsetAngle</b>	Sets the angle offset. Usage: This command is available only for the circular conveyor.
<b>Cnv_OffsetAngle Function</b>	Returns the offset angle.
<b>Cnv_Point Function</b>	Returns a robot point in the specified conveyor's coordinate system derived from sensor coordinates. Usage: Use this function when registering a point in the queue.
<b>Cnv_PosErr Function</b>	Returns deviation in current tracking position compared to tracking target.
<b>Cnv_Pulse Function</b>	Returns the current position of a conveyor in pulses.
<b>Cnv_QueueAdd</b>	Adds a robot point to a conveyor queue. Usage: Use this command to register a point in the queue.
<b>Cnv_QueueGet Function</b>	Returns a point from the specified conveyor's queue. Usage: Use this command for robot tracking motion.
<b>Cnv_QueueLen Function</b>	Returns the number of items in the specified conveyor's queue. Usage: Use this command to keep the robot waiting until the part (queue) enters the tracking area.
<b>Cnv_QueueList</b>	Displays a list of items in the specified conveyor's queue.
<b>Cnv_QueueMove</b>	Moves data from upstream conveyor queue to downstream conveyor queue. Usage: Use this command for the multi conveyor system.
<b>Cnv_QueueReject</b>	Sets / displays the minimum distance to prevent the double conveyors register.
<b>Cnv_QueueReject Function</b>	Sets / returns and displays the queue reject distance for a conveyor.
<b>Cnv_QueueRemove</b>	Removes items from a conveyor queue.
<b>Cnv_QueueUserData Function</b>	Sets / returns and displays user data associated with a queue entry.
<b>Cnv_RobotConveyor Function</b>	Returns the conveyor being tracked by a robot.
<b>Cnv_Speed Function</b>	Returns the current speed of a conveyor.
<b>Cnv_Trigger</b>	Latches current conveyor position for the next Cnv_QueueAdd statement. Usage: Use this command when using the software trigger.
<b>Cnv_Upstream</b>	Returns the upstream limit for the specified conveyor.

<b>Cnv_Mode</b>	Sets the tracing mode.
<b>Cnv_Mode Function</b>	Returns the tracking mode



To track a part as the conveyor moves, you must use Cnv\_QueGet in a motion command statement. For example:

```
Jump Cnv_QueGet(1) ' this tracks the part
```

You cannot assign the result from Cnv\_QueGet to a point and then track it by moving to the point.

```
P1 = Cnv_QueGet(1)
Jump P1 ' this does not track the part!
```

When you assign the result from Cnv\_QueGet to a point, the coordinate values correspond to the position of the part when the point assignment was executed.

## 15.8 Key Terms

Here explains key terms used in this section.

<b>Queue</b>	Waiting queue of the FIFO (First-In, First-Out) type for each conveyor.  With the queue, you can register the pose data of work pieces running on the conveyor and user data. When you add data, it will be registered to the end of the queue. When you delete data from the queue, the remaining data in the queue will be moved up automatically.
<b>Queue depth</b>	The number of data entries registered in a queue. Maximum number of queue data is 1000.
<b>Queue user data</b>	Optional real value that can be registered in a queue.  You can store additional information such as sorted data or part type determined by the image processing.
<b>Downstream Conveyor</b>	Use this when using multiple conveyors and you run them continuously. By making an association (upstream/downstream) between conveyors, you can move a queue using the Cnv_QueMove command. "Multiple conveyors" is not necessarily more than one conveyor. You can use one long physical conveyor and set upstream side and downstream side as different logical conveyors. This enables the robots cooperative work, for instance, robot at the downstream side can pick up the work pieces that the robot at upstream fails to pick in time.
<b>Upstream Limit</b>	Dividing line in the upstream side of the Pickup Area.
<b>Downstream Limit</b>	Dividing line in the downstream side of the Pickup Area.
<b>Pickup Area</b>	The area between the upstream limit and downstream limit.  The robot picks parts which flow in the Pickup Area. The robot starting pickup near the downstream limit continues its operation over the downstream limit. Make sure that the Pickup Area covers the whole robot motion range.  For details, refer to <i>15.15 Pickup Area</i> .

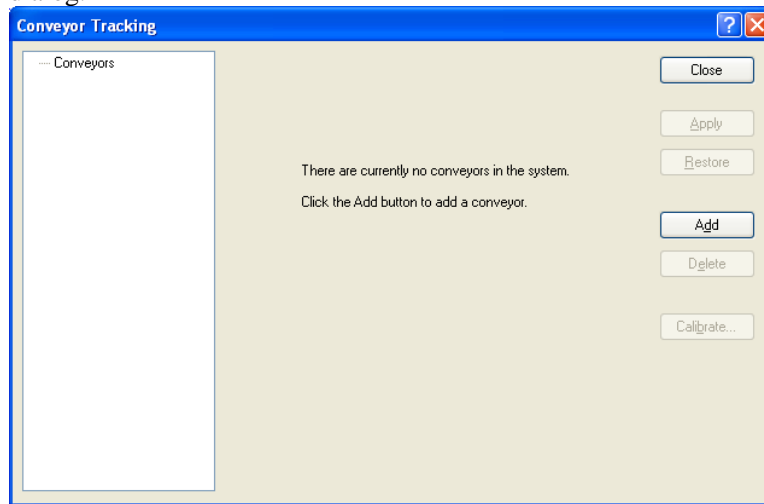
## 15.9 Creating Conveyors in a Project

Conveyors are configured for each EPSON RC+ project. Up to 16 conveyors can be created per project. A conveyor is a logical entity that combines a robot with one or more conveyors.

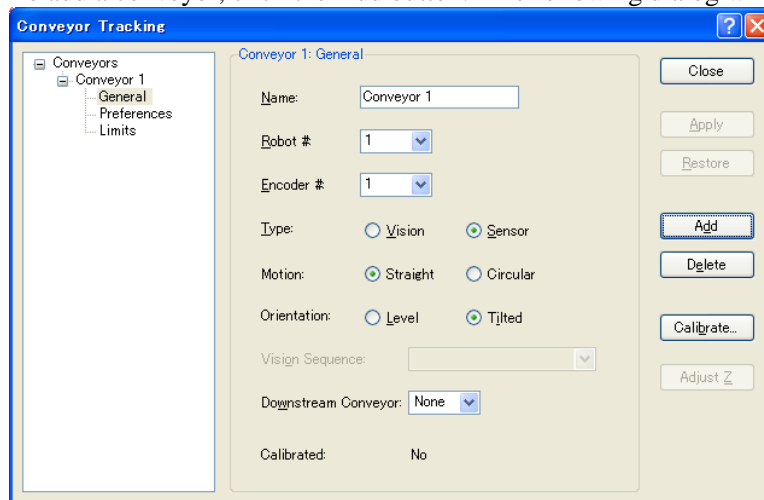
There are two types of conveyors: vision and sensor. If you will be using a vision camera to find the parts on the conveyor, you must first create a vision sequence to find the parts. This vision sequence is required when you define the conveyor.

### To add a conveyor to a project

1. Select Tools | Conveyor Tracking to open the Conveyor Tracking configuration dialog.



2. To add a conveyor, click the Add button. The following dialog will appear.



3. Enter a name for the conveyor, then specify the Robot #, Encoder #, Type, Motion, and Orientation.



- A default conveyor name is created automatically when a new conveyor is added. You can change the name as desired.
- When you use a straight conveyor, select “Straight” for [Motion].
- When you use a circular conveyor, select “Circular” for [Motion].

## 15.10 Configuring Conveyors

After a conveyor has been created, you can change its parameters.

1. Select Tools | Conveyor Tracking.
2. Click on the conveyor you want to change.
3. There are three setup pages shown in the tree under each conveyor: **General**, **Preferences**, and **Limits**.

To change the parameters for the upstream and downstream limits, click on [Limits].  
For details on the **Limits** settings, refer to *15.15 Pickup Area - Changing the Upstream / Downstream limits positions*.

To change the settings of Reject Distance and queue position data sort, click on **Preferences**.

To change other parameters, click on **General**.

4. Click on **General** or **Preferences**.  
The following dialog appears. Edit any of the configuration options.

5. Click **Apply** to save changes.



If you changed Robot #, Encoder #, Orientation, Type, or Vision Sequence, then you need to calibrate the conveyor again.

The following table explains the parameters you can edit in the **General** and **Preferences** pages.

<b>Name</b>	You can name conveyors.
<b>Robot #</b>	You can select a robot number from the robots currently configured in the controller.
<b>Encoder #</b>	You can select an encoder number from the encoders currently configured in the controller.
<b>Type</b>	Vision: Detects work pieces using vision search. Sensor: Detects work pieces using a sensor.
<b>Motion</b>	You can select the conveyor motion; Straight conveyor or Circular conveyor.
<b>Orientation</b>	When you selected Straight conveyor, you can specify if the conveyor is level or tilted.  <Tilted> is selected by default and normally you don't have to change it.  Tilted: Conveyor slope is detected during the calibration.  Level: Conveyor slope is not detected during the calibration. You need to observe the following: The conveyor must be level with the robot X and Y planes.
<b>Vision Sequence</b>	Select a vision sequence for the calibration.  This is only necessary when using the vision type.
<b>Downstream Conveyor</b>	When two or more conveyors have been set, you can select a conveyor number for the downstream conveyor.
<b>Calibrate...</b>	Click this button to execute the calibration.  The calibration procedure is different for each type and conveyor orientation.
<b>Adjust Z</b>	After the calibration is completed, you can calibrate the Z coordinate value of the conveyor again.
<b>Reject Distance</b>	You can set a minimum distance to prevent the registration of duplicate conveyors.  <ul style="list-style-type: none"> <li>• The distance also can be set from the SPEL program using the Cnv_QueueReject command.</li> <li>• If the distance is different from the one set by Cnv_QueueReject command, the Cnv_QueueReject command setting has precedence</li> </ul>
<b>Queue position data sorted on X axis</b>	You can select whether to sort the queue or not.



## 15.11 Vision Conveyors

A vision conveyor uses a camera to locate parts that will be retrieved by one or more robots. In this section, instructions are provided for vision conveyor calibration and programming.

The straight conveyor and circular conveyor have different calibration and programming methods.

### Vision conveyor camera and lighting

It is important to choose the correct camera and lighting for the vision conveyors used in your application.

For applications with a slow moving conveyor and non-critical pick up constraints, you may be able to use a Vision Guide camera and simple lighting with no strobe.

For applications with fast moving parts, you will need to use a camera that is capable of asynchronous reset along with a strobe lamp. This method is more expensive.

If you are using multiple asynchronous reset cameras in multiple tasks, you must use SyncLock to lock the vision system during VRun and waiting until the picture is acquired.

For example:

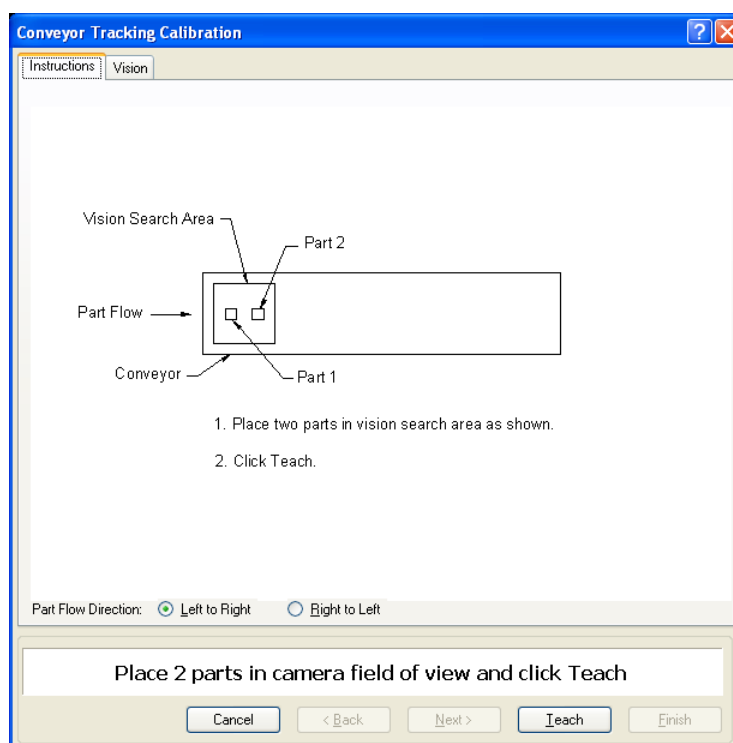
```
SyncLock 1      ' Lock vision for this task
VRun FindPart
On strobe, .2
Do
    VGet FindPart.AcquireState, state
Loop Until state = 3
SyncUnlock 1    ' Unlock vision
```

### Vision calibration sequence

Before you can calibrate a vision conveyor, you must first create a calibration sequence. This sequence is used by the system during the calibration process and must be linked to a camera calibration. The conveyor system commands use camera coordinates in millimeters. Although you can use any type of Vision Guide camera calibration, you only need to use a Standalone calibration.

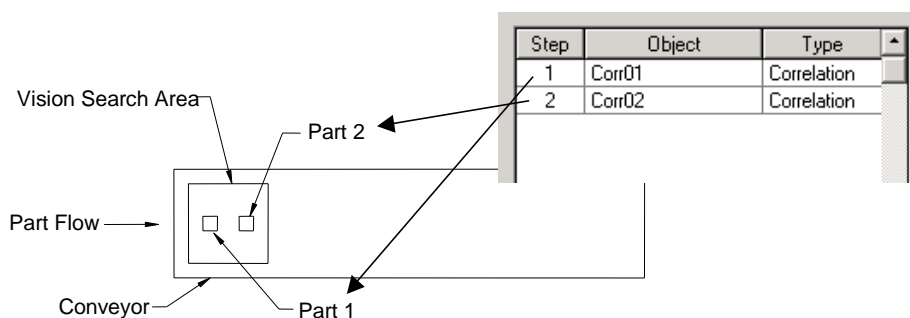
The calibration sequence needs a sequence that uses one object for each work piece.

Place two work pieces on the conveyor as shown below.



The two parts can be anywhere in the field of view. However, the first object of a sequence must be taught with the robot as Part 1. The second object of a sequence must be taught with the robot as Part 2.

Also, the two parts can be anywhere in the field of view. However, to make it as easy as possible for operators to calibrate the conveyor, the parts that will be found in the vision sequence should be located such that part 2 is after part 1 in the direction of part flow. In the figure below, object 1 in the vision sequence is Corr01, which locates Part 1. Object 2 is Corr02, which locates Part 2.



### Vision conveyor calibration (Straight conveyor)

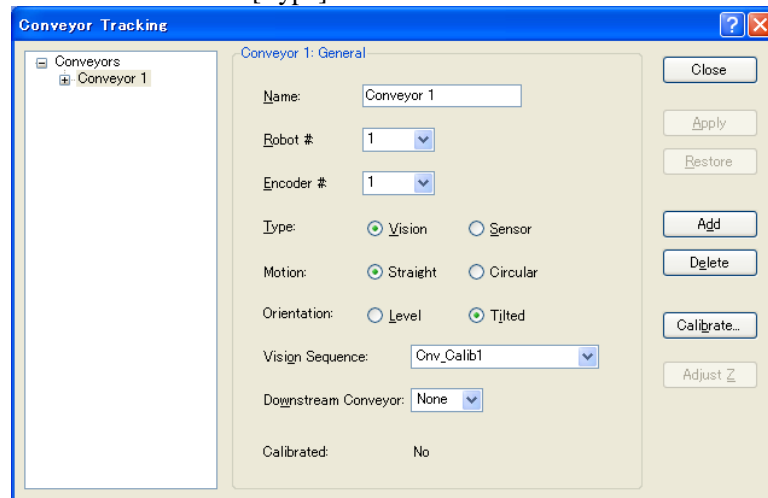
Follow these steps to calibrate a straight vision conveyor:



NOTE

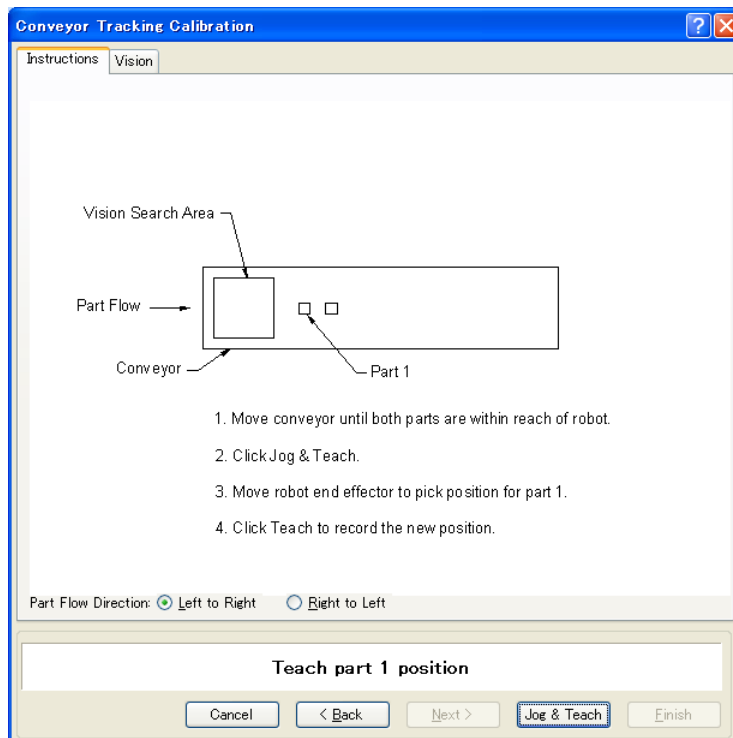
- When teaching part positions with the robot during calibration, it is important to position X, Y, and Z of each point accurately. The conveyor is calibrated in X, Y, Z, U, V, and W.
- To perform the fine calibration, in the step 15 and 17, set as wide distance as possible between the upstream limit and downstream limit. After the calibration, adjust the Pickup Area by resetting the upstream / downstream limits.
- For the level orientation, it determines the conveyor height with the position of robot end effector taught in the step 12. It cannot be used for the tilted conveyor for it does not detect the conveyor slope. The steps 19 to 20 are not displayed.
- For the tilted orientation, it calibrates the conveyor slope with the position of robot end effector taught in the steps 12, 14, 16, 18, and 20.

1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to calibrate.
3. Select **Vision** for the [Type].

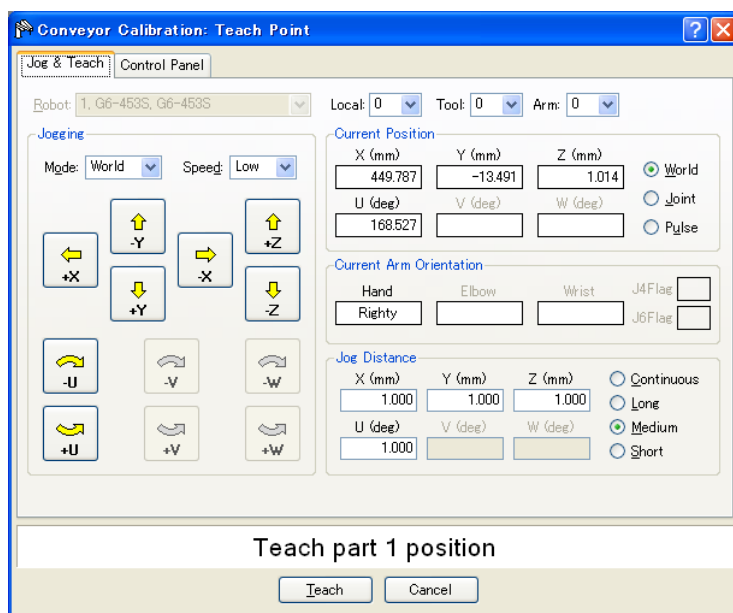


4. Set the [Vision Sequence].
5. Click the **Apply** button.
6. Click the **Calibrate** button. The Conveyor Tracking Calibration wizard will appear. Follow the instructions for each step. Before you can proceed to the next step, you must click the **Teach** button. You can go back to previous steps using the **Back** button.
7. Select the Part Flow Direction to best match the conveyor you are calibrating. The instruction pictures will change according to the setting. Part Flow Direction is only used to aid in the instructions. It has no impact on the calibration.
8. Place two parts on the conveyor as shown in the figure in the wizard.
9. Select the Vision tab to see live video. The camera orientation may not be the same as the picture.

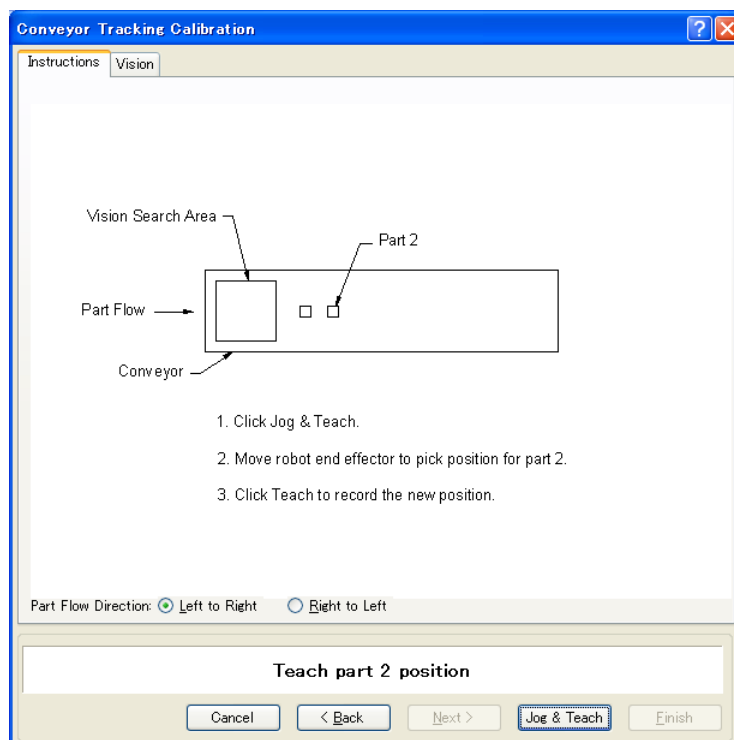
10. Arrange the parts to be inside the range correctly and click **Teach** button. Use the camera video to ensure that the parts are within the correct search area for each. Click the **Teach** button. If the Vision tab is selected when you click **Teach**, you will see the vision sequence graphical results displayed. In this case, the wizard will not advance to the next step and you must click the **Next** button to view the next step. This allows you to click **Teach** more than one time in case you want to adjust the parts.
11. Move the conveyor by hand until both parts are within reach of the robot. Do not move the parts, only the conveyor. Click the **Jog & Teach** button.



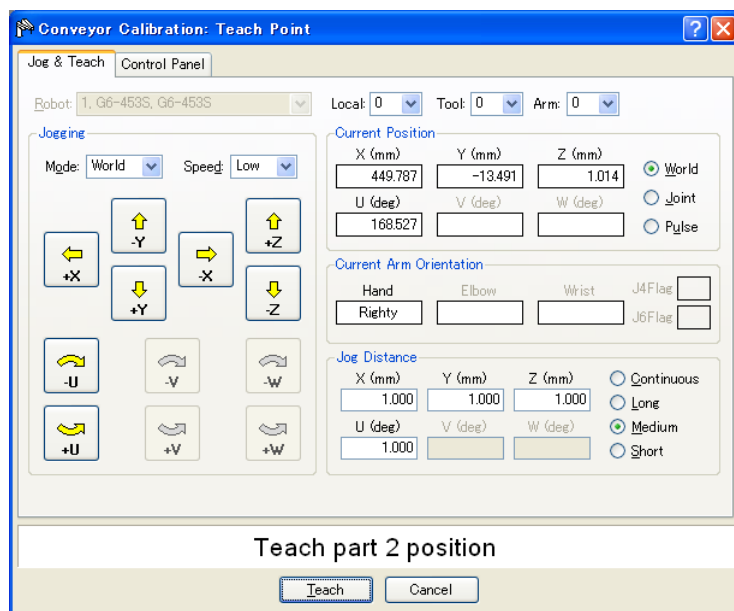
12. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position for Part 1. Click the **Teach** button.



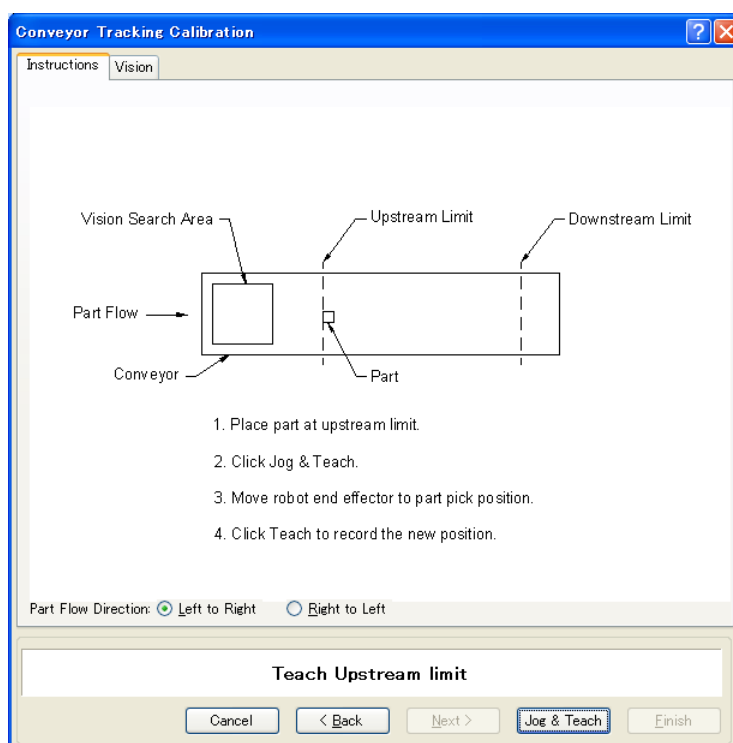
13. Click the **Jog & Teach** button.



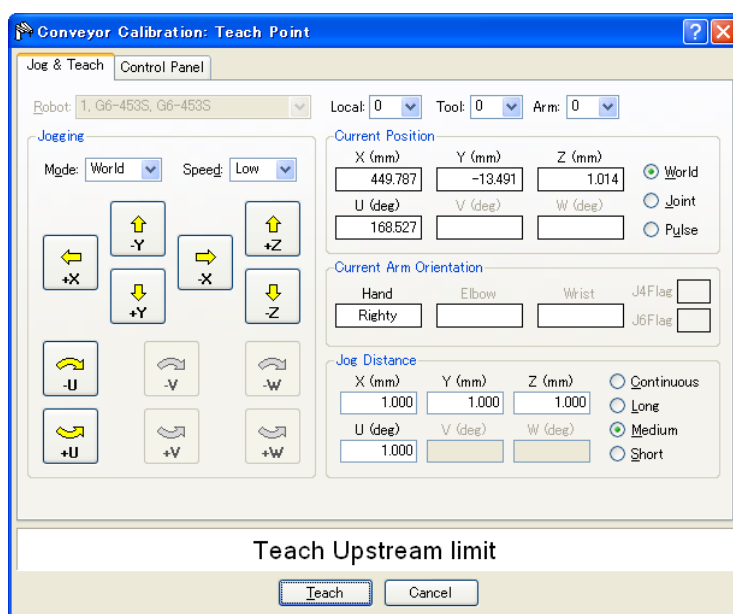
14. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position for Part 2. Click the **Teach** button.



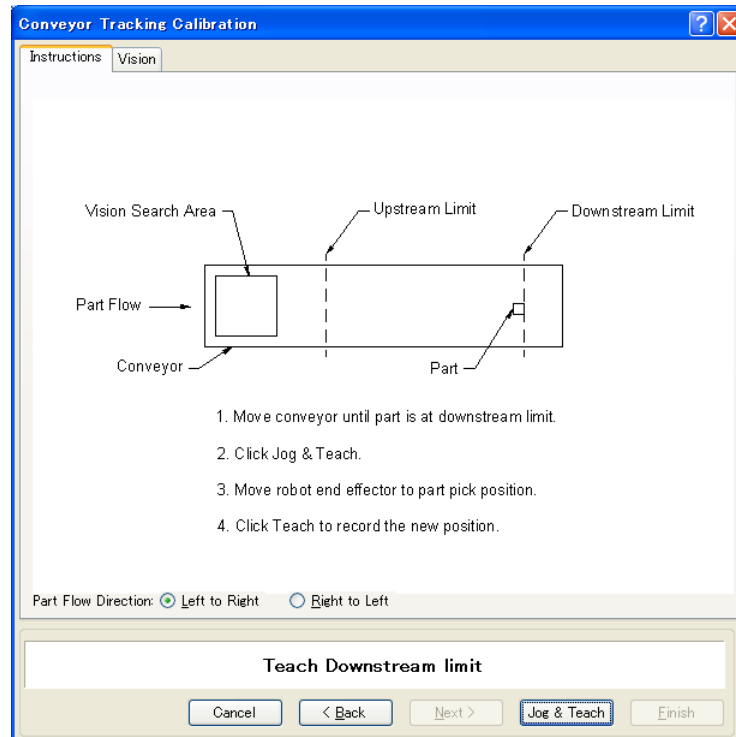
15. Now move or place the part at the upstream limit. Click the **Jog & Teach** button.



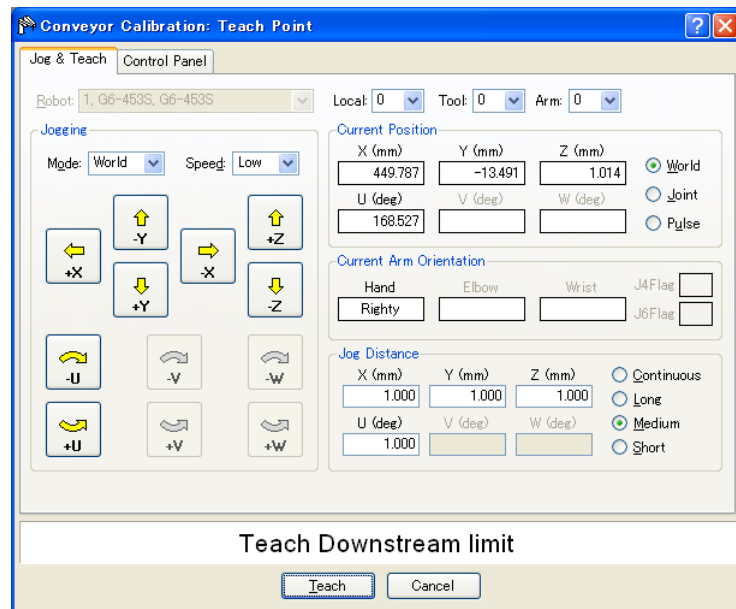
16. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



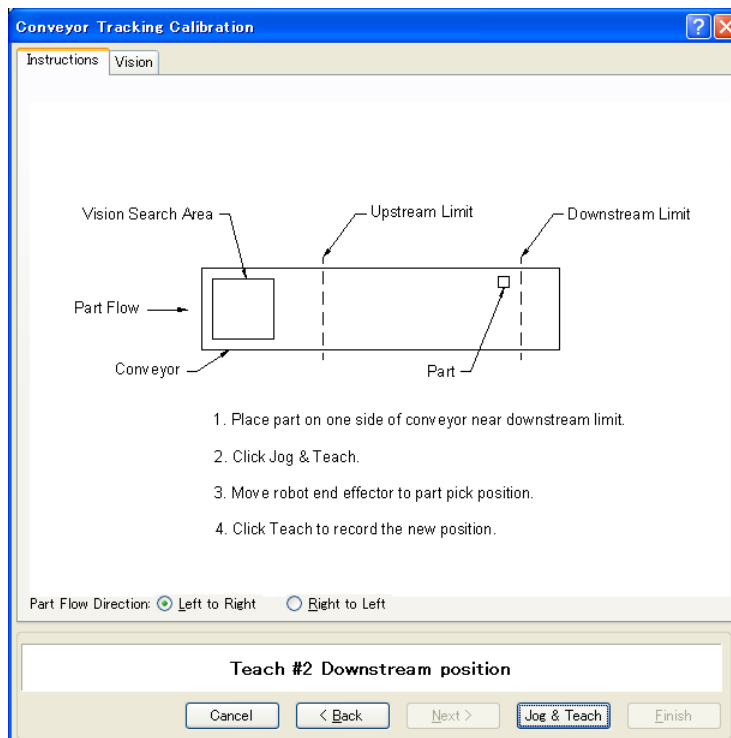
17. Move the conveyor so the part is at the downstream limit. Do not move the part, only the conveyor. Click the **Jog & Teach** button.



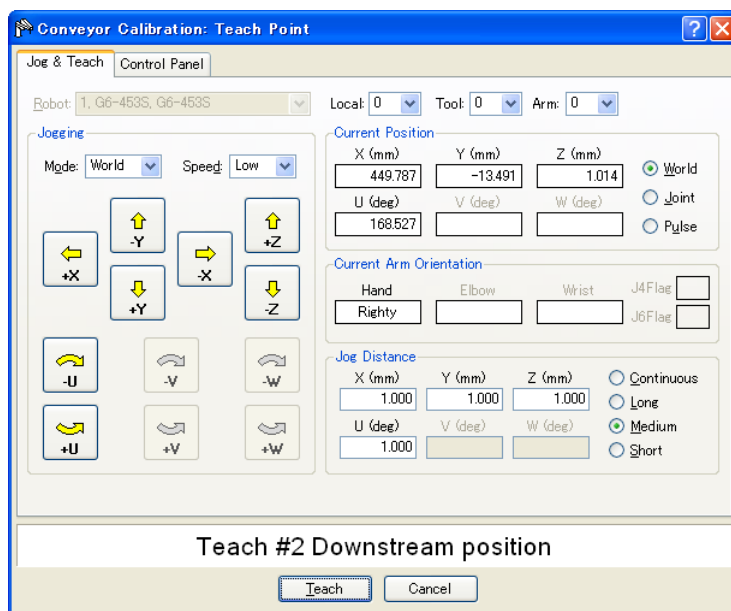
18. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector towards the part. Click the **Teach** button.



19. Place a part on one side of the conveyor near the downstream limit. This point is used to determine the tilt of the conveyor from side to side. Click the **Jog & Teach** button.

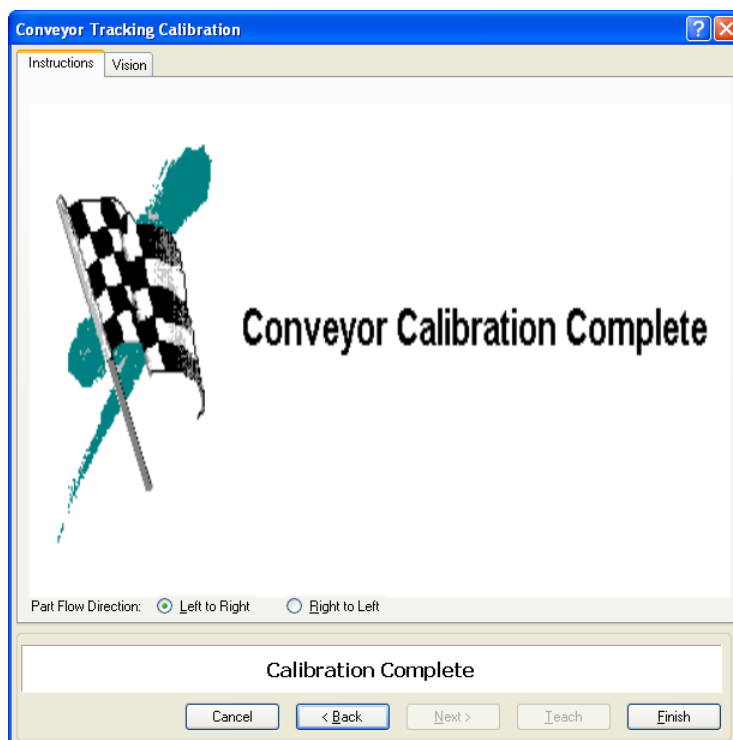


20. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the part position. Click the **Teach** button.





21. The calibration complete picture will be displayed. Click the Finish button.



### Vision conveyor calibration (Circular conveyor)

Follow these steps to calibrate a circular vision conveyor:

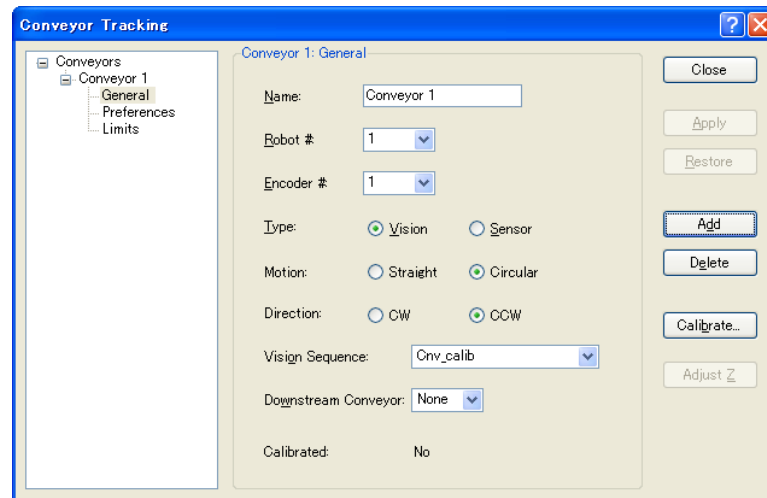


- When teaching part positions with the robot during calibration, it is important to position X, Y, and Z of each point accurately. The conveyor is calibrated in X, Y, Z, U, V, and W.
- To perform the fine calibration, in steps 13, 17, and 19, teach the position when the robot is directly above the parts 1 and set as wide a distance as possible between the points to be taught.

1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to calibrate.
3. Select **Vision** for the [Type].
4. Select **Circular** for the [Motion].
5. Select the conveyor rotating direction for the [Direction].

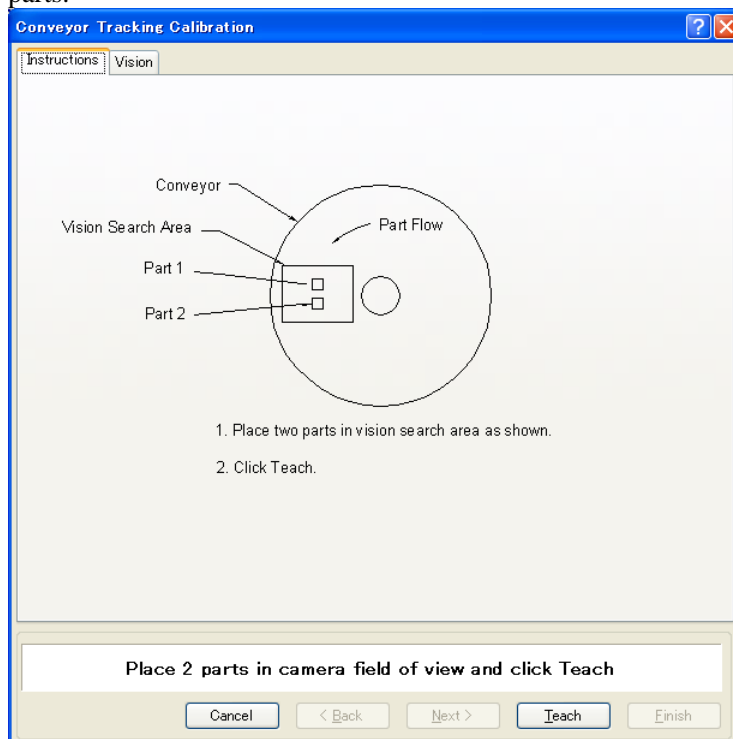


Be careful not to calibrate with a wrong direction, otherwise, the robot will not track the parts.

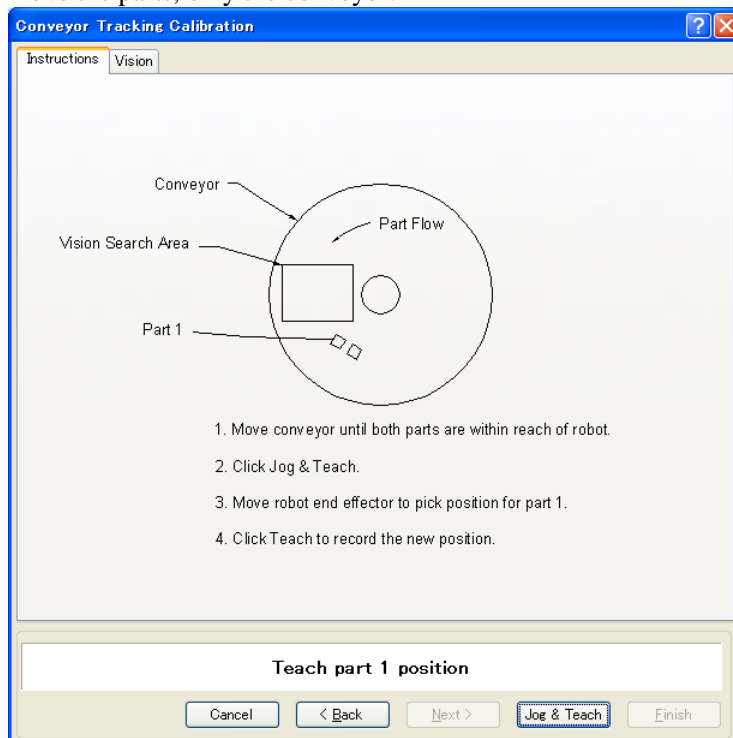


6. Select the [Vision Sequence].
7. Click the **Apply** button.
8. Click the **Calibrate** button. The Conveyor Tracking Calibration wizard will appear. Follow the instructions for each step. Before you can proceed to the next step, you must click the **Teach** button. You can go back to previous steps using the **Back** button.
9. Check if the conveyor direction shown in the wizard is the same as the conveyor you want to use.
10. Place two parts on the conveyor as shown in the figure in the wizard.
11. Select the Vision tab to see live video. The camera orientation may not be the same as the picture.

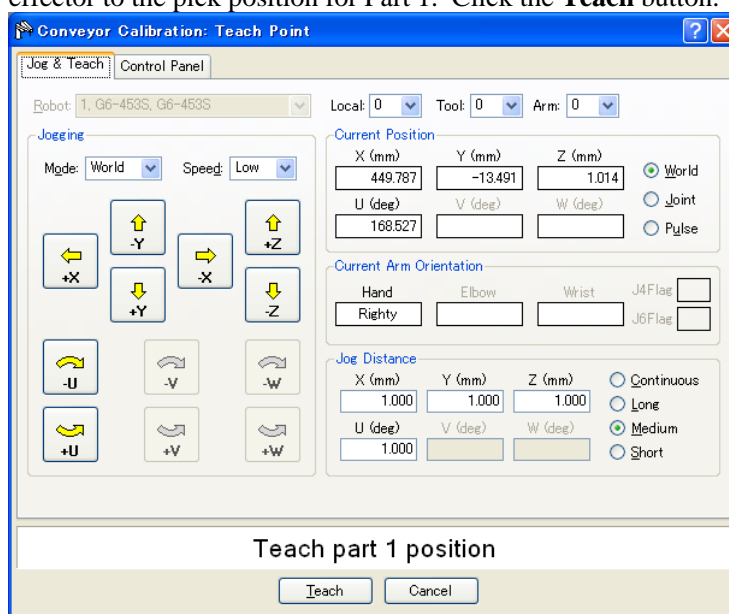
12. Arrange the parts to be inside the range correctly and click the **Teach** button. If the Vision tab is selected when you click **Teach**, you will see the vision sequence graphical results displayed. In this case, the wizard will not advance to the next step and you must click the **Next** button to view the next step. This allows you to click **Teach** more than one time in case you want to adjust the parts.



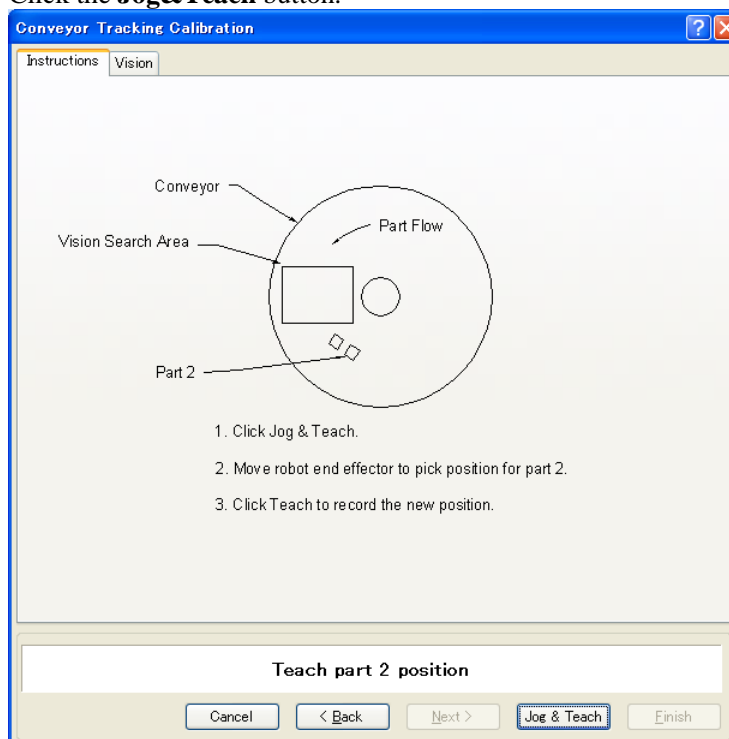
13. Move the conveyor by hand until both parts are within reach of the robot. Do not move the parts, only the conveyor.



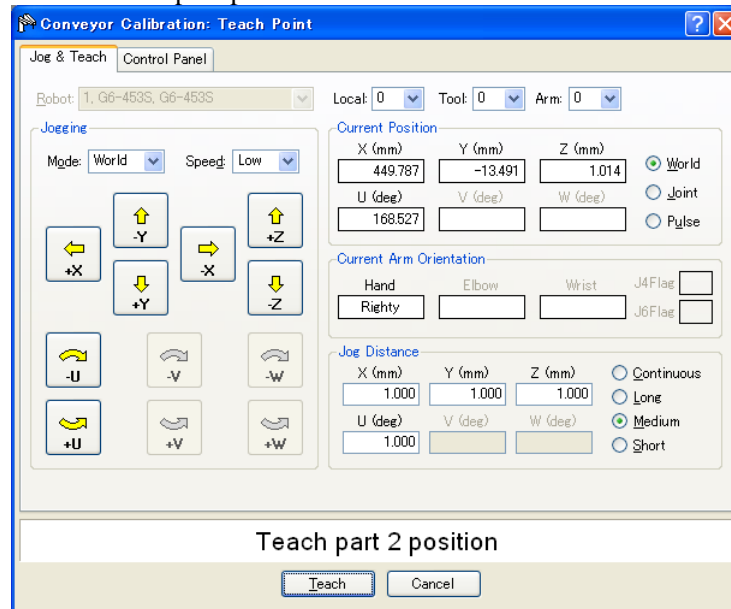
14. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position for Part 1. Click the **Teach** button.



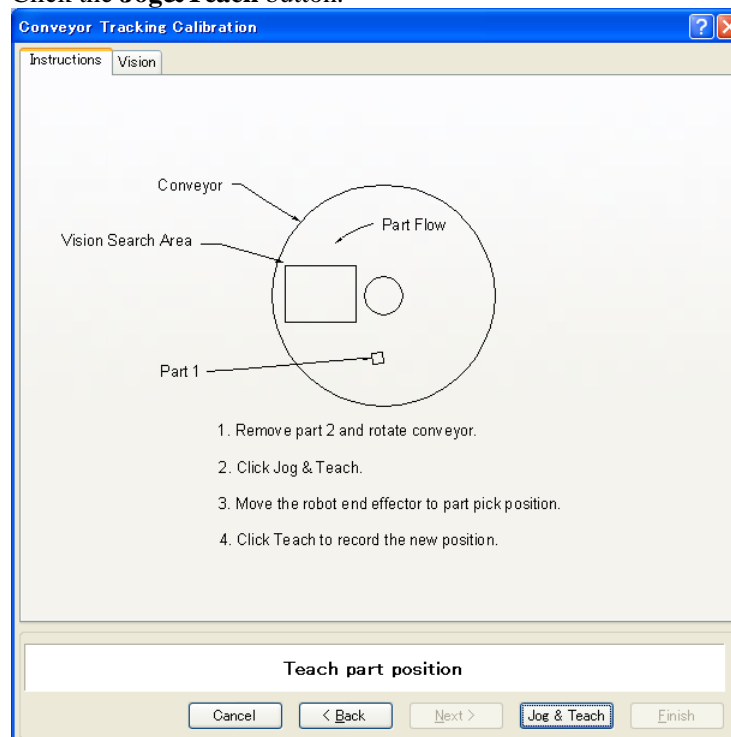
15. Click the **Jog&Teach** button.



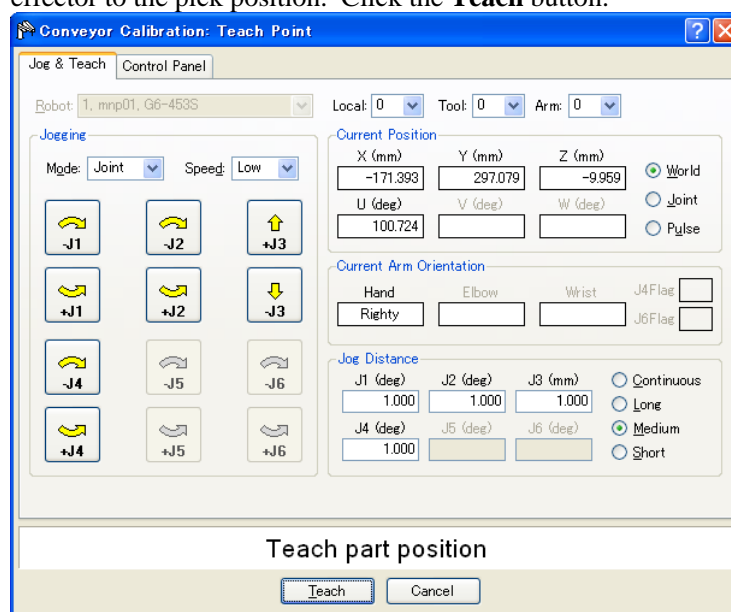
16. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position for Part 2. Click the **Teach** button.



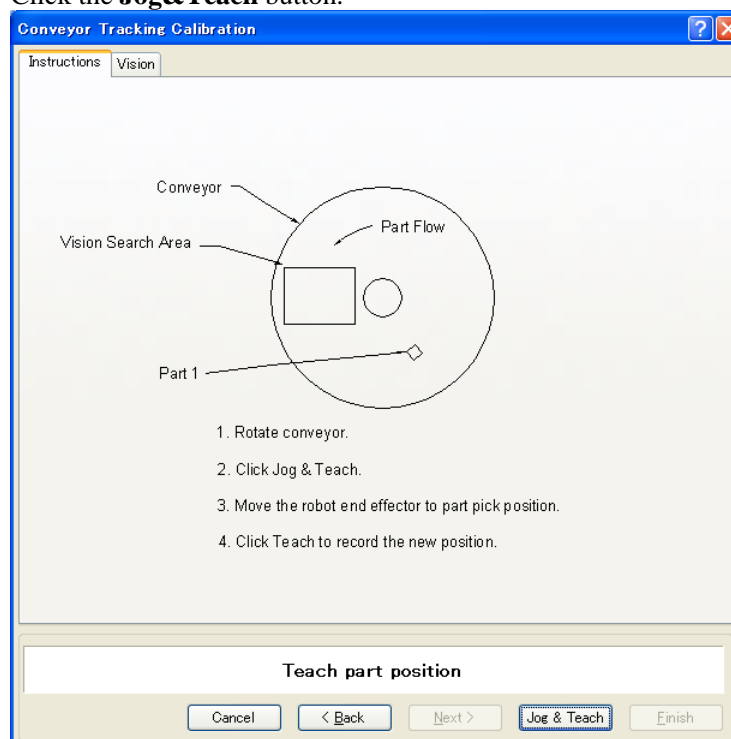
17. Remove Part 2. Move the conveyor by hand to move Part 1. Click the **Jog&Teach** button.



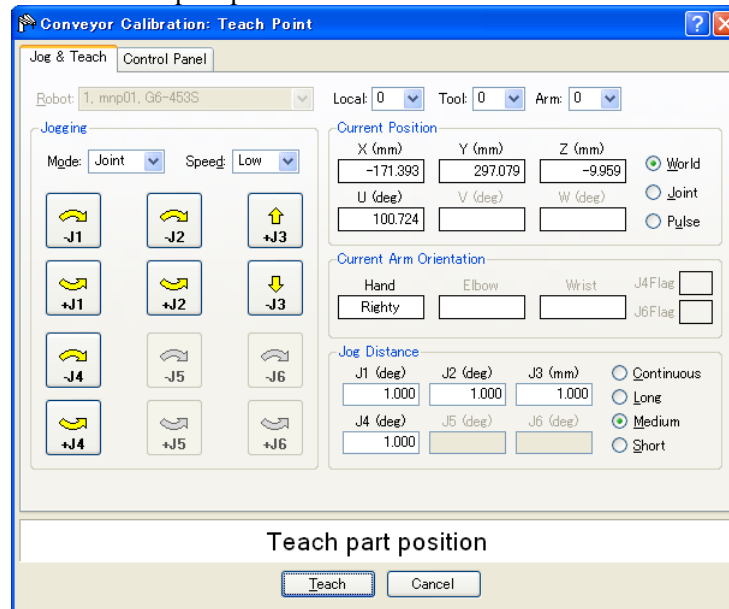
18. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



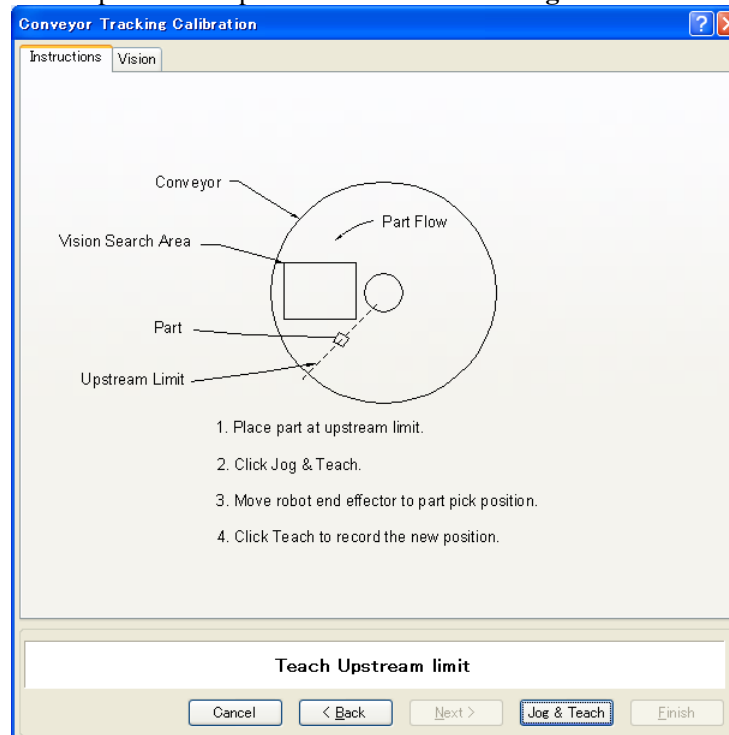
19. Move the conveyor by hand to move Part 1. Click the **Jog&Teach** button.



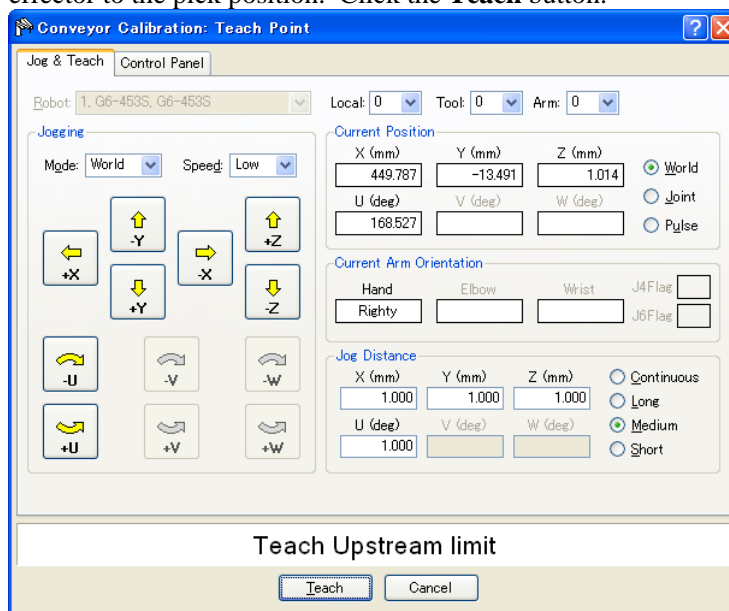
20. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



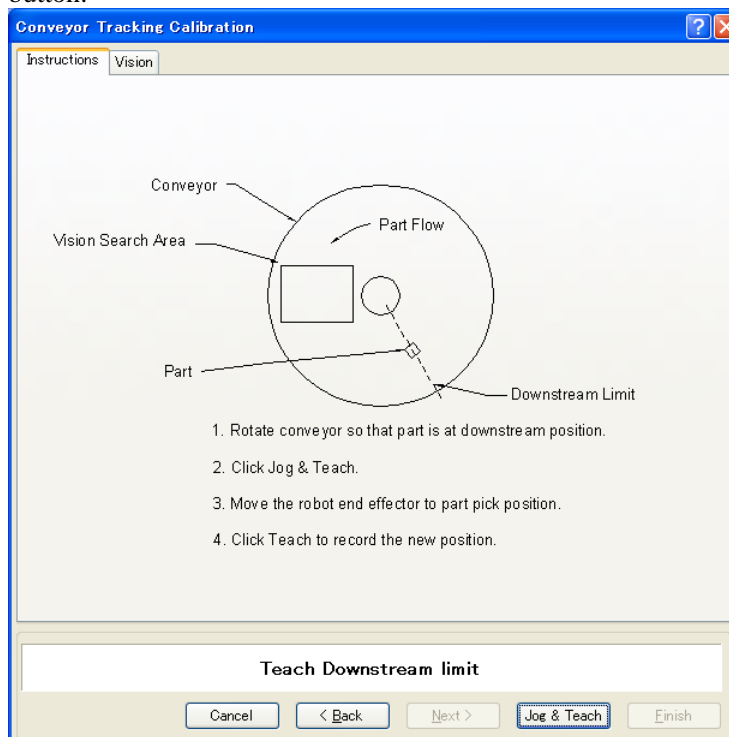
21. Place a part on the upstream limit. Click the **Jog & Teach** button.



22. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.

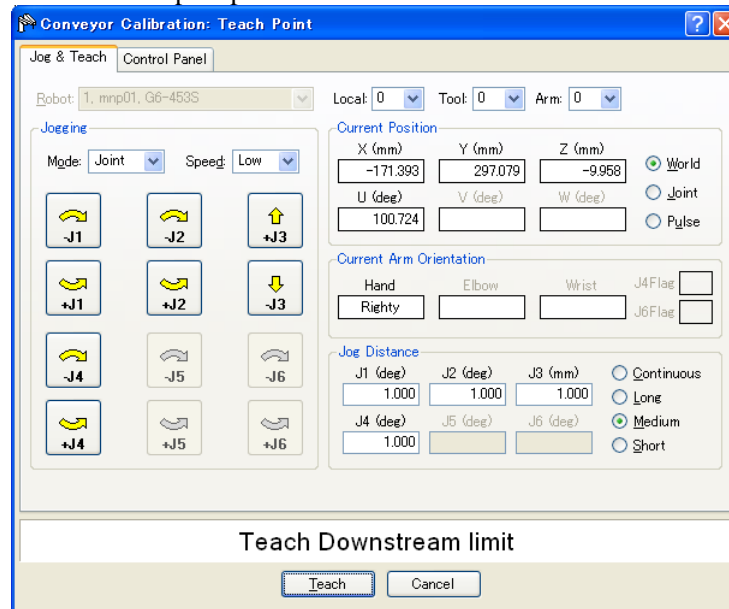


23. Move the conveyor so the part is on the downstream limit. Click the **Jog & Teach** button.

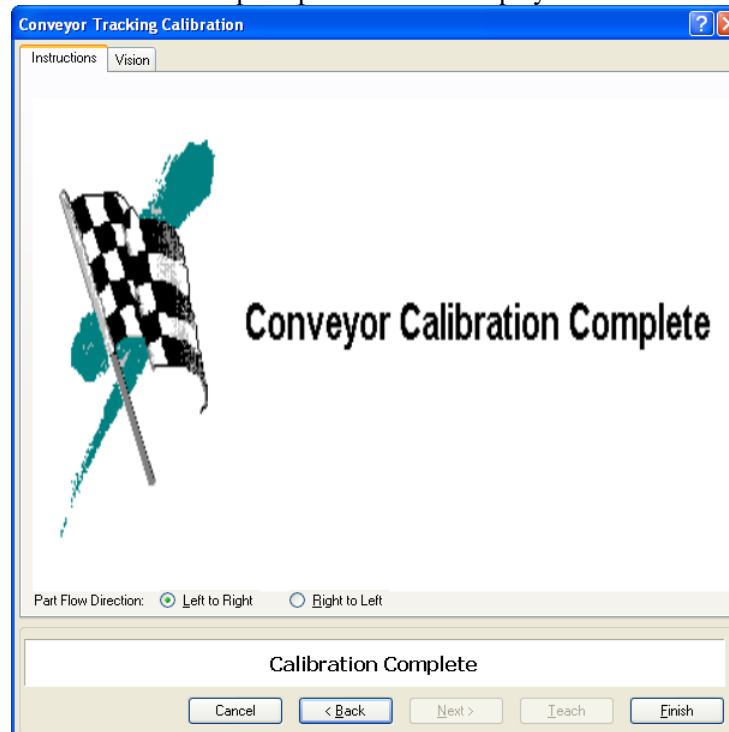




24. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



25. The calibration complete picture will be displayed. Click the Finish button.



### Vision conveyor operation check

After the calibration, we recommend that you check if the vision conveyor works properly. Use the sample program “ScanConveyorStrobed” and the Command Window and follow the procedure below.

In this section, the operation of Conveyor #1 will be checked.

1. Clear the all queue data registered to the conveyor.  
`>Cnv_QueueRemove 1,all`
2. Place parts in the vision search area.
3. Execute the program “ScanConveyorStrobed” to register a queue.
4. Halt the program “ScanConveyorStrobed” and move the conveyor until parts reach the Pickup Area.
5. Pick up parts.  
`>Jump Cnv_Queueget(1)`
6. Check if the robot end effector is over the center of the part to pick.
7. Move the conveyor and check if the robot follows the part. At this point, the end effector will be off the center of part but this is no problem.
8. Stop the tracking motion.  
`>Cnv_AbortTrack`

In case the following symptoms occur, the Vision Guide or conveyor calibration was not executed correctly. Perform the calibration again.

- Robot cannot pick a part in the center.
- Robot cannot follow parts when the conveyor is moving.

### Vision conveyor programming

Typically, two tasks are used to operate a vision conveyor. One task finds parts with the vision system and adds them to the conveyor queue.

The other task checks for parts in the Pickup Area of the conveyor queue. When a part is in the Pickup Area, the robot is commanded to pick up the part and place it to the specified position.

The following example shows two tasks. The scanning task uses the vision system to find parts and add them to the conveyor queue. There are two examples for the scanning task. “ScanConveyorNonStrobed” does not use a strobe lamp and hardware trigger. In this case, “Cnv\_Trigger” must be called before running the vision sequence. “ScanConveyorStrobed” uses a strobe lamp and hardware conveyor trigger. “PickParts” waits for parts to be present in the Pickup Area and commands the robot to pick and place each part.

If you are using an asynchronous reset camera and strobe, then the strobe trigger should also be wired to the trigger on the PG board. In this case, the vision sequence “RuntimeAcquire property” must be set to “Strobed”.

The following program is a sample with Conveyor #1.

```
Function main
    Xqt ScanConveyorStrobed      ' Task that registers queues
    Xqt PickParts                ' Task that tracks parts (queue)
Fend

Function ScanConveyorNonStrobed
    Integer i, numFound
    Real x, y, u
    Boolean found
    Cnv_OffsetAngle 1,xx        ' Command used for only circular conveyors
                                ' Adjust the tracking error with an offset value in xx
    Do
        Cnv_Trigger 1          ' Latch the encoder with software trigger
        ' Search for parts on the conveyor
        VRun FindParts
        VGet FindParts.Parts.NumberFound, numFound
        ' Register the part as a queue
        For i = 1 to numFound
            VGet FindParts.Parts.CameraXYU(i), found, x, y, u
            Cnv_QueAdd 1, Cnv_Point(1, x, y)
        Next i
        Wait .1
    Loop
Fend

Function ScanConveyorStrobed
    Integer i, numFound, state
    Real x, y, u
    Boolean found
    Cnv_OffsetAngle 1,xx        ' Command used for only circular conveyors
                                ' Adjust the tracking error with an offset value in xx
    ' Turn OFF the camera shutter and I/O (conveyor trigger)
```

```
Off trigger; off Cv_trigger
Do
    ' Search for parts on the conveyor
    VRun FindParts
    ' Turn ON the camera shutter and I/O (conveyor trigger)
    On Trigger; On Cv_Trigger
    Do
        VGet FindParts.AcquireState, state
    Loop Until state = 3
    VGet FindParts.Parts.NumberFound, numFound
    ' Register the part that has been shot as a queue
    For i = 1 to numFound
        VGet FindParts.Parts.CameraXYU(i), found, x, y, u
        Cnv_QueueAdd 1, Cnv_Point(1, x, y)
    Next i
    ' Turn OFF the camera shutter and I/O (conveyor trigger)
    Off Trigger; Off Cv_Trigger
    Wait .1
Loop
Fend
```

```
Function PickParts
    OnErr GoTo ErrHandler
    Integer ErrNum
    WaitParts:
    Do
        ' Wait until a part (queue) enters the Pickup Area
        Wait Cnv_QueueLen(1, CNV_QUELEN_PICKUPAREA) > 0
        ' Start tracking the parts
        Jump Cnv_QueueGet(1)
        On gripper
        Wait .1
        ' Move the picked part to a specified place
        Jump place
        Off gripper
        Wait .1
        ' Clear the picked part (queue)
        Cnv_QueueRemove 1, 0
    Loop
    ' Clear the parts (queue) in the downstream side from the Pickup Area
    ' When error 4406 occurs, restore automatically
    ErrHandler:
        ErrNum = Err
        If ErrNum = 4406 Then
            Cnv_QueueRemove 1, 0
            EResume WaitParts
```

```

        ' When an error except error 4406 occurs, display the error
    Else
        Print "Error!"
        Print "No.", Err, ":", ErrMsg$(Err)
        Print "Line :", Erl(0)
    EndIf
Fend

```



When you use the strobe light and software trigger, use the “ScanConveyorStrobed” function shown below.

```

Function ScanConveyorStrobed
    Integer i, numFound, state
    Real x, y, u
    Boolean found
    Cnv_OffsetAngle 1,xx      ' Command used only for circular conveyors
                                ' Adjust the tracking error with an offset value in xx
    ' Turn OFF the camera shutter
    Off trigger
    Cnv_Trigger 1             ' Latch the encoder with software trigger
    Do
        ' Search for parts on the conveyor
        VRun FindParts
        ' Turn ON the camera shutter
        On Trigger; On Cv_Trigger
    Do
        VGet FindParts.AcquireState, state
    Loop Until state = 3
    VGet FindParts.Parts.NumberFound, numFound
    ' Register the part that has been shot as a queue
    For i = 1 to numFound
        VGet FindParts.Parts.CameraXYU(i), found, x, y, u
        Cnv_QueueAdd 1, Cnv_Point(1, x, y)
    Next I
    ' Turn OFF the camera shutter
    Off Trigger
    Wait .1
    Loop
Fend

```

## 15.12 Sensor Conveyors

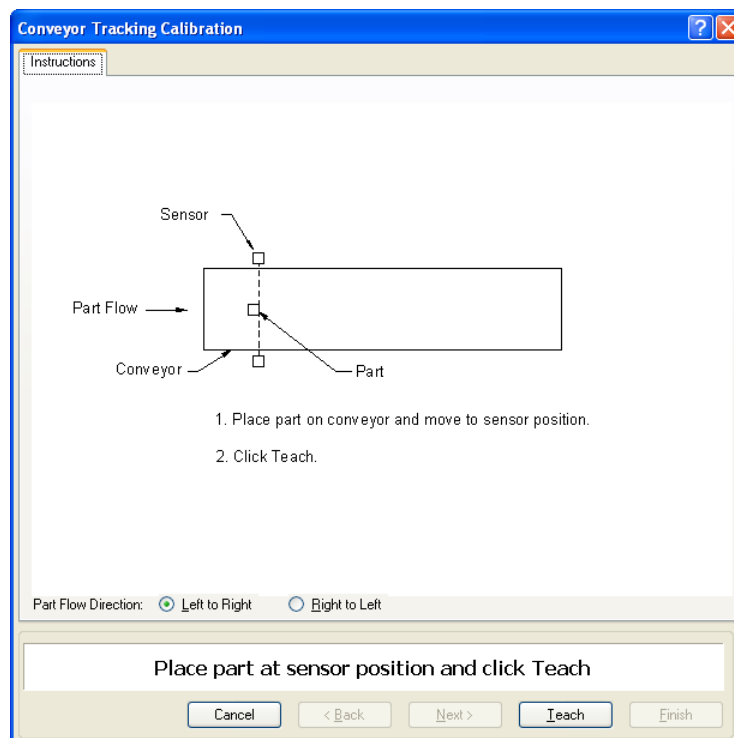
### Sensor conveyor calibration (Straight conveyor)

Follow these steps to calibrate a straight sensor conveyor:

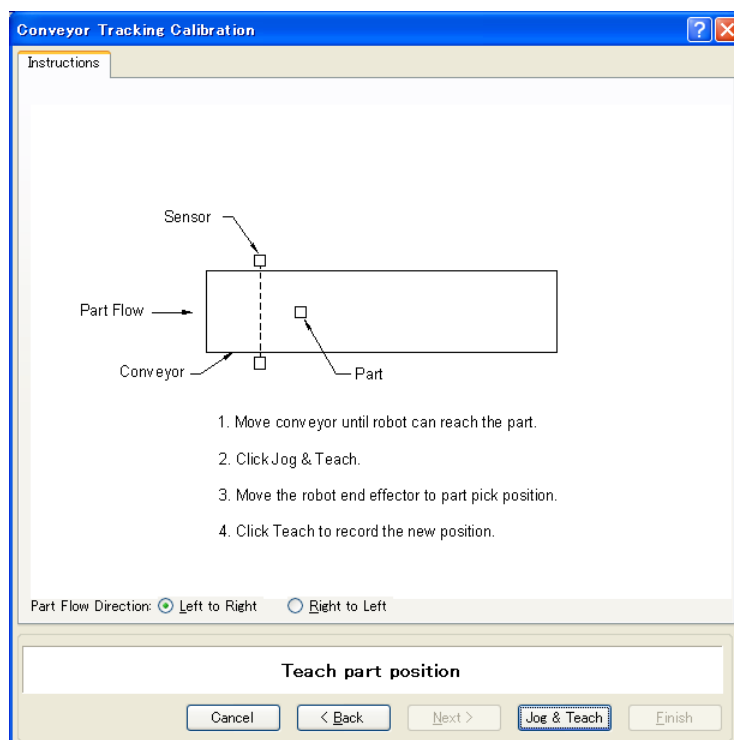


- When teaching part positions with the robot during calibration, it is important to position X, Y, and Z of each point accurately. The conveyor is calibrated in X, Y, Z, U, V, and W.
- To perform the fine calibration, in steps 9 and 11, set as wide a distance as possible between the upstream limit and the downstream limit. After calibration, adjust the Pickup Area by resetting the upstream / downstream limits.
- For the level orientation, the conveyor height is determined by the position of the robot end effector taught in step 8. It cannot be used for the tilted conveyor for it does not detect the conveyor slope. Steps 19 to 20 are not displayed.
- For the tilted orientation, it calibrates the conveyor slope with the position of robot end effector taught in the steps 8, 10, 12, and 14.

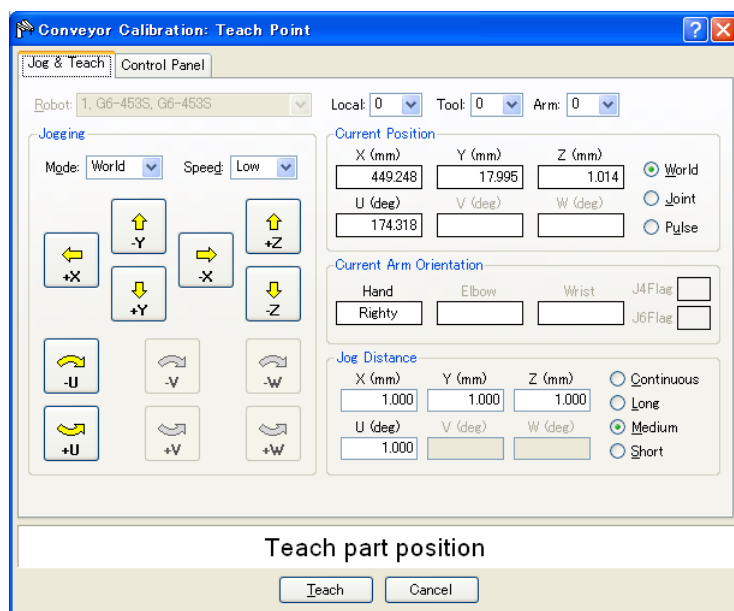
1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to calibrate.
3. Click the **Calibrate** button. The Conveyor Tracking Calibration wizard will appear.
4. Follow the instructions for each step. Before you can proceed to the next step, you must click the **Teach** button. You can go back to previous steps using the **Back** button.
5. Select the Part Flow Direction to best match the conveyor you are calibrating. The instruction pictures will change according to the setting. Part Flow Direction is only used to aid in the instructions. It has no impact on the calibration.
6. For the first wizard step, place a part on the conveyor and move the conveyor toward the sensor until the sensor just turns on. Click the **Teach** button.



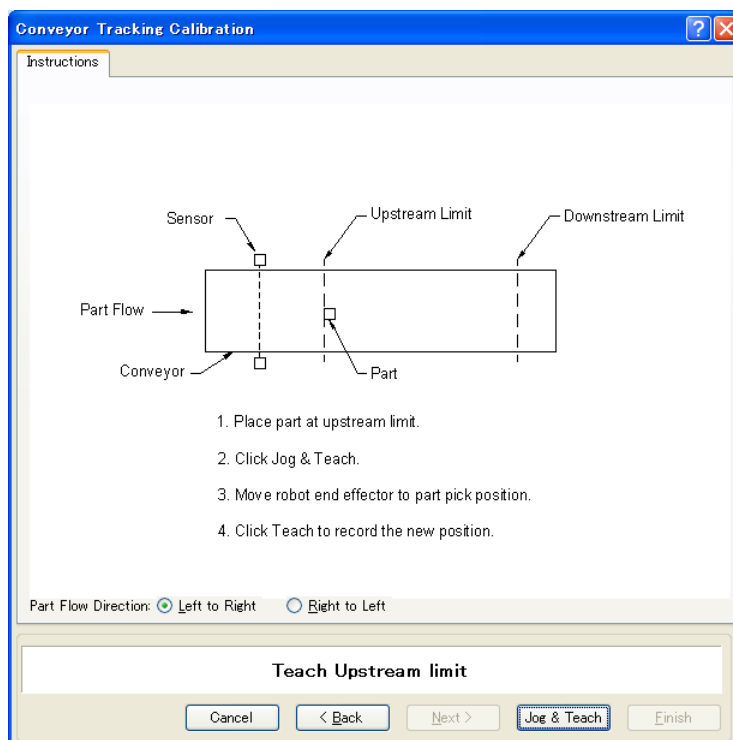
7. Move the conveyor by hand until the part is within reach of the robot. Do not move the part itself, only the conveyor. Click the **Jog & Teach** button.



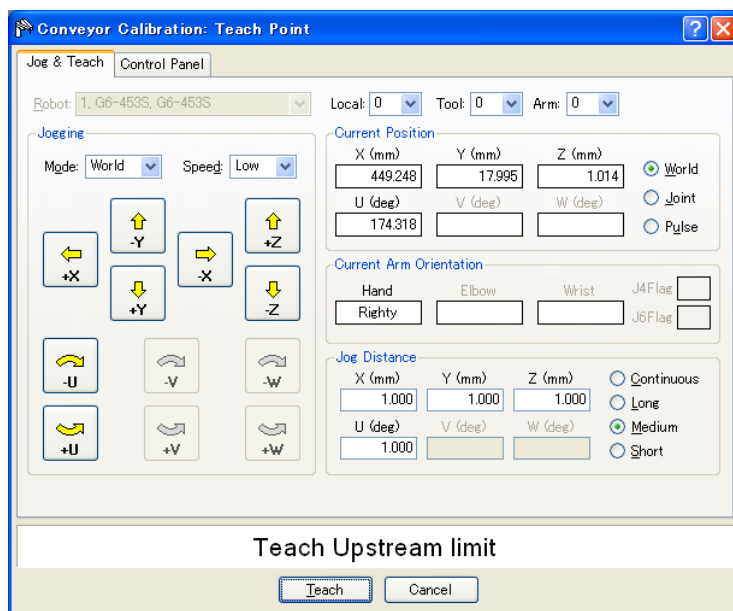
8. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



9. Now move or place the part at the upstream limit. Click the **Jog & Teach** button.

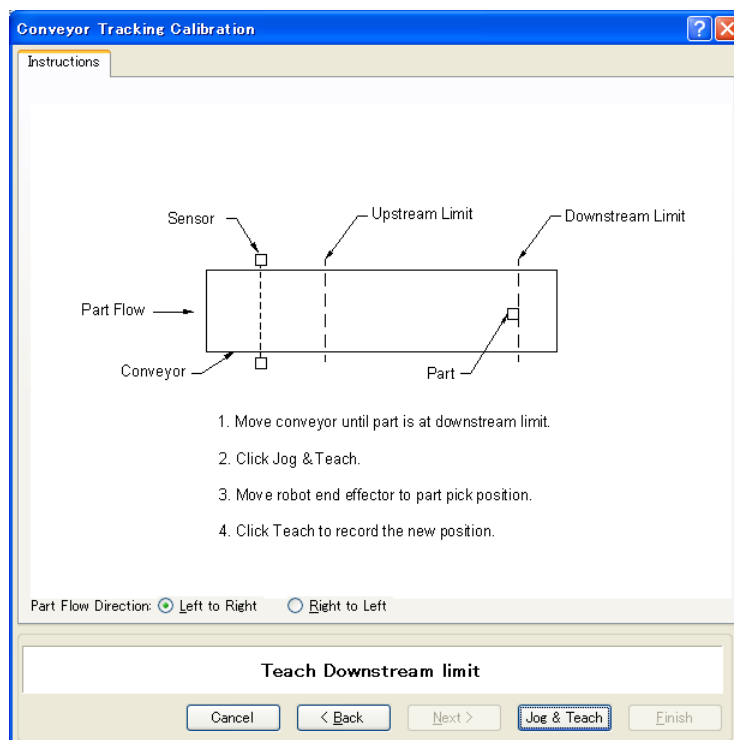


10. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.

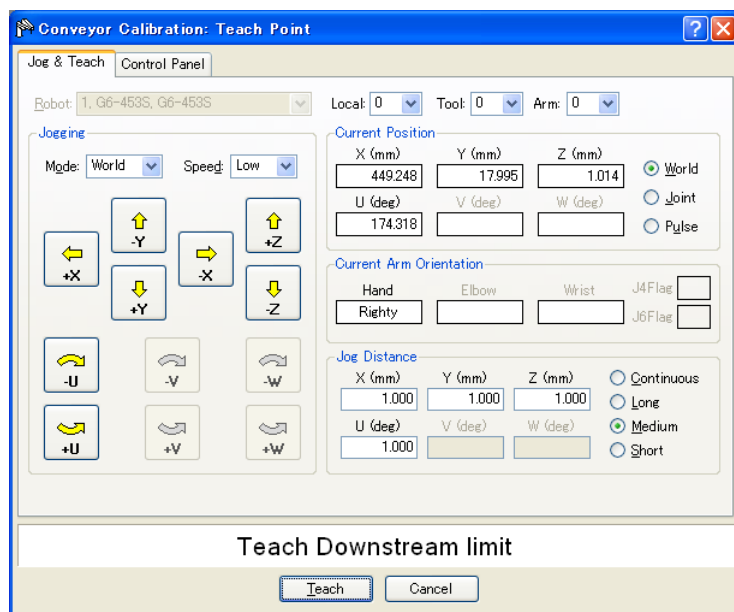




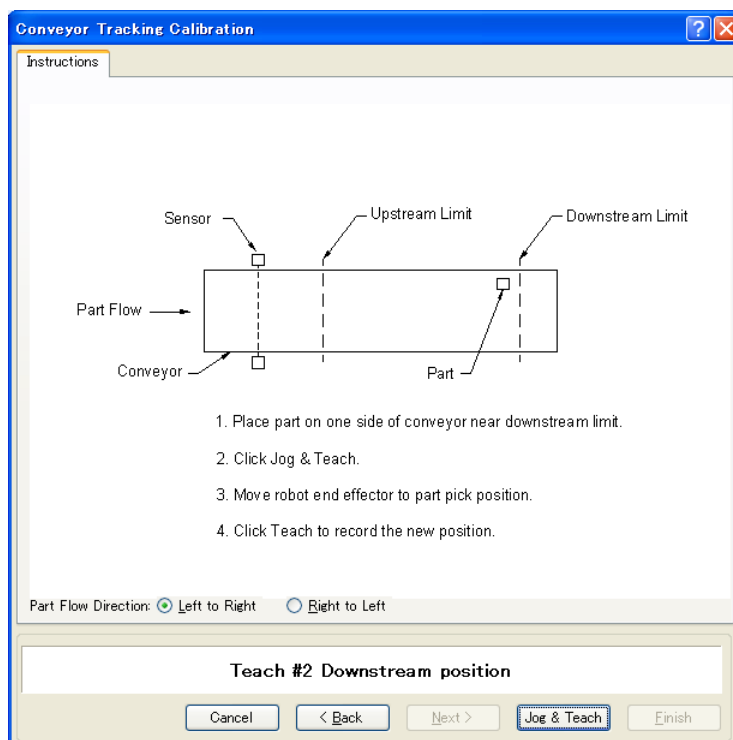
11. Move the conveyor so the part is at the downstream limit. Do not move the part, only the conveyor. Click the **Jog & Teach** button.



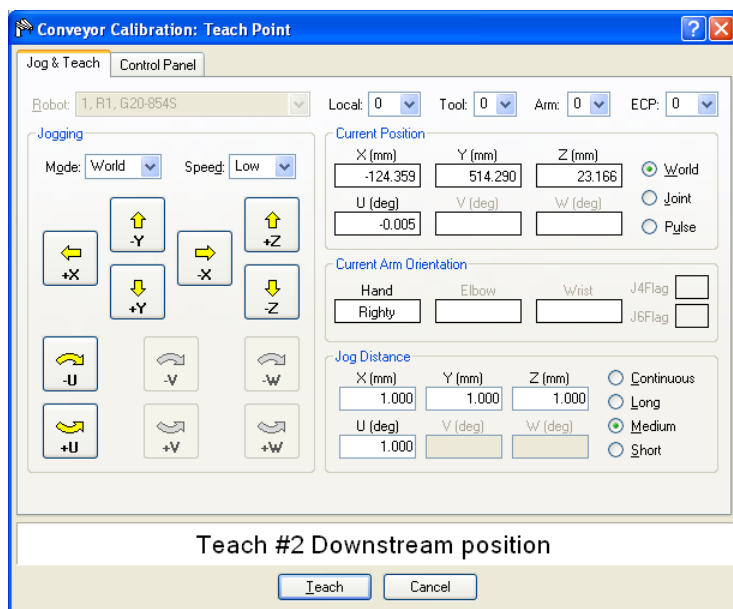
12. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



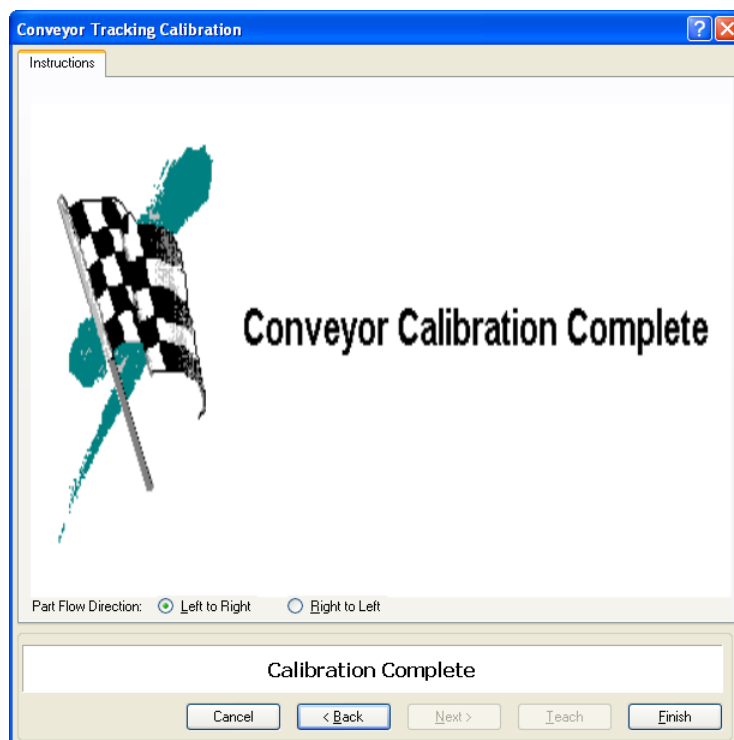
13. Place a part on one side of the conveyor near the downstream limit. This point is used to determine the tilt of the conveyor from side to side. Click the **Jog & Teach** button.



14. The Jog & Teach window will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



15. The calibration complete picture will be displayed. Click the **Finish** button.



### Sensor Conveyor Calibration (Circular conveyor)

Follow these steps to calibrate a circular sensor conveyor:

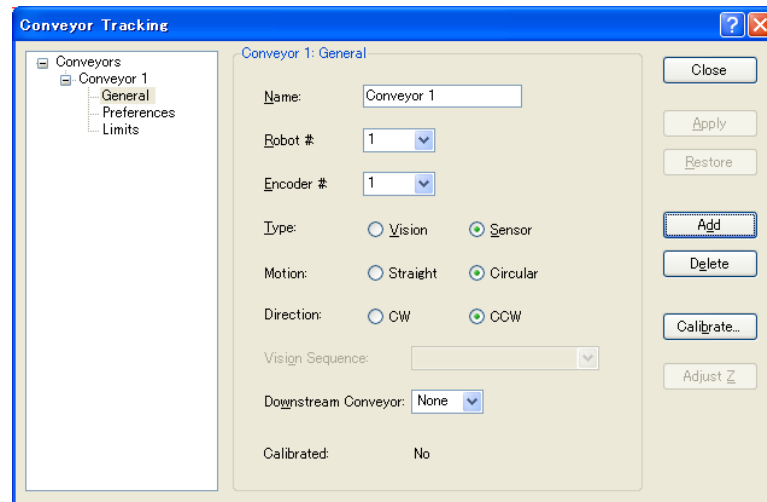


- When teaching part positions with the robot during calibration, it is important to position X, Y, and Z of each point accurately. The conveyor is calibrated in X, Y, Z, U, V, and W.
- To perform the fine calibration, in steps 10, 12, and 14, teach the position when the robot is directly above the parts and set as wide a distance as possible between the points being taught.

1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to calibrate.
3. Select **Sensor** for the [Type].
4. Select **Circular** for the [Motion].
5. Select the conveyor rotating direction for the [Direction].

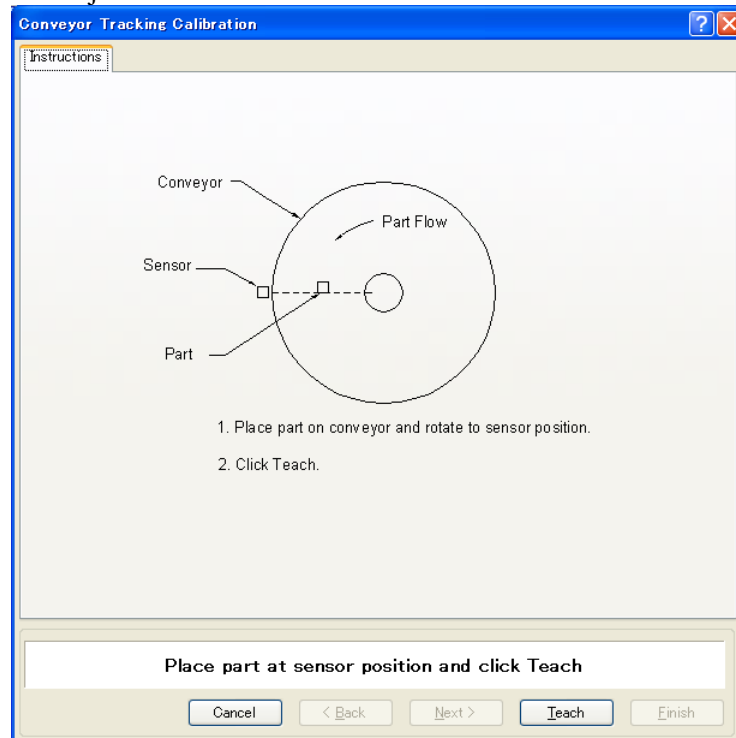


Be careful not to calibrate with a wrong direction, otherwise, the robot will not track the parts.

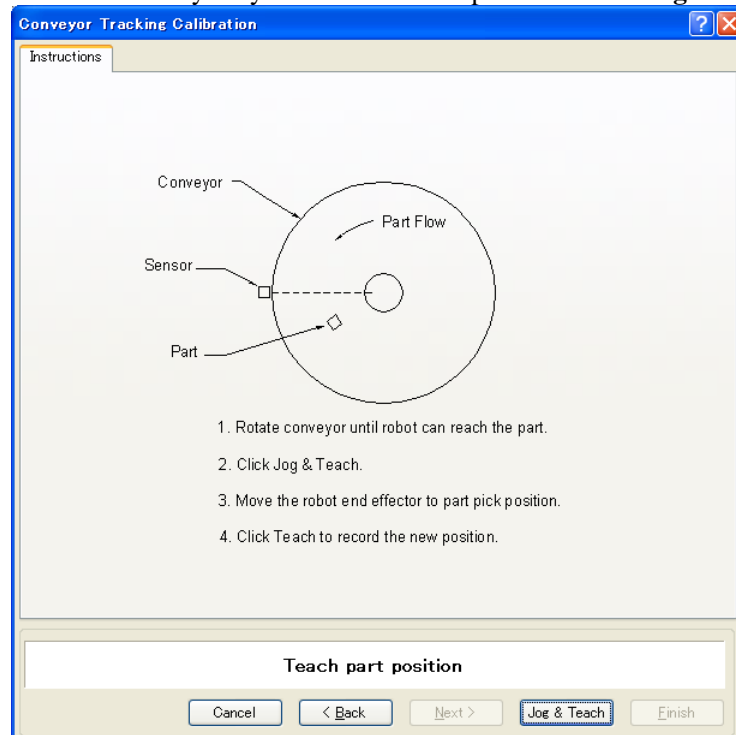


6. Click the **Apply** button.
7. Click the **Calibrate** button. The Conveyor Tracking Calibration wizard will appear. Follow the instructions for each step. Before you can proceed to the next step, you must click the **Teach** button. You can go back to previous steps using the **Back** button.
8. Check if the conveyor direction shown in the wizard is the same as the conveyor you want to use.

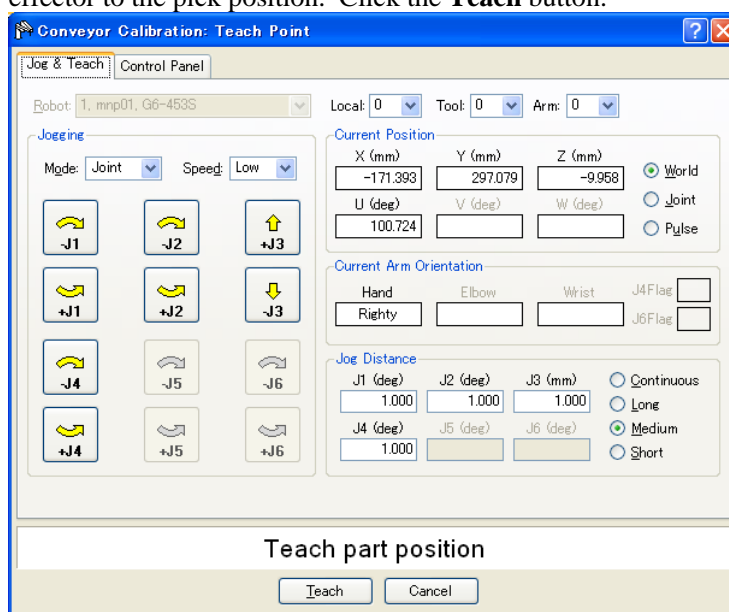
9. Place a part on the conveyor and move the conveyor toward the sensor until the sensor just turns on. Click the **Teach** button.



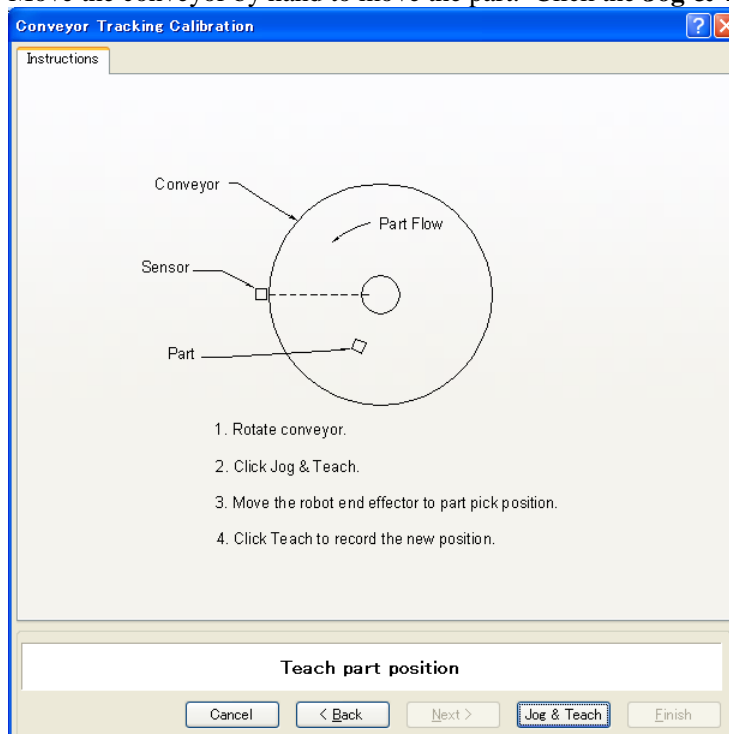
10. Move the conveyor by hand to move the part. Click the **Jog & Teach** button.



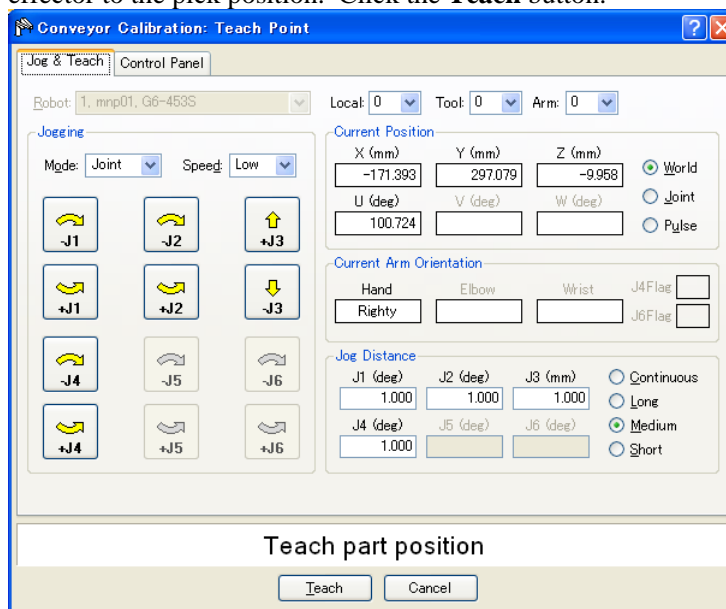
11. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



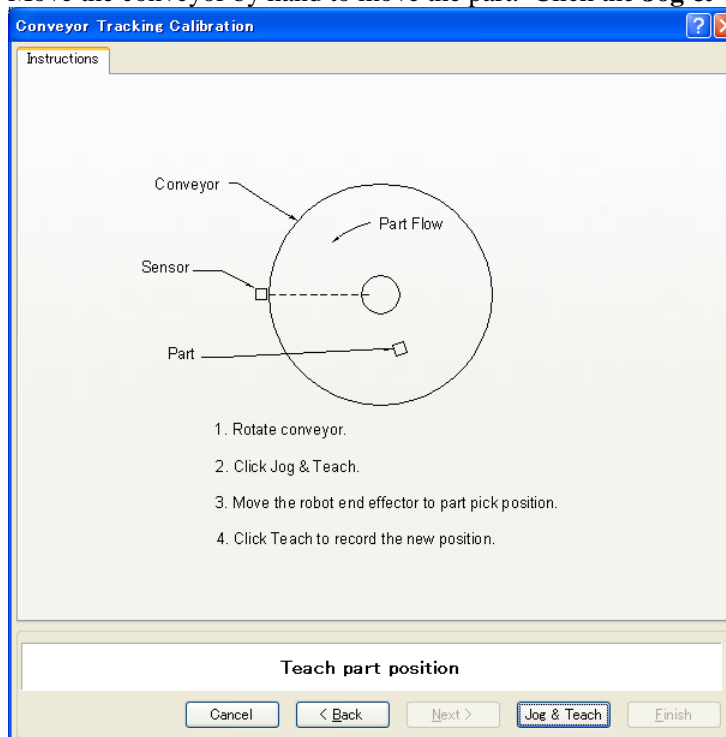
12. Move the conveyor by hand to move the part. Click the **Jog & Teach** button.



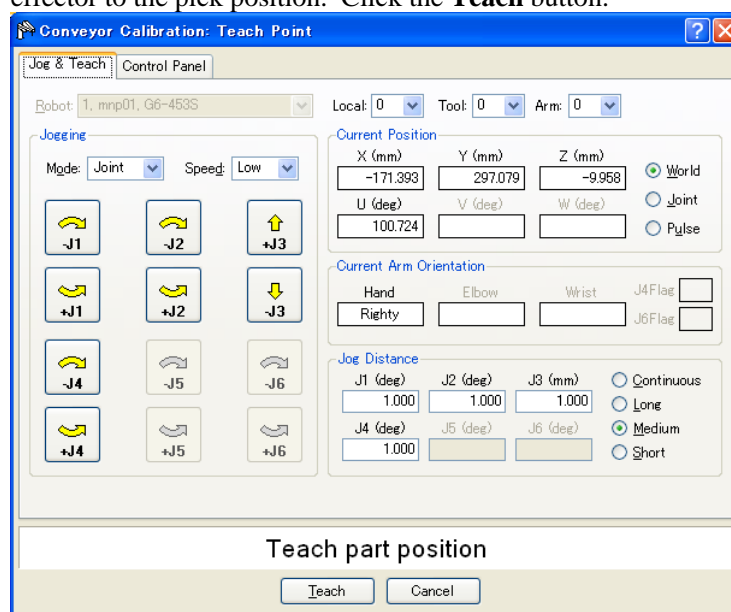
13. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



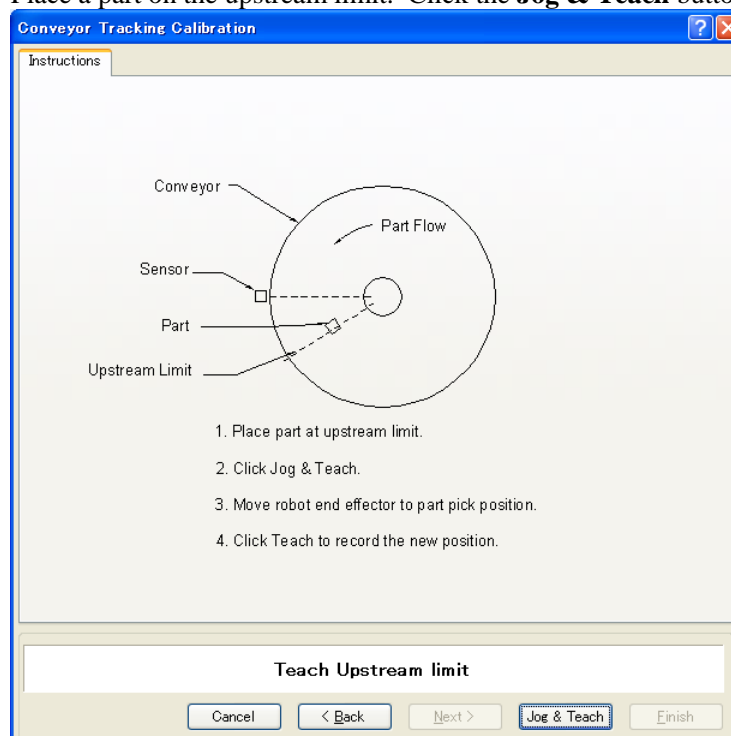
14. Move the conveyor by hand to move the part. Click the **Jog & Teach** button.



15. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.

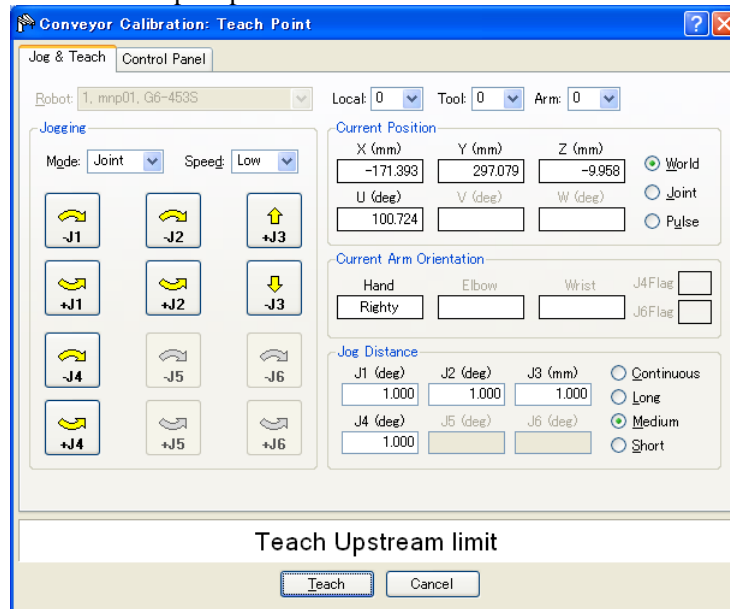


16. Place a part on the upstream limit. Click the **Jog & Teach** button.

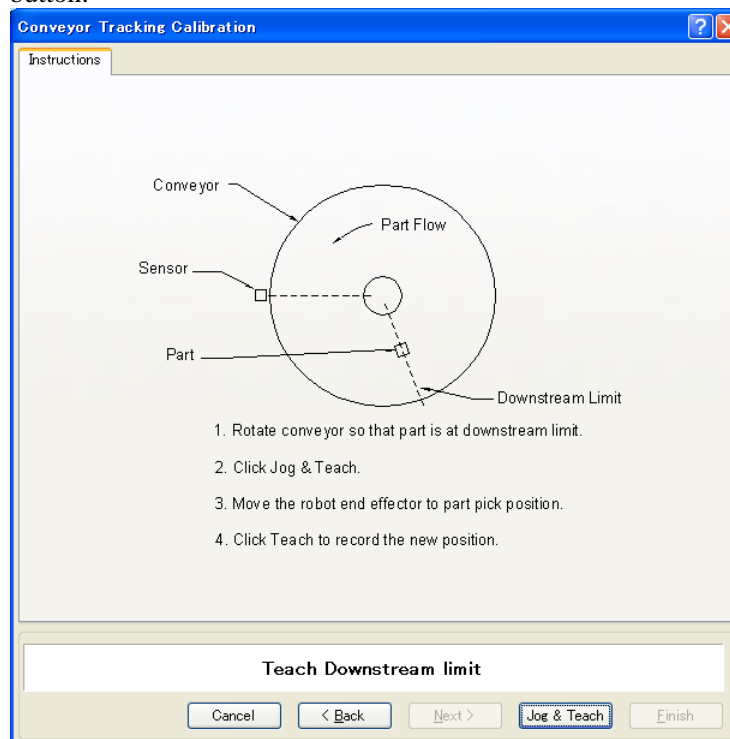




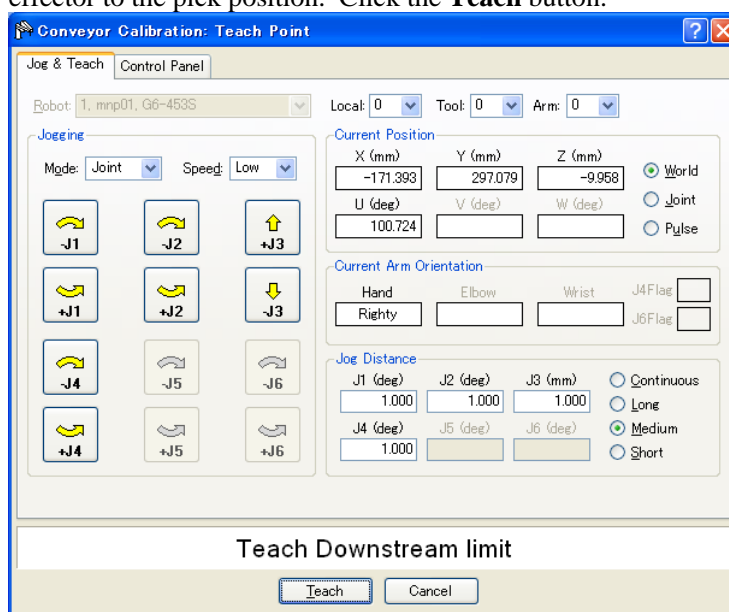
17. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



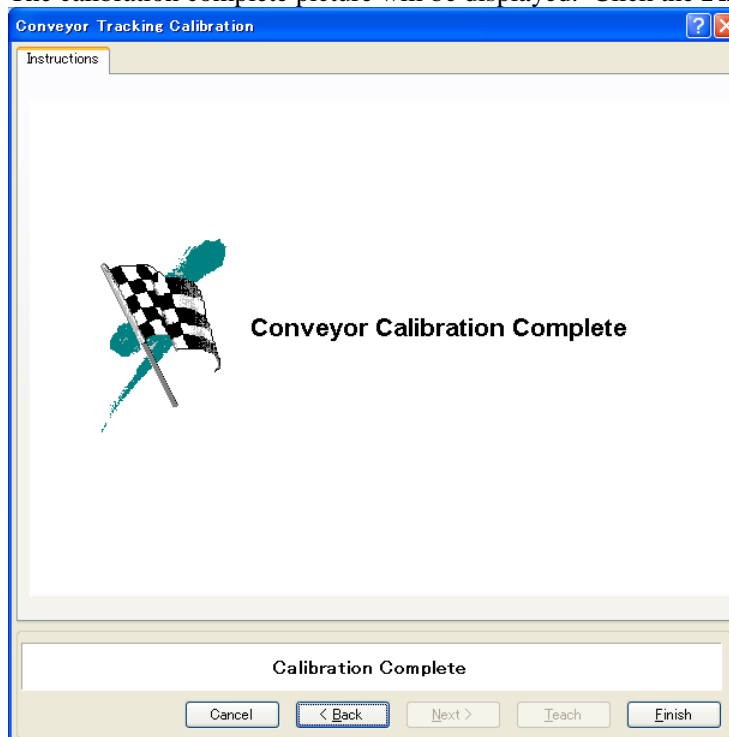
18. Move the conveyor so the part is at the downstream limit. Click the **Jog & Teach** button.



19. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



20. The calibration complete picture will be displayed. Click the **Finish** button.



### Sensor conveyor operation check

After the calibration, we recommend that you check if the sensor conveyor works properly. Use the Command Window and follow the procedure below.

In this section, the operation of Conveyor #1 is checked.

1. Clear the all queue data registered to the conveyor.  
`>Cnv_QueueRemove 1,all`
2. Place parts before the sensor area. Move the conveyor until the sensor turns ON.
3. Register a queue data.  
`>Cnv_QueueAdd 1,Cnv_Point(1,0,0)`
4. Move the conveyor until the parts reach the Pickup Area.
5. Pick up parts.  
`>Jump Cnv_QueueGet(1)`
6. Check if the robot end effector is over the center of the part to pick.
7. Move the conveyor and check if the robot follows the part. At this point, the end effector will be off the center of the part but this is no problem.
8. Stop the tracking motion.  
`>Cnv_AbortTrack`

In case the following symptoms occur, perform the calibration again.

- Robot cannot pick parts in the center.
- Robot cannot follow parts while the conveyor is moving.

**Sensor conveyor programming**

Typically, two tasks are used to operate a sensor conveyor. One task waits for a part to trip the sensor and add it to the conveyor queue. The other task checks for parts in the Pickup Area of the conveyor queue. When a part is in the Pickup Area, the robot is commanded to pick up the part and place it to the specified position.

```
Function main
    Xqt ScanConveyor      ' Task that registers queues
    Xqt PickParts         ' Task that tracks parts (queue)
Fend

Function ScanConveyor
    Double lpulse1        ' Previous latch pulse
    lpulse1 = Cnv_LPulse(1) ' Register the latch pulse as lpulse1
    Do
        ' Register a part as a queue only when it passes the sensor
        If lpulse1 <> Cnv_LPulse(1) Then
            Cnv_QueueAdd 1, Cnv_Point(1, 0, 0)
            lpulse1 = Cnv_LPulse(1) ' Update lpulse1
        EndIf
    Loop
Fend

Function PickParts
    OnErr GoTo ErrHandler
    Integer ErrNum
    WaitParts:
    Do
        ' Wait until a part (queue) enters the Pickup Area
        Wait Cnv_QueueLen(1, CNV_QUELEN_PICKUPAREA) > 0
        ' Start tracking the parts
        Jump Cnv_QueueGet(1)
        On gripper
        Wait .1
        ' Move the picked part to a specified place
        Jump place
        Off gripper
        Wait .1
        ' Clear the picked part (queue)
        Cnv_QueueRemove 1, 0
    Loop
    ' Clear the parts (queue) in the downstream side from the Pickup Area
    ' When error 4406 occurs, restore automatically
```

```
ErrorHandler:
    ErrNum = Err
    If ErrNum = 4406 Then
        Cnv_QueRemove 1, 0
        EResume WaitParts
        ' When an error other than error 4406 occurs, display the error
    Else
        Print "Error!"
        Print "No.", Err, ":", ErrMsg$(Err)
        Print "Line :", Erl(0)
    EndIf
Fend
```

### 15.13 Multiple Conveyors

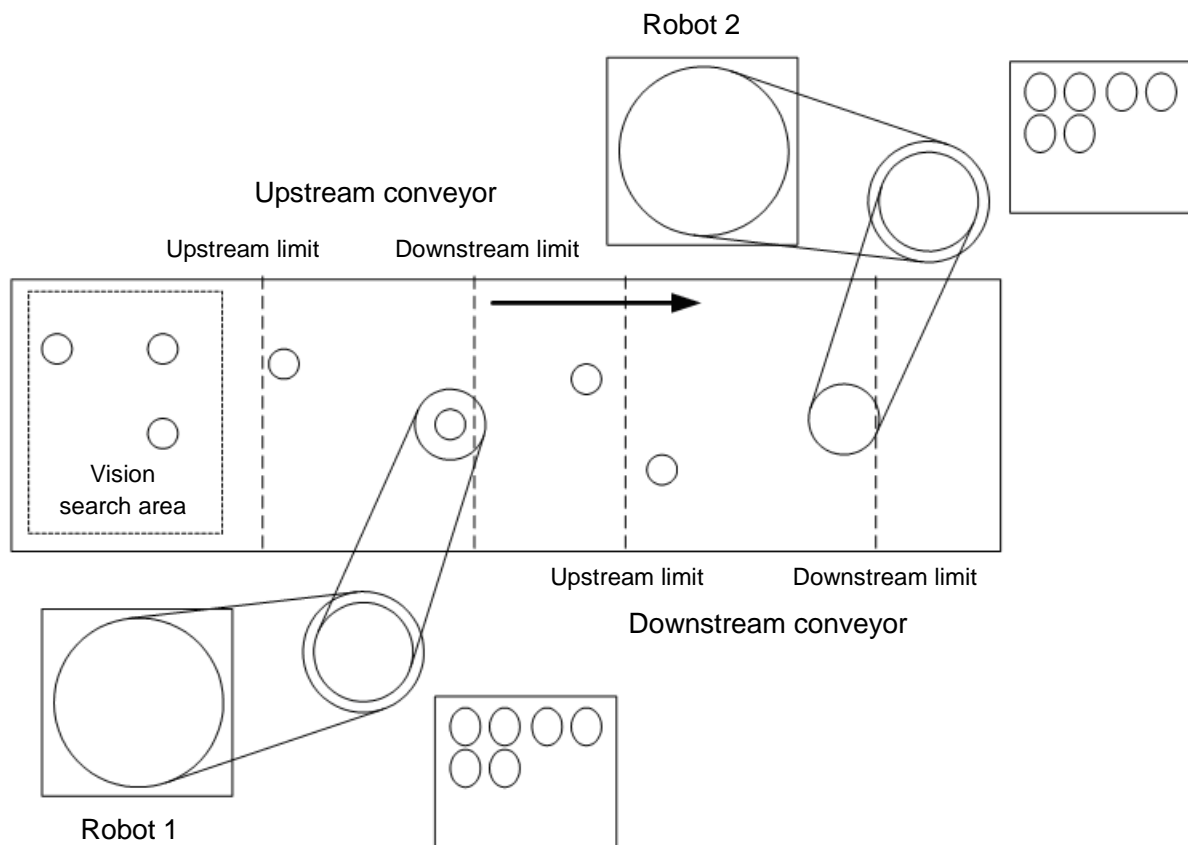
EPSON RC+ 6.0 supports multiple logical conveyors and robots. You can use multiple robots on one conveyor, or multiple robots with multiple conveyors.

This section describes a conveyor system that uses two or more robots with one conveyor and a conveyor system that uses one robot with two or more conveyors.

#### Multiple Conveyors

A system that uses multiple conveyors is a system that:

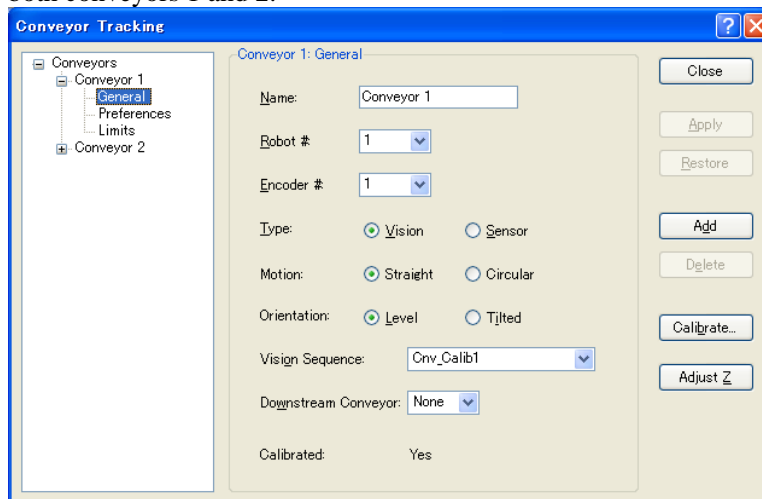
- Uses two or more robots and logical conveyors with one physical conveyor as shown below, and the parts that are not picked up by the first robot (upstream) can be picked up by the second robot (downstream).
- Uses several robots with one camera or sensor, one encoder, and one conveyor.



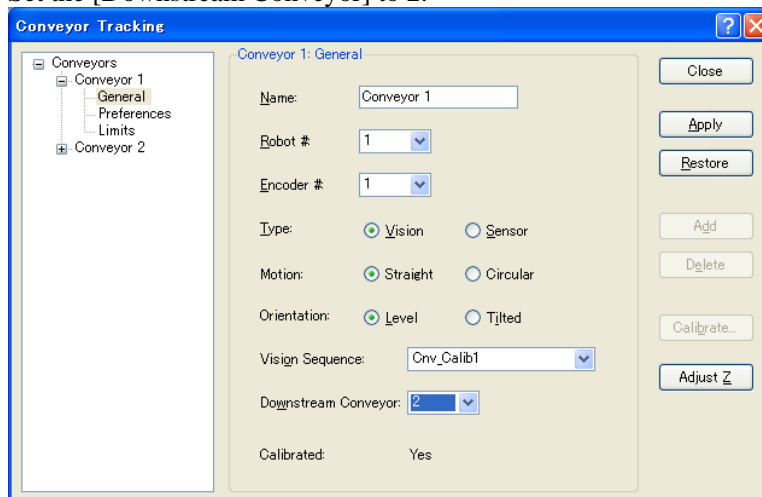
### How to Use Multiple Conveyors

When using multiple conveyors, set the upstream conveyor and downstream conveyor. Instructions for using multiple conveyors are described below.

1. Refer to *15.9 Creating Conveyors in a Project* and create conveyor 1 and conveyor 2. (Set the robot in the upstream side to Conveyor 1.)
2. For the [Encoder] and [Vision Sequence], set the same number and sequence for both conveyors 1 and 2.



3. Calibrate Conveyor 1.
4. Refer to *15.11 Vision Conveyors* or *15.12 Sensor Conveyors* and check the conveyor operation.
5. Set the [Downstream Conveyor] to 2.



6. Calibrate Conveyor 2.
7. Check the operation of Conveyor 2.
  - (7)-1 Clear the all queue data registered to each conveyor.
 

```
>Cnv_QueueRemove 1,all
```

```
>Cnv_QueueRemove 2,all
```
  - (7)-2 Place parts in the vision search area.
  - (7)-3 Execute the program “ScanConveyorStrobed(ScanConveyor)” and register a queue.

- (7)-4 Halt the program “ScanConveyorStrobed” and move the conveyor until the parts reach the Pickup Area.
- (7)-5 Stop the program “ScanConveyorStrobed” and move the conveyor until the part reaches the Pickup Area of the conveyor 2.
- (7)-6 Execute the command below to move the queue data from conveyor 1 to conveyor 2.  

>Cnv\_QueueMove 1,0
- (7)-7 Pick up the parts.  

>Jump Cnv\_Queueget(2)
- (7)-8 Check if the robot end effector is over the center of a part to pick. If the robot end effector is not over the center of a part, perform the calibration again.
- (7)-9 Move the conveyor and check if the robot follows the part. At this point, the end effector will be off the center of part but this is not a problem.
- (7)-10 Stop the tracking motion.  

>Cnv\_AbortTrack

8. The following program is a sample.

```

Function main
  Xqt ScanConveyorStrobed    'Task that registers queues
  'Task for the upstream robot to track parts (queue)
  Xqt PickParts1
  'Task for the downstream robot to track parts (queue)
  Xqt PickParts2
Fend

Function ScanConveyorStrobed
  Integer i, numFound, state
  Real x, y, u
  Boolean found
  'Turn OFF the camera shutter and I/O (conveyor trigger)
  Off trigger; off Cv_trigger
  Do
    'Search for a part on the conveyor
    VRun FindParts
    'Turn ON the camera shutter and I/O (conveyor trigger)
    On Trigger; On Cv_Trigger
  Do
    VGet FindParts.AcquireState, state
  Loop Until state = 3
  VGet FindParts.Parts.NumberFound, numFound
  'Register the part that has been found in the queue of conveyor 1
  For i = 1 to numFound

```



```

        VGet FindParts.Parts.CameraXYU(i), found, x, y, u
        Cnv_QueAdd 1, Cnv_Point(1, x, y)
    Next I
    'Turn OFF the camera shutter and I/O (conveyor trigger)
    Off Trigger; Off Cv_Trigger
    Wait .1
    Loop
Fend

Function PickParts1
    OnErr GoTo ErrHandler
    Integer ErrNum
    WaitParts:
    Do
        ' Wait until a part (queue) enters the Pickup Area
        Wait Cnv_QueLen(1, CNV_QUELEN_PICKUPAREA) > 0
        ' Start tracking the part
        Jump Cnv_QueGet(1)
        On gripper
        Wait .1
        ' Move the picked part to a specified place
        Jump place
        Off gripper
        Wait .1
        ' Clear the picked part (queue)
        Cnv_QueRemove 1, 0
    Loop
    ' Move the parts (queue) in the downstream side from the Pickup Area of conveyor
    1 to the conveyor 2
    ' When error 4406 occurs, restore automatically
    ErrHandler:
        ErrNum = Err
        If ErrNum = 4406 Then
            Cnv_QueMove 1, 0
            EResume WaitParts
        ' When an error except error 4406 occurs, display the error
        Else
            Print "Error!"
            Print "No.", Err, ":", ErrMsg$(Err)
            Print "Line :", Erl(0)
        EndIf
    Fend

```

```

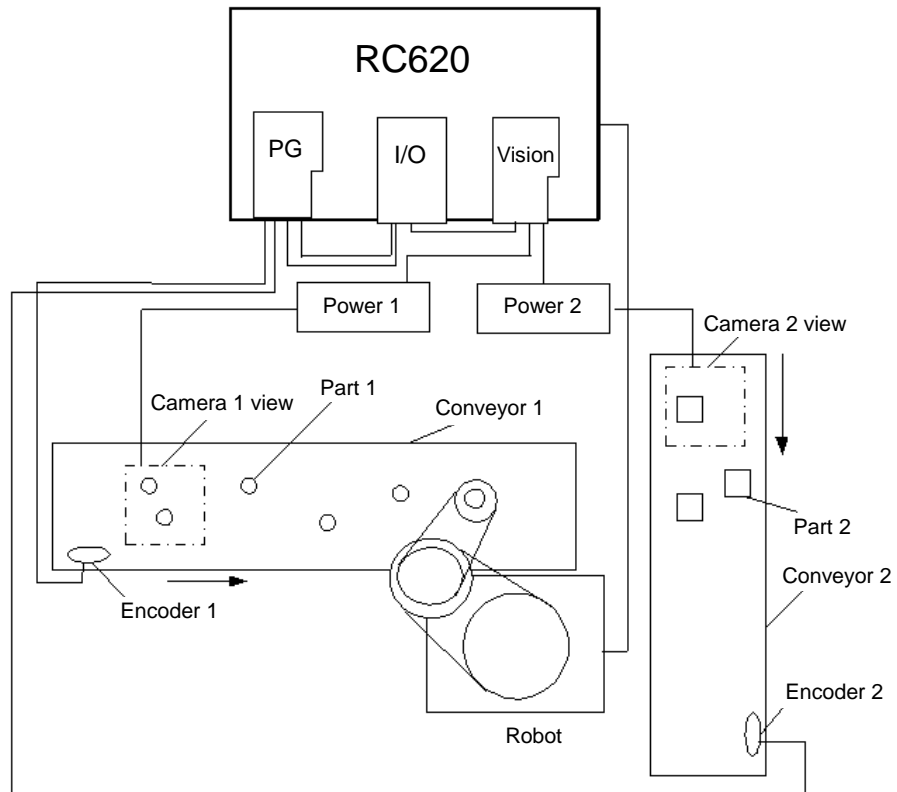
Function PickParts2
    OnErr GoTo ErrHandler
    Integer ErrNum
    WaitParts:
    Do
        ' Wait until a part (queue) enters the Pickup Area
        Wait Cnv_QueLen(2, CNV_QUELEN_PICKUPAREA) > 0
        ' Start tracking the part
        Jump Cnv_QueGet(2)
        On gripper
        Wait .1
        ' Move the picked part to a specified place
        Jump place
        Off gripper
        Wait .1
        ' Clear the picked part (queue)
        Cnv_QueRemove 2, 0
    Loop
    ' Clear the parts (queue) in the downstream side from the Pickup Area of conveyor 2
    ' When error 4406 occurs, restore automatically
    ErrHandler:
        ErrNum = Err
        If ErrNum = 4406 Then
            Cnv_QueRemove 2, 0
            EResume WaitParts
        ' When an error except error 4406 occurs, display the error
        Else
            Print "Error!"
            Print "No.", Err, ":", ErrMsg$(Err)
            Print "Line :", Erl(0)
        EndIf
    Fend

```

### Conveyor Tracking for Several Conveyors

This section describes a conveyor system where one robot picks up “Part 1” from Conveyor 1 and puts the picked parts above “Parts 2” on Conveyor 2 as shown in the figure below.

In this conveyor system, each conveyor needs one encoder and camera (sensor).



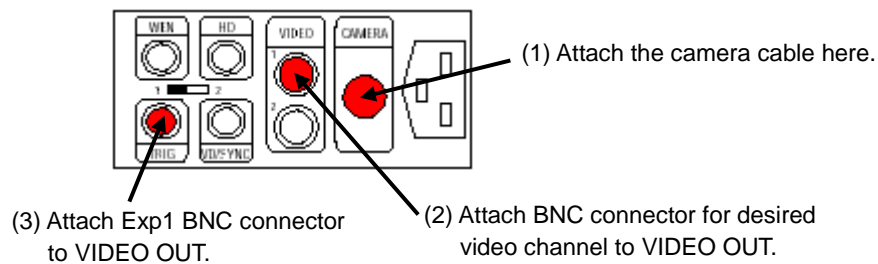
### Wiring Two Asynchronous Reset Cameras

The wiring for a vision system using two asynchronous cameras (Frame Grabber) is described below.

#### Vision System Wiring

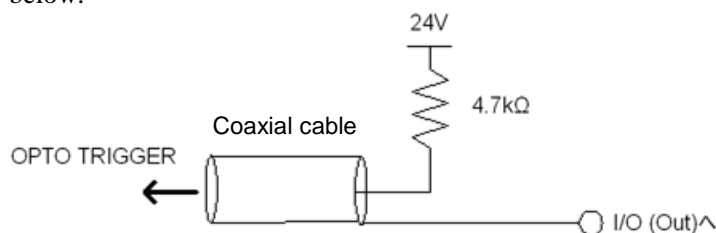
(Example: Sony XC-HR50)

- (1) Attach one end of the camera 1 cable to the camera to be used as camera 1 and the other end of the cable to the rear panel receptacle of the Camera Power Junction box 1 (see the figure below).  
Next, attach one end of the camera 2 cable to the camera to be used as camera 2 and the other end of the cable to the rear panel receptacle of the Camera Power Junction box 2.



DC-700 Camera Power Junction Box (Rear View)

- (2) Pick up the BNC to D-Sub cable that is now attached to the frame grabber. Locate the cable to the Camera Power Junction box 1. Attach the Video 1 BNC connector cable to the VIDEO OUT BNC female connector on the Camera Power Junction box 1.  
Attach the Video 2 BNC connector cable to the VIDEO OUT BNC female connector on the Camera Power Junction box 2.
- (3) Attach the Exp 1 BNC connector cable branching to the TRIG connectors on the Camera Power Junction boxes 1 and 2.
- (4) Connect the OPTO TRIGGER BNC connector cable and the coaxial cable as shown below.



- (5) Refer to the option manual, *Vision Guide 6.0 - Appendix A: Camera Interfaces* and set the camera external setting to the asynchronous reset mode.

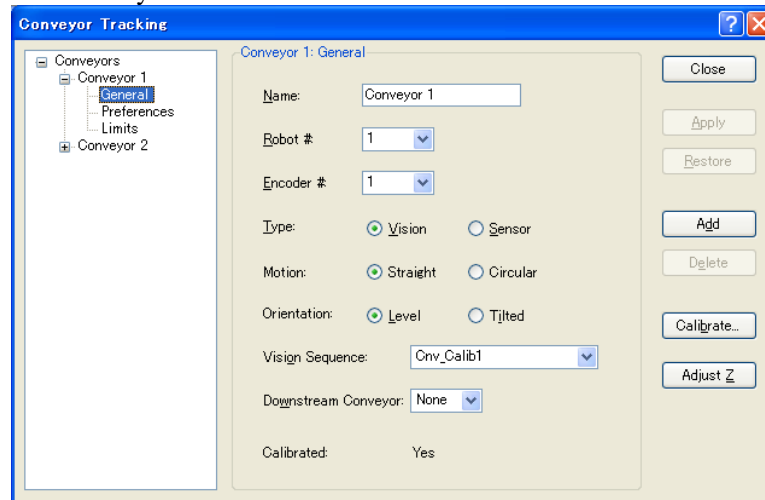


When you use the Smart Camera that needs the wiring not described above, refer to *Vision Guide 6.0 - 2. Installation* and attach the cables.

## How to Use Several Conveyors

The usage of several conveyors is described below.

1. Refer to *15.9 Creating Conveyors in a Project* and create Conveyor 1 and Conveyor 2. (Set the robot in the upstream side to the Conveyor 1.)
2. For the [Encoder] and [Vision Sequence], set the different number and sequence for each conveyor 1 and 2.



3. Calibrate Conveyor 1.
4. Refer to *15.11 Vision Conveyors* or *15.12 Sensor Conveyors* and check the conveyor operation.
5. Calibrate Conveyor 2.
6. Check the operation of Conveyor 2.
7. The following program is a sample.

```
Function main
```

```
    'Task that registers parts in the queues
```

```
    Xqt ScanConveyorStrobed
```

```
    'Task that tracks parts (queue)
```

```
    Xqt PickParts
```

```
End
```

```
Function ScanConveyorStrobed
```

```
    Integer i, j, numFound, state
```

```
    Real x, y, u
```

```
    Boolean found
```

```
    'Turn OFF the camera shutter and I/O (conveyor trigger)
```

```
    'Cv_trigger1: Conveyor 1, Cv_trigger2: Conveyor 2
```

```
    Off trigger; off Cv_trigger1; off Cv_trigger2
```

```
    Do
```

```
        'Register the parts (queue) of the Conveyor 1
```

```
            'Search for a part on the conveyor
```

```
            VRun FindParts1
```

```
'Turn ON the camera shutter and I/O (conveyor trigger)
On Trigger; On Cv_Trigger1
Do
    VGet FindParts1.AcquireState, state
Loop Until state = 3
VGet FindParts1.Parts.NumberFound, numFound
'Register the part that has been shot as a queue
For i = 1 to numFound
    VGet FindParts.Parts.CameraXYU(i), found, x, y, u
    Cnv_QueAdd 1, Cnv_Point(1, x, y)
Next i
'Turn OFF the camera shutter and I/O (conveyor trigger)
Off Trigger; Off Cv_Trigger
Wait .1

'Register the parts (queue) of the Conveyor 2
'Search for part on the conveyor
VRun FindParts2
'Turn ON the camera shutter and I/O (conveyor trigger)
On Trigger; On Cv_Trigger1
Do
    VGet FindParts2.AcquireState, state
Loop Until state = 3
VGet FindParts2.Parts.NumberFound, numFound
'Register the part that has been shot as a queue
For j = 1 to numFound
    VGet FindParts2.Parts.CameraXYU(j), found, x, y, u
    Cnv_QueAdd 2, Cnv_Point(2, x, y)
Next j
'Turn OFF the camera shutter and I/O (conveyor trigger)
Off Trigger; Off Cv_Trigger2
Wait .1

Loop
Fend

Function PickParts
    OnErr GoTo ErrHandler
    Integer ErrNum
    MemOff 1
    MemOff 2
    Do
```

```

WaitPickup1:
    'Tracking of Conveyor 1
    'Turn ON the memory I/O when the Conveyor 1 tracking phase starts
    MemOn 1
    'Clear the parts (queue) in the downstream side from the downstream limit
    Do While Cnv_QueLen(1 CNV_QUELEN_DOWNSTREAM) > 0
        Cnv_QueRemove 1, 0
    Loop
    'Move to the standby position when there is no part (queue) in the Pickup Area
    If Cnv_QueLen(1, CNV_QUELEN_PICKUPAREA) = 0 Then
        Jump place
    EndIf
    'Wait until a part (queue) enters the Pickup Area
    Wait Cnv_QueLen(1, CNV_QUELEN_PICKUPAREA) > 0
    'Start tracking the parts
    Jump Cnv_QueGet(1)
    On gripper
    Wait .1
    'Clear the picked part (queue)
    Cnv_QueRemove 1,0
    'Finish the Conveyor 1 tracking phase
    MemOff 1

    'Tracking of the Conveyor 2
    WaitPickup2:
    'Start the Conveyor 2 tracking phase
    MemOn 2
    'Clear the parts (queue) in the downstream side from the downstream limit
    Do While Cnv_QueLen(2, CNV_QUELEN_DOWNSTREAM) > 0
        Cnv_QueRemove 2, 0
    Loop
    'Move to the standby position when there is no part (queue) in the Pickup Area
    If Cnv_QueLen(2, CNV_QUELEN_PICKUPAREA) = 0 Then
        Jump place
    EndIf
    'Wait until a part (queue) enters the Pickup Area
    Wait Cnv_QueLen(2 CNV_QUELEN_PICKUPAREA) > 0
    'Start tracking the parts
    Jump Cnv_QueGet(2)
    Off gripper
    Wait .1
    'Clear the picked part (queue)

```

```
Cnv_QueueRemove 2, 0
' Finish the Conveyor 2 tracking phase
  MemOff 2
Loop
' Parts (queue) in the downstream side from the Pickup Area
' When error 4406 occurs, restore automatically
ErrorHandler:
  ErrNum = Err
  If ErrNum = 4406 Then
    If MemSw(1) = On Then
      Cnv_QueueRemove 1
      EResume WaitPickup1
    EndIf
    If MemSw(2) = On Then
      Cnv_QueueRemove 2
      EResume WaitPickup2
    EndIf
    ' When error 4406 occurs, restore automatically
  Else
    Print "Error!"
    Print "No.", Err, ":", ErrMsg$(Err)
    Print "Line :", Erl(0)
  EndIf
Fend
```



### To pick up the same parts with multiple robots

When both robots are handling the same type of parts, you will want to move the parts in the conveyor 1 downstream to the conveyor 2 upstream. This allows parts that were not picked up by robot 1 to be picked up by robot 2.

The following example is of Vision conveyor and Multi-conveyor using two robots in one conveyor.

1. Set one conveyor for each robot.  
(Robot 1 - Conveyor 1, Robot 2 - Conveyor 2)
2. Set the downstream conveyor to the Conveyor 1.
3. Create the following program.

```
Function main

    Xqt ScanConveyorStrobed
    Xqt PickParts1
    Xqt PickParts2

Fend

Function ScanConveyorStrobed

    Integer i, numFound, state
    Real x, y, u
    Boolean found
    Off Trigger; Off Cv_trigger
    Do
        VRun FindParts
        On Trigger; On Cv_Trigger    'Turn on the external
                                     trigger of camera and
                                     conveyor
    Do
        VGet FindParts.AcquireState, state
    Loop Until state = 3
    VGet FindParts.Parts.NumberFound, numFound
    For i = 1 to numFound
        VGet FindParts.Parts.CameraXYU(i), found, x, y, u
        Chv_QueAdd 1, Chv_Point(1, x, y)
    Next i
    Off Trigger; Off Cv_Trigger    'Trigger OFF
    Wait .1
    Loop
Fend
```

```
Function PickParts1

    Robot 1
    OnErr GoTo ErrHandler
    Integer ErrNum

WaitParts:
    Do
        Wait Cnv_QueLen(1, CNV_QUELEN_PICKUPAREA) > 0
        Jump Cnv_QueGet(1)
        On gripper
        Wait .1
        Jump place
        Off gripper
        Wait .1
        Cnv_QueRemove 1, 0
    Loop

ErrHandler:
    ErrNum = Err
    If ErrNum = 4406 Then
        Cnv_QueMove 1, 0
        EResume WaitParts
    Else
        Print "Erro occurs!"
        Print "No.", Err, ":", ErrMsg$(Err)
        Print "Line :", Erl(0)
    EndIf
Fend
```

```

Function PickParts2
    Robot 2
    OnErr GoTo ErrHandler
    Integer ErrNum
WaitParts:
    Do
        Wait Cnv_QueLen(2, CNV_QUELEN_PICKUPAREA) > 0
        Jump Cnv_QueGet(2)
        On gripper
        Wait .1
        Jump place
    Off gripper
    Wait .1
    Cnv_QueRemove 2, 0
    Loop
ErrHandler:
    ErrNum = Err
    If ErrNum = 4406 Then
        Cnv_QueRemove 2, 0
        EResume WaitParts
    Else
        Print "Error occurs!"
        Print "No.", Err, ":", ErrMsg$(Err)
        Print "Line :", Erl(0)
    EndIf
Fend

```

### To pick up different part types with multiple robots

When both robots are handling different types of parts, you can create a conveyor for each robot. When the vision system finds parts, it adds the part positions to the correct queue, depending on the part type.

### 15.14 Adjusting the Z value

You can adjust the conveyor Z value after the calibration is completed.

Adjusting the Z value is a function to change the work pickup height that has been determined during calibration.

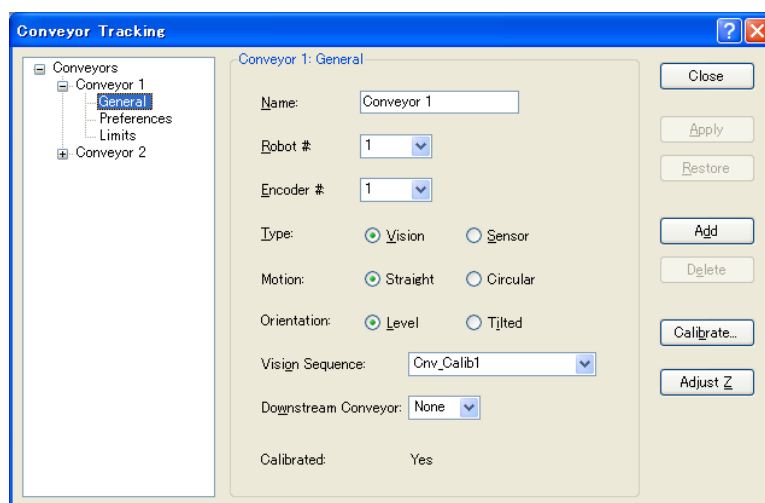
In the following cases, adjust the Z value:

- To use a pickup area that is different from the one defined during calibration.
- The tool has been changed on the robot after calibration.

To adjust the Z value:

1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to edit.
3. Click on **General**.
4. The dialog shown below appears.

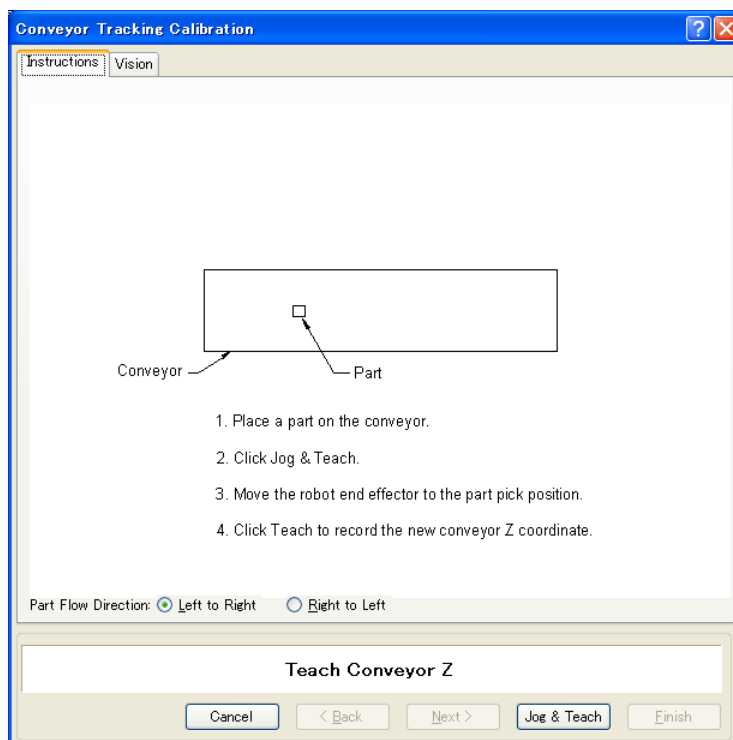
Click on the **Adjust Z** button.



5. The dialog shown below appears.

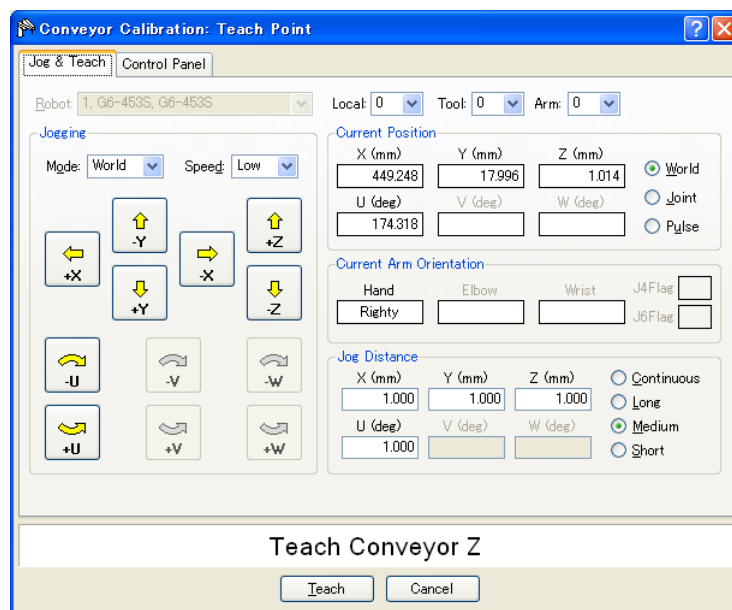
Place a part on the conveyor in the robot motion range.

Click on the **Jog & Teach** button.



6. The Jog & Teach dialog will appear. Click the jog buttons to move the robot end effector to the pick position.

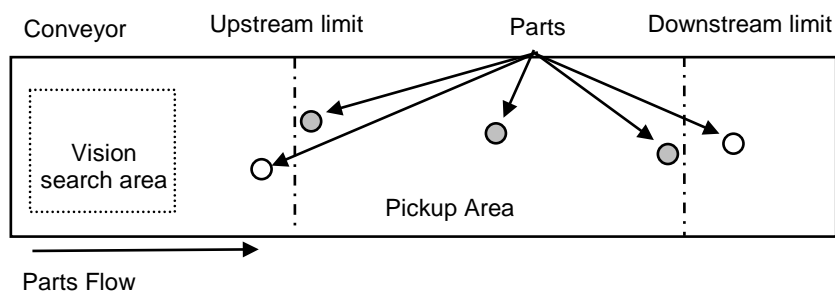
Click the **Teach** button.



## 15.15 Pickup Area

The Pickup Area is the range where the robot can pick up parts.

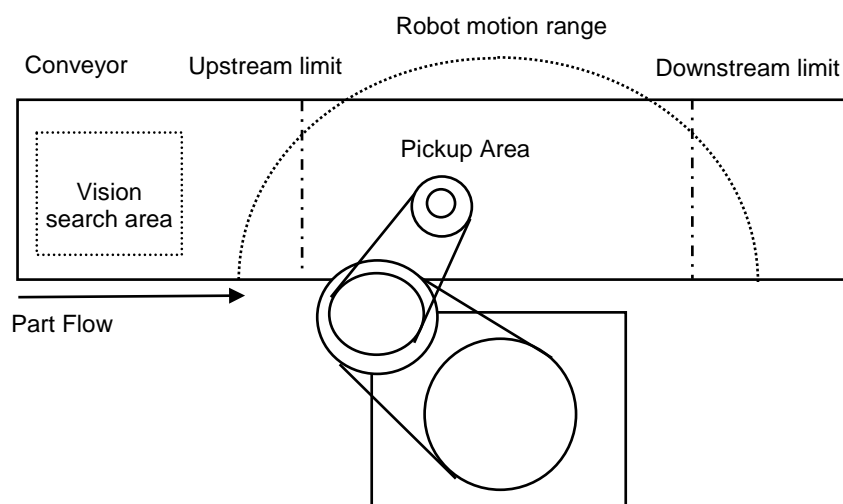
In the figure below, the robot can pick up the parts in gray.



If the Pickup Area is not appropriate, the robot cannot pick up parts. Follow the steps and cautions below to carefully set the Pickup Area.

To define the Pickup Area:

1. After calibration, the Pickup Area will be defined as shown in the following figure. Note that the positions of upstream limit and downstream limit depend on the positions you teach during the calibration.

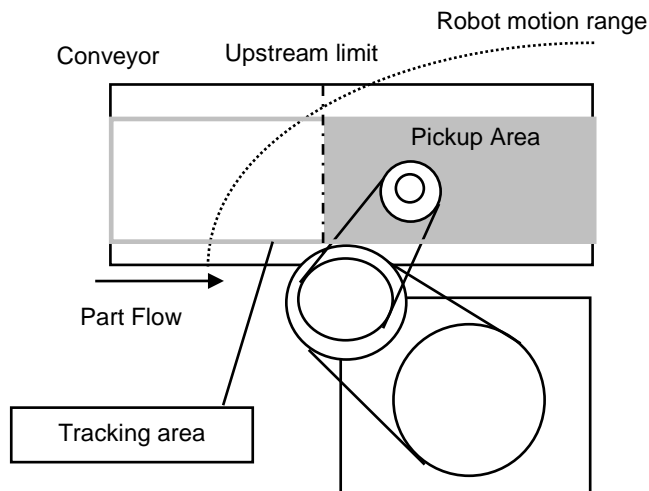


Decide the upstream limit position.

The robot starts pickup from the line defined by the upstream limit. The Pickup Area from the upstream limit must be within the robot motion range. (See the figure below.)



The robot does not start pickup until parts cross the upstream limit. If you set the upstream limit in uppermost position, you can reduce the robot standby time.

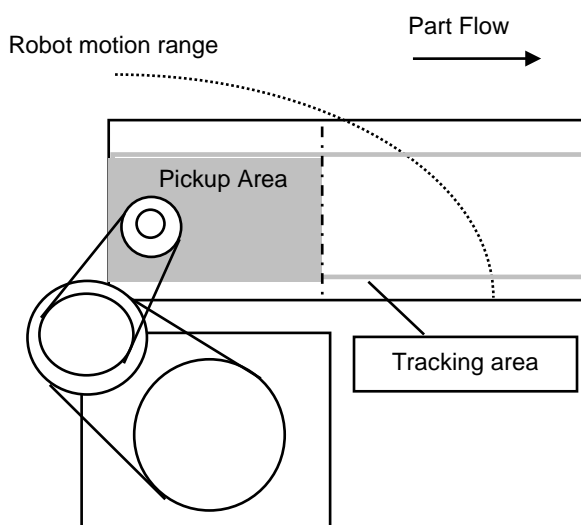


## 2. Decide the downstream limit position.

Once the robot starts pickup, it continues its operation even over the downstream limit to complete the whole operation. Therefore, set the downstream limit in uppermost position so that the robot can operate within its motion range until it completes the operation. (See the figure below.)



The downstream limit position depends on the conveyor speed and robot position when it starts pickup. If the robot goes over the motion range during the operation, move the downstream limit to upper side.



### Changing the Upstream / Downstream limits positions

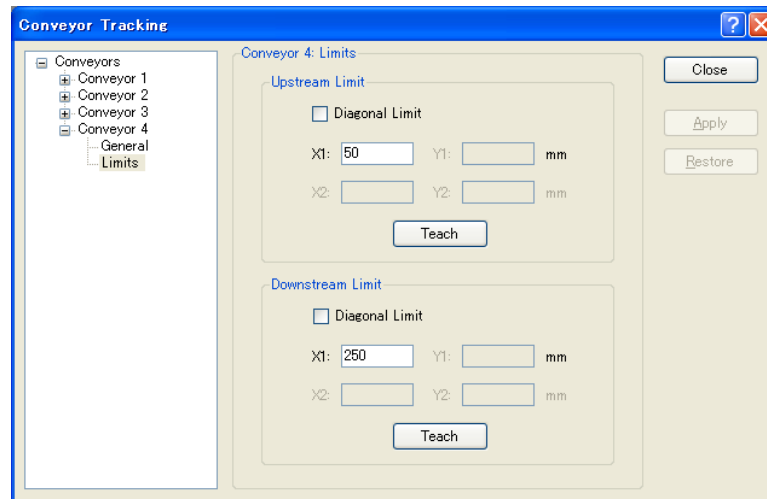
To change the upstream limit and downstream limit positions, follow the steps below.

To change the Upstream Limit:

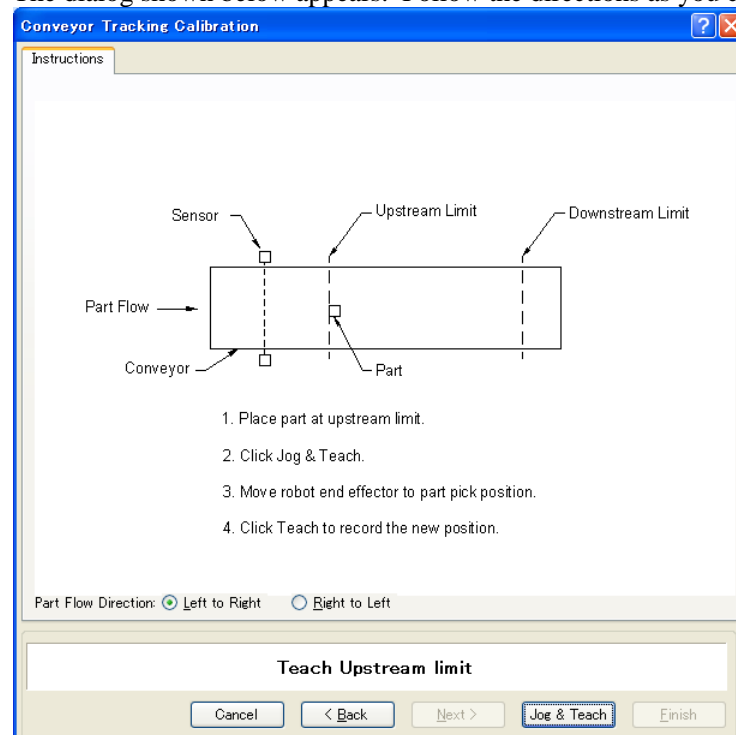
1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to edit.
3. Click **Limits**.
4. The dialog shown below appears.

Edit the [Upstream Limit] value.

To define the X1 value, enter a value directly or use Jog & Teach. Entering values directly is for fine adjustment.



5. When you directly specify the value, enter the value in the box and click **Apply**.
6. When you use Jog & Teach, click the **Teach** button.
7. The dialog shown below appears. Follow the directions as you do during calibration.



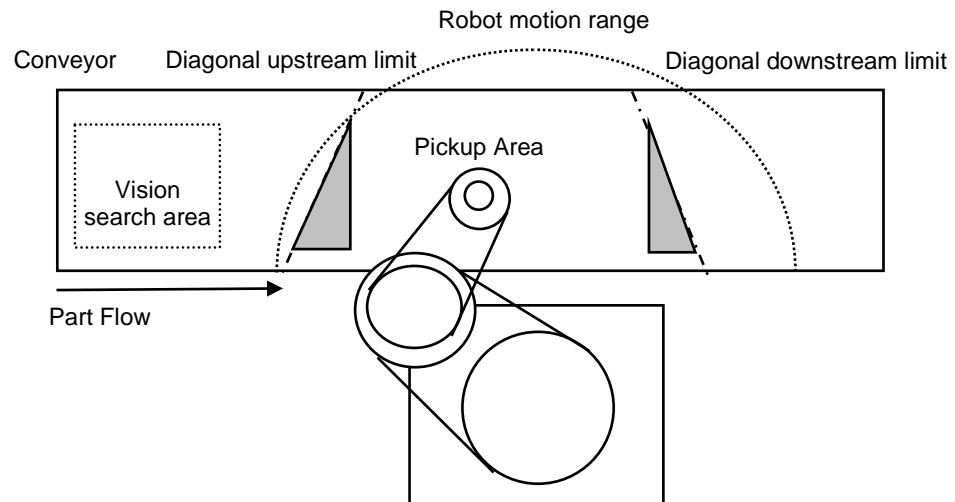


To change the downstream limit, edit the [Downstream Limit] value the same as described for the upstream limit.

### Diagonal Upstream / Downstream Limits

After the calibration, you can set the dividing lines for the Pickup Area (upstream limit / downstream limit) directed diagonally to the part flow.

When you change the dividing lines to diagonal positions, the Pickup Area also changes as shown below. The area indicated in gray is widened by changing the dividing lines to diagonal positions. In addition, diagonal dividing lines are called the diagonal upstream / downstream limits.



The following are the advantages you can get by widening the Pickup Area.

- Reduce robot standby time by widening the upper side Pickup Area.
- Less possibility of missing parts which flow longer after the downstream limit.

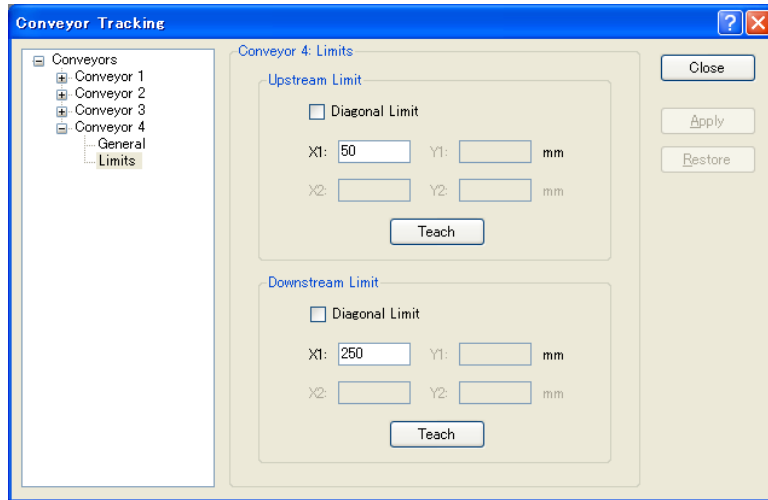


If there are too many parts on the conveyor for the robot to pick up, it only makes the robot move for longer distance and longer time and the number of parts the robot can pick up may decrease, even in a widened Pickup Area. For such a case, you may want to consider using the multiple conveyors system. To use the multiple conveyors system, set a short Pickup Area in the upstream conveyor and wide Pickup Area in the downstream conveyor.

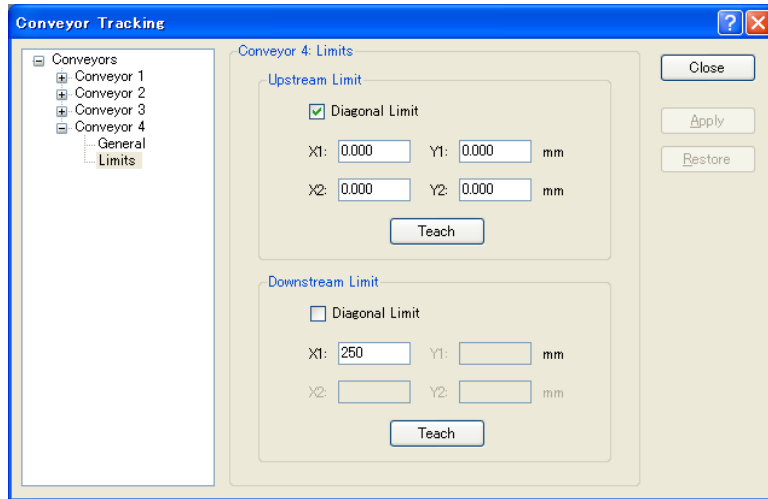
The robot capacity (how fast or how many parts robot can pick up) depends on the Pickup Area width, robot standby position, and conveyor speed.

To set the diagonal upstream limit:

1. Select Tools | Conveyor Tracking.
2. Select the conveyor you want to edit.
3. Click on **Limits**.
4. The dialog shown below appears.



Check the <Diagonal Limit> check box in the [Upstream Limit] area and click **Apply**.  
The following dialog appears.

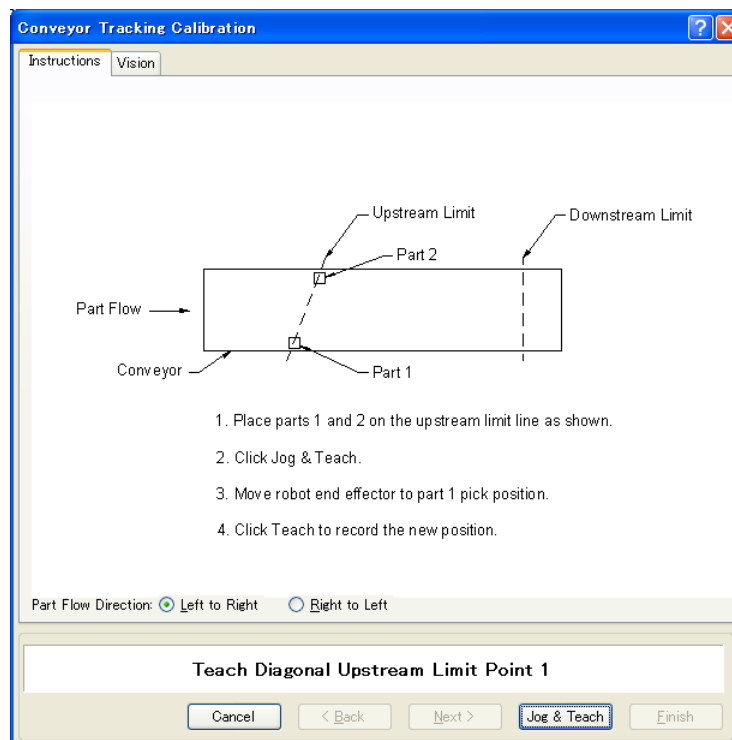


To define the values for X1, Y1, X2, Y2, enter the values directly or use Jog & Teach. Entering values directly is for fine adjustment.

5. When you directly specify the values, enter the values in the boxes and click **Apply**.

6. When you use Jog & Teach, click **Teach**.

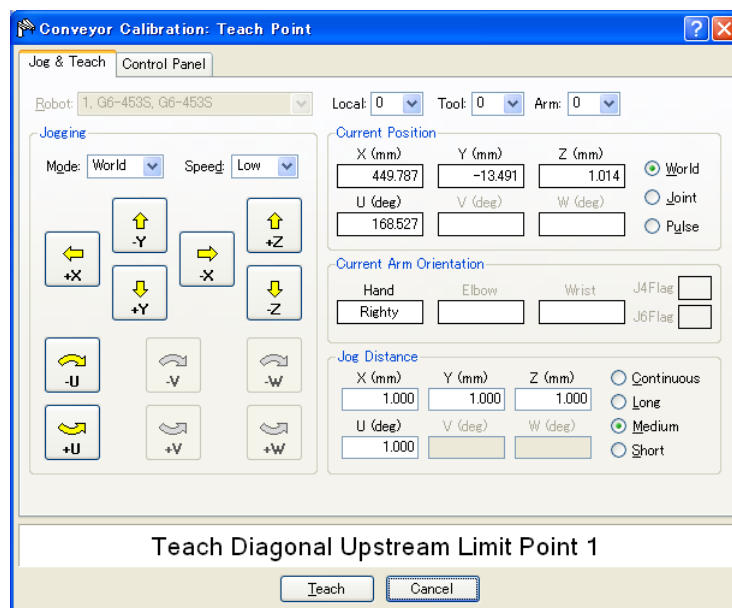
The dialog shown below appears.



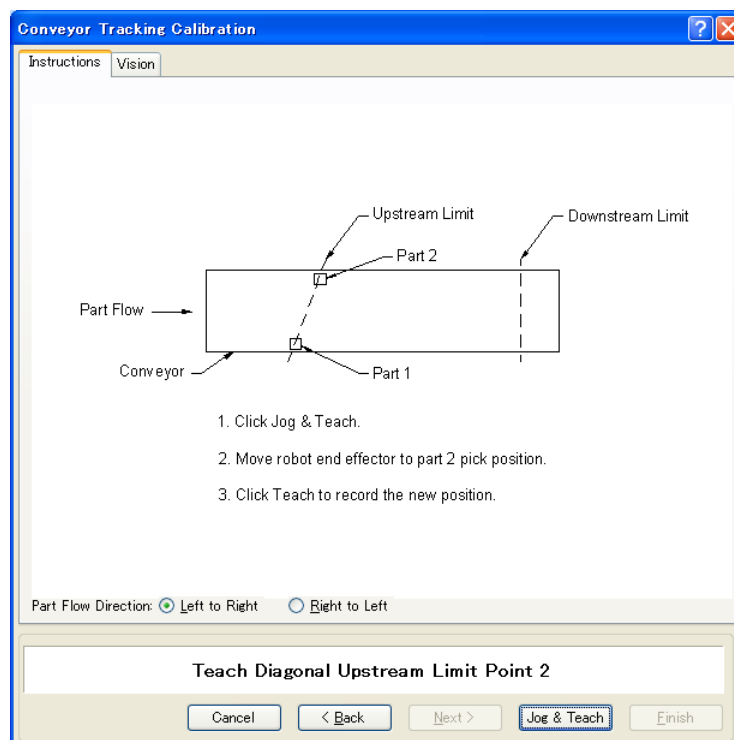
7. Place two parts on the conveyor.

Click the **Jog & Teach** button.

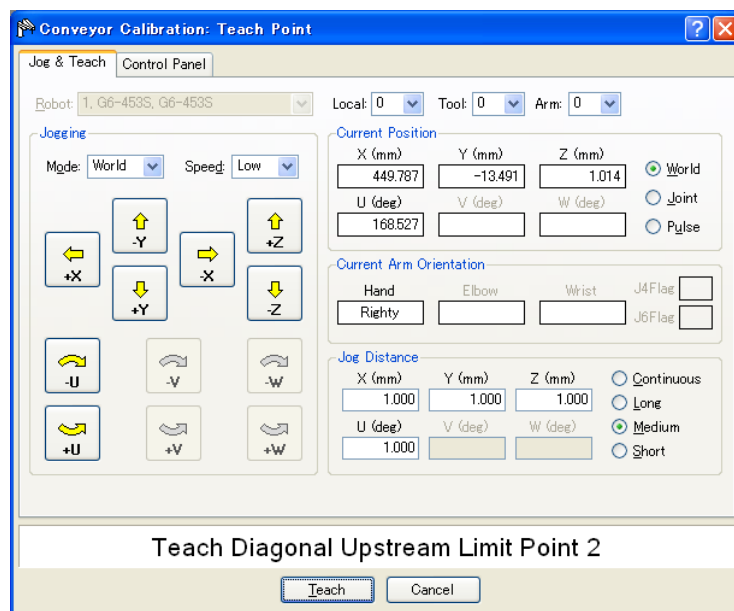
8. The Jog & Teach dialog appears. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



9. The dialog shown below appears. Click the **Jog & Teach** button.

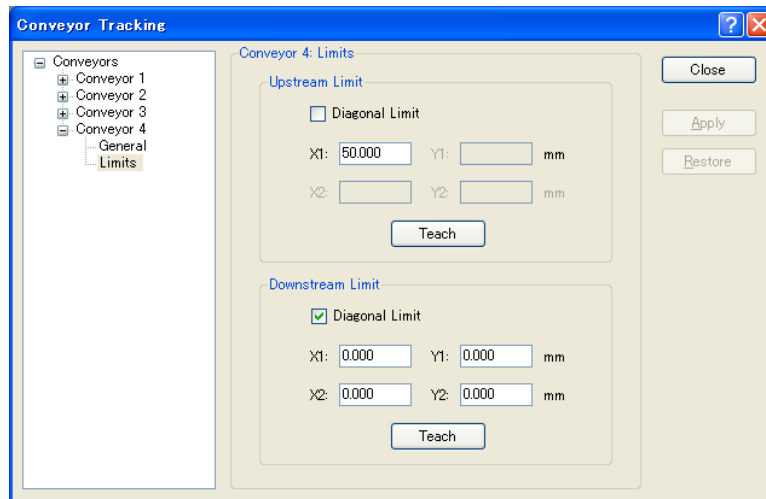


10. The Jog & Teach dialog appears. Click the jog buttons to move the robot end effector to the pick position. Click the **Teach** button.



To set the diagonal downstream limit, check the <Diagonal Limit> check box in the [Downstream Limit] area and click **Apply**.

The following dialog appears. Click the **Teach** button and follow the directions in the wizard.



Note that error 4415 occurs when the diagonal upstream / downstream limits are defined as in the following cases.

- They are perpendicular to the part flow direction.
- They are parallel to the part flow direction.
- The diagonal upstream limit and downstream limit cross on the conveyor.

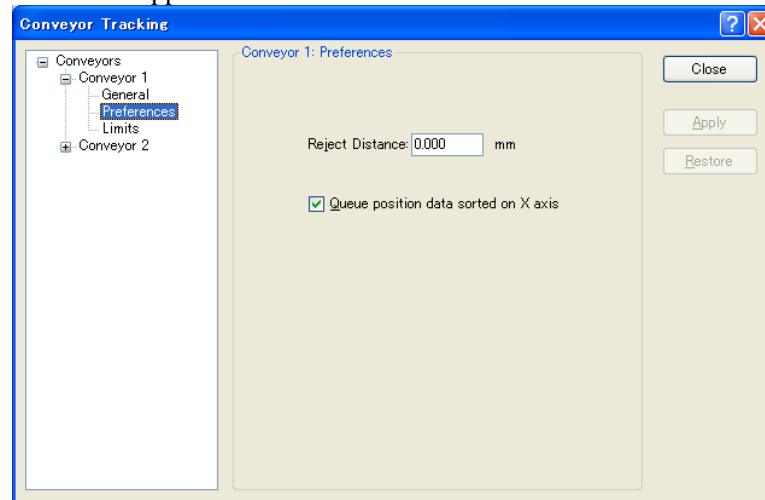
## 15.16 Queue Sorting

When you set the queue sorting, it registers the queue data in the order of position along the X axis in the conveyor local coordinate system.

Set 0 for the index number of Cnv\_QueGet command. If you set nothing, the robot picks up parts from the downstream side.

### To set the queue sorting

1. Select Tools | Conveyor Tracking.
2. Click the conveyor you want to configure and select the [Preferences]. The dialog below will appear.



3. Set the [Queue position data sorted on X axis] box.
4. Click the **Apply** button.



When you set a diagonal upstream limit, register the queue data in the order of entering the Pickup Area.

Also, when you set a diagonal upstream limit, note that the queue sorting cannot be canceled.

## 15.17 Abort Tracking

There are some situations when you want to abort tracking a part that moves out of the Pickup Area while the robot is tracking the part. In this case, use the Cnv\_AbortTrack command in a separate task that monitors the conveyor queue.

```
Function MonitorDownstream
  Robot 1
  Do
    If Cnv_QueLen(1, CNV_QUELEN_DOWNSTREAM) > 0 Then
      Cnv_AbortTrack 0
    EndIf
    Wait .1
  Loop
Fend
```

## 15.18 Conveyor Tracking with 6-Axis Robot

When you use a 6-axis robot in a conveyor tracking system, you need to set the values of U, V, and W. For this, use the Cnv\_QueGet command.

The following shows the case where the robot end effector moves toward a part during the pickup.

```
Go Cnv_Queget (Conveyor number , [ Index ] ) : U ( 90 ) : V ( 0 ) : W ( 180 )
```

## 15.19 Tracking Mode

There are two tracking modes: picking quantity-priority mode and picking accuracy-priority mode. The mode can be selected by Cnv\_Mode command.

Tracking mode selection is only available for linear conveyors. For circular conveyors, the picking quantity-priority mode is only available.

### Picking quantity-priority mode

Picking quantity-priority mode prioritizes reducing time to catch up with the work piece (queue) over the picking accuracy. This mode is suitable for the conveyor tracking system in which space between the work pieces is narrow.



When the picking quantity-priority mode is selected, tracking delay (situation in which the Manipulator does the picking motion at the posterior part of the work piece to the direction of the conveyor motion) may occur. If the tracking delay occurs, write the program as follows.

```
Go Cnv_Queget (Conveyor number , [ Index ] ) + X ( ** )
```

### Picking accuracy-priority mode

Picking accuracy-priority mode improves the picking accuracy while it takes more time to catch up with the work piece. This mode is suitable for the conveyor tracking system for small work pieces.

Picking accuracy-priority mode should be used for the conveyors of 350mm/sec or less.



When the conveyor of 350mm/sec or more is used, the tracking mode will be picking quantity-priority mode regardless of the setting of Cnv\_Mode.



Although the tracking delay does not occur in the picking accuracy-priority mode, the Manipulator may slide to the direction of the conveyor motion in Go, Move, or Jump3 motions after the downward motion of the Z-axis. If this occurs, take following countermeasures (these may not be effective for Go and Move motions)

- For Go motion: Change to Jump motion. Or, reduce the values of Accel and Speed.
- For Move and Jump3 motions: Reduce the values of AccelS and SpeedS.

### 15.20 Manipulator Posture

Manipulator posture during the tracking motion is always the default posture regardless of the posture at the conveyor tracking calibration. To specify the posture for the tracking, write a program as follows.

Example: tracks the work piece with Lefty arm position

```
jump Cnv_Queueget(Conveyor number,[ndex])/L
```

**NOTE**



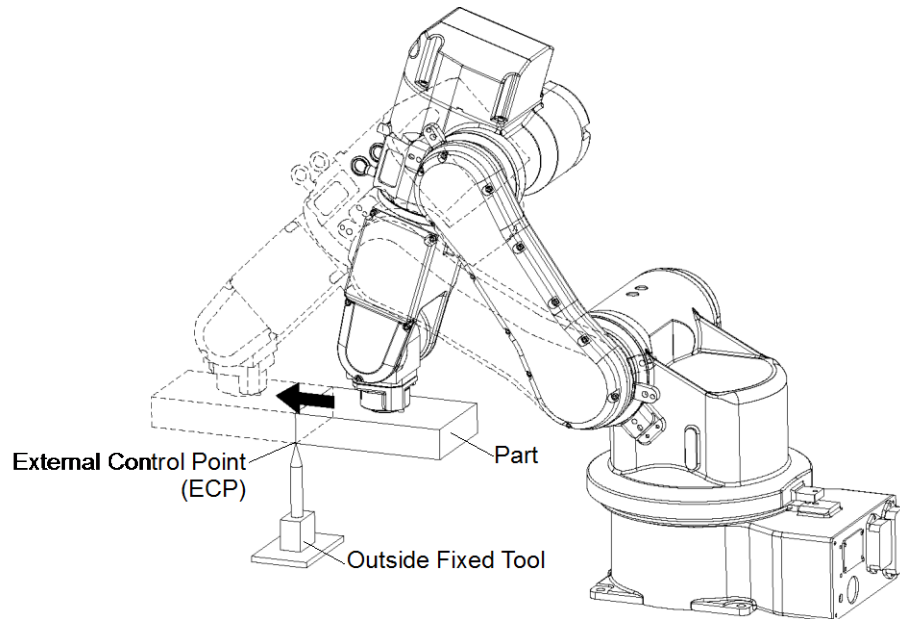
During the tracking motion, singularity avoiding function cannot be used. Therefore, set the positions of the Manipulator and the conveyor so that the Manipulator does not pass through the singularity.



## 16. ECP Motion

### 16.1 Overview

An ECP (external control point) motion is when the robot arm holding a part follows a specified trajectory (part's edges, etc.) using an outside fixed tool.



The ECP option supports the following:

- ECP definition by ECPSet statement and selection by ECP statement
- ECP motion commands (additional functions of Move, Arc3, Curve, and CVMove commands)
- Teaching with ECP jogging

This option is available for SCARA (including RS series), Cartesian and 6-axis robots. Also, it can be used with multi-robot systems.

Up to 15 ECP coordinate systems can be defined.

### 16.1.1 How to move the arm with ECP motion

In the following paragraphs, the process for moving the 6-axis robot arm with ECP motion is explained as an example.

#### 1. Setting an ECP

The ECP (external control point) is a coordinate system data used for defining the robot position and orientation at a processing point on the tip of the outside fixed tool.

The ECP should be defined based on the robot coordinate system or desired local coordinate system.

For example, when a drawing shows that the ECP is located at X=300, Y=300, Z=300 based on the robot coordinate system, specify it as shown below.

```
ECPSet 1,XY(300,300,300,0,0,0) ' Defines ECP No.1
```

When you have no ECP location data, you can specify it by teaching.

As an example, attach the tool of which you know the data precisely and bring the tip of the tool close to the ECP and then teach its position anywhere as P0. Then, specify the ECP using P0 coordinate data as shown below.

```
ECPSet 1,P0 :U(0) :V(0) :W(0) ' Defines ECP No.1
```

The orientation data (U, V, W) were set to 0 in the above examples. In these cases, the orientation in the ECP coordinate system is equal to that in the reference robot coordinate system.

You can specify U, V, and W coordinates in the ECP coordinate system. However, this data is valid only during the tangential correction mode ON in the Curve statement and ECP jog motion.

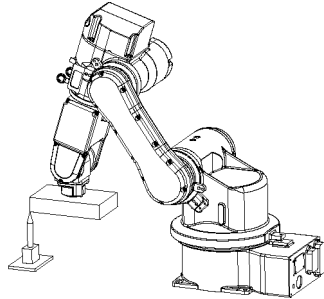
## 2. Teaching

Teach the point data while moving the robot arm holding the actual part. In this section, the part is assumed to be a rectangular solid and the arm is moved straight so that it touches one side of the part of the ECP specified in the previous section 1. *Setting the ECP*.

For details on teaching, refer to 5.11.1 *Robot Manager (Tools Menu)*.

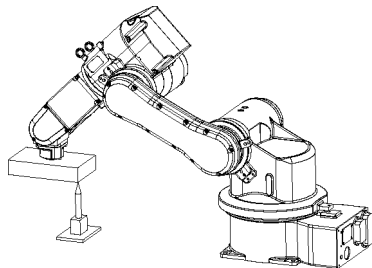
### 2-1 Teaching the motion start point

Move the arm to the motion start point and teach it as P1.



### 2-2 Teaching the motion end point

Move the arm to the motion end point and teach it as P2.



#### NOTE ECP Jog Mode:

The ECP jog mode is an additional jog mode used for teaching besides the Joint, World, and Tool jog modes.

The ECP jog mode is based on the selected ECP coordinate system.

## 3. Executing Motion

To move the arm with ECP motion, add the “ECP” parameter to a motion command.

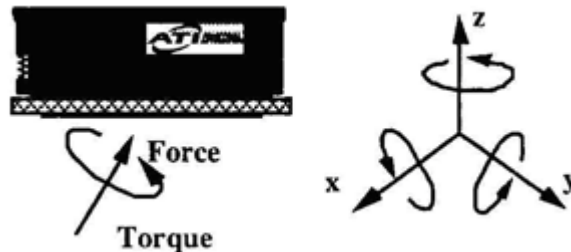
ECP 1	'	Select ECP
Go P1	'	Moves the arm to the motion start point
Move P2 ECP	'	Executes ECP motion

Use the Arc3 command to move the arm in an arc trajectory with the fixed tool. Use the Curve and CVMove commands to move the arm in cubic spline curves.

## 17. Force Sensing

### 17.1 Overview

The EPSON RC+ Force Sensing Option allows you to integrate force sensing in your applications. The force sensor is typically mounted on the robot's U axis. The sensor has 6 axes: ForceX, ForceY, ForceZ, TorqueX, TorqueY, TorqueZ.



With this option you can do the following:

- Read one or all 6 force/torque sensing axes values.
- Set triggers for motion commands.
- Use multiple force sensors in the same application.



*G Series robot with Gamma force sensor*

### 17.2 Specifications

EPSON RC+ supports ATI force sensors using PCI interface boards.

For the PCI interface board, we support the following products of National Instruments.

PCI-6220	Connect one force sensor
PCI-6224	Connect one or two force sensors
PCI-6034E	Connect one force sensor (Conventional)

Note that we offer only the software license of this option. If you need ATI Force Sensor, a set of the PCI interface board and the sensor, please purchase it separately.

When ordering force sensors and interface boards from ATI, be sure to mention that the equipment is being used with an Epson robot and also mention the controller model.

For specifications on force sensors, please see the ATI website:

<http://www.ati-ia.com/products/ft/sensors.aspx>

## 17.3 Installation

The Force Sensing Option must be enabled in the RC620 controller. If you purchased the option with your system, then the option will already be installed and configured.

You can also purchase the Force Sensing Option and install it in the field. See the chapter *Installing EPSON RC+ Options* for details.

### Installing the force transducer circuit board

If you are adding Force Sensing in the field, you must install the force transducer board(s) in your system, then run the NI-DAQmx driver installer.

NOTE



When ordering force transducers from ATI, you need to mention that they will be used in the EPSON RC620 controller. You can specify a board that connects with one force transducer, or a board that connects with two force transducers.

### Board Installation

Before installing the force sensing circuit board(s), you must first install the National Instruments DAQmx drivers that came with the board, then install the board. To install the National Instruments DAQ drivers:

1. Run the NI-DAQmx driver installer.
2. Accept defaults for each step of the installation wizard.
3. Shutdown the system.
4. Install the board(s).
5. Start the system.
6. Run the National Instruments Measurement & Automation Explorer program once to verify that the board(s) that were installed are recognized.

You do not need to install the ATI software that came with the force transducer.

NOTE



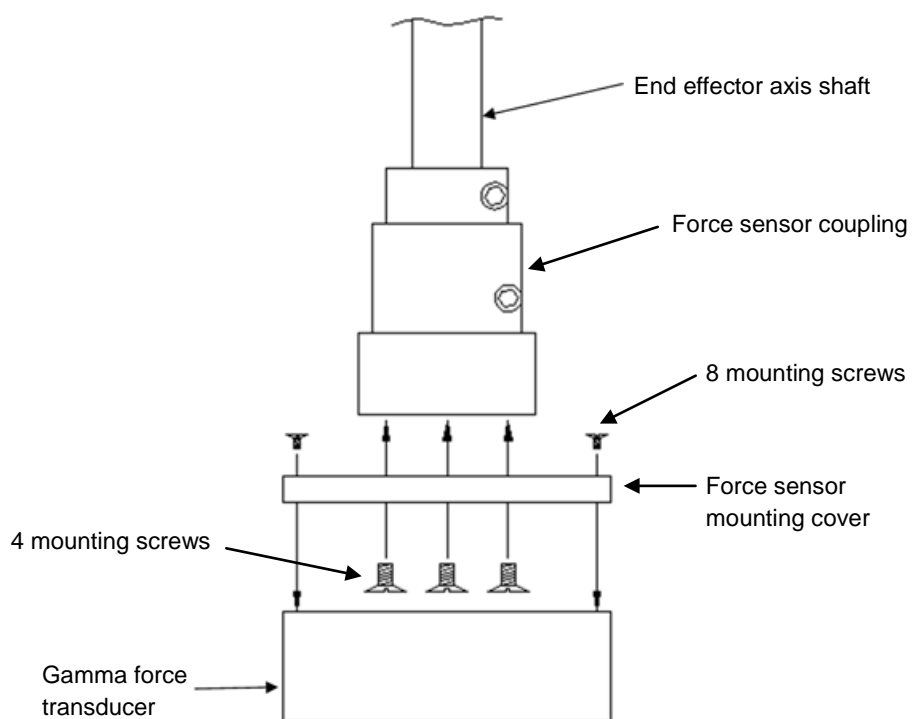
The calibration for the transducer must be loaded into memory. EPSON RC+ 6.0 handles this when you import the calibration data file as described in the section *Software Configuration* later in this chapter. The calibration data file can be located on the CD that came with the force transducer.

### Mounting the force transducer

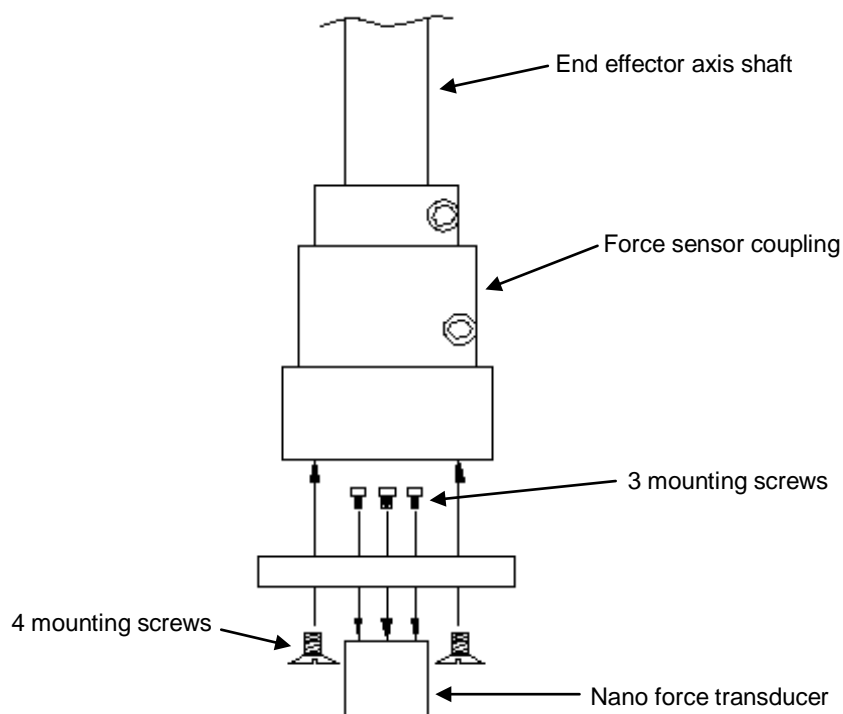
To mount the force transducer to the robot:

1. Remove the top cover of the transducer.
2. Remove the robot's end effector axis coupling and mount it to the transducer cover.
3. Install the transducer cover / coupling assembly onto the transducer.
4. Install the entire assembly on the end effector axis.

The following figures show mounting for gamma and nano transducers.



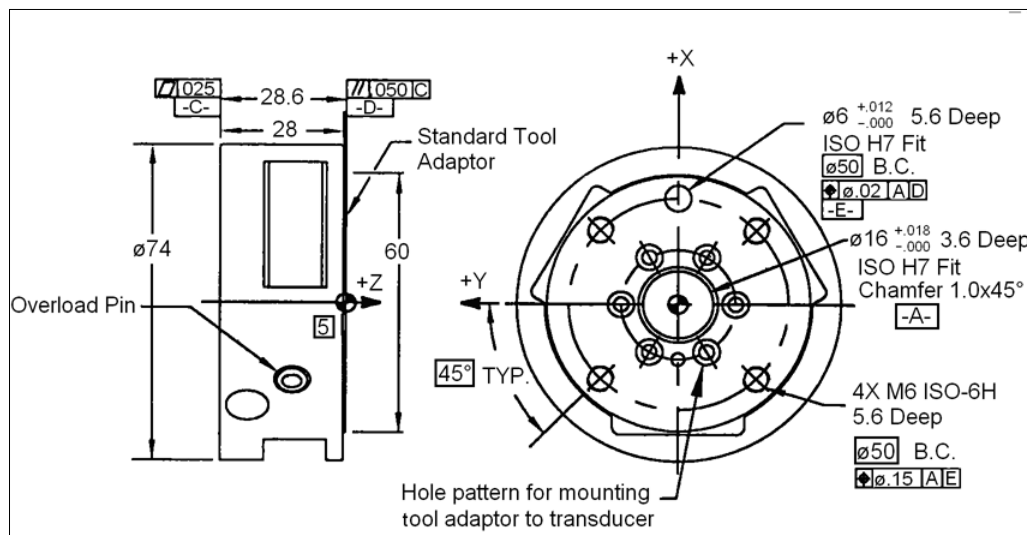
*Mounting the Gamma Force Sensor*



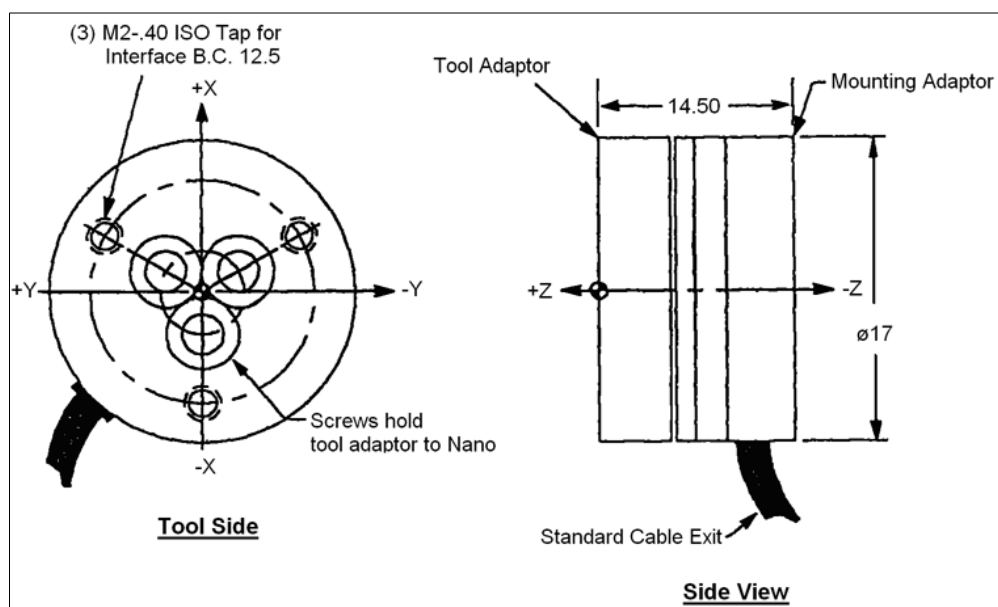
*Mounting the Nano Force Sensor*

### Mounting tooling to force sensor

The following diagrams show the tooling mounting dimensions for Gamma and Nano force transducers.



*Tool mounting for Gamma transducer*



*Tool mounting for Nano transducer*

### Connecting the force transducer

Use the cable supplied with the transducer to connect it to the PC board. The Nano transducer connects to an external interface box which in turn connects to the PC board.

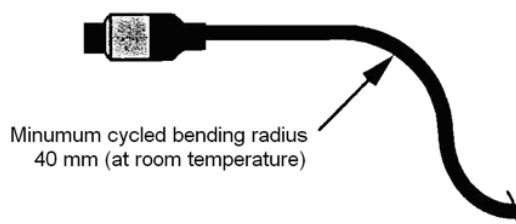


- Make sure power is off before connecting or disconnecting the force transducer. Protect transducer from electro-static discharge. Do not touch the internal electronics or connector pins.

### Routing the transducer cable

The transducer cable must be routed so that it is not stressed, pulled, kinked, cut, or otherwise damaged throughout the full range of motion. If the cable will rub other cables during cycling, use a plastic spiral wrap to protect it.

When the cable is cycling below the minimum bending radius the cable may fail due to fatigue. A small radius can be used if the cable is not being moved.

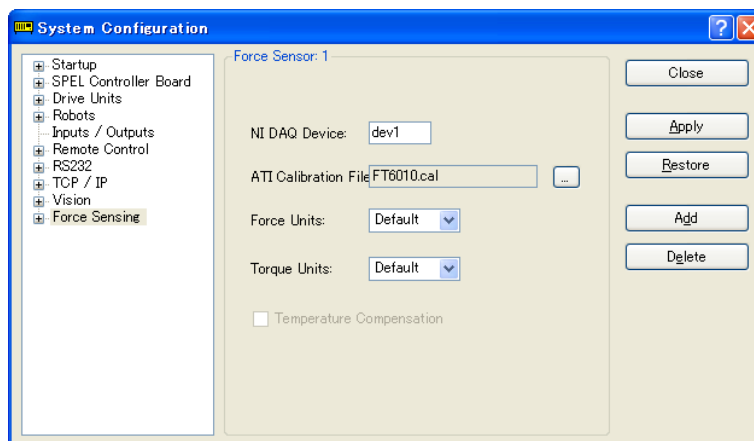




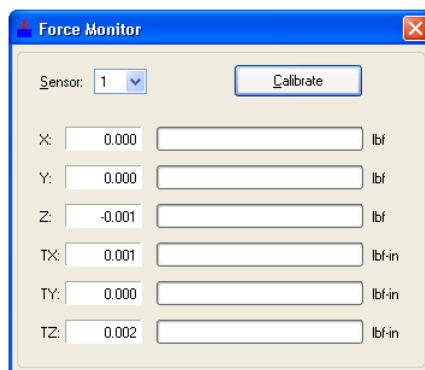
## Software Configuration

To configure EPSON RC+ 6.0 Force Sensing:

1. Start EPSON RC+ 6.0, then select Setup | System Configuration.
2. Click on the Force Sensing item in the tree on the left. If Force Sensing item is not visible, then the software options key for Force Sensing has not been enabled.
3. To add a board, click the Add button. A new force sensor will be shown in the tree on the left, and the controls used to configure the sensor will become enabled.



4. Enter the NI DAQ device name. This is assigned by the National Instruments software. To view NI DAQ device numbers, run the National Instruments Measurement & Automation Explorer.
5. Click the button shown on the right of ATI Calibration File to import a calibration file for the sensor. This can be found on the CD that came with the sensor. Navigate to the calibration file whose name includes the serial number of the sensor. Click Open, and the file will be copied to the EpsonRC60\force directory.
6. Leave the force and torque units at the default setting to use the native units. The actual units will be displayed in the sensor list after you click Apply. Or, you can select the desired units.
7. Click Apply to accept the new sensor.
8. From the Tools menu, select Force Monitor. This will open the Force Monitor window.



9. Apply pressure to the sensor. You should see the values change on the Force Monitor window. If you are using multiple sensors, change the sensor number on the monitor and verify that each sensor is working.

### 17.4 Force Sensing Commands

All Force Sensing commands begin with the same prefix: "Force\_". Here is a list of all of the commands. For details, please see the online help or SPEL<sup>+</sup> Language Reference Manual.

<b>Force_Calibrate</b>	Zeros out all axes for the current sensor.
<b>Force_ClearTrigger</b>	Clears all trigger conditions for the current sensor.
<b>Force_GetForce</b>	Returns the current value for one axis for the current sensor.
<b>Force_GetForces</b>	Returns the current values for all axes for the current sensor in an array.
<b>Force_Sensor</b>	Sets / returns the current sensor for the current task.
<b>Force_SetTrigger</b>	Sets /displays the force limit triggers for the current sensor.

## 17.5 Using the Force Sensing Trigger

You can configure the system to stop the robot after the force sensing trigger has been activated. You can set the trigger to activate when one or more force sensing axes reaches a preset limit. You use the Till command to check the trigger condition during motion.

### Stopping Motion along Z axis

Use a trigger on the ZForce axis to stop the robot during Z axis motion.

For example:

' Set the force trigger to fire when force on Z axis is less than -10

```
Force_ClearTrigger
Force_SetTrigger FORCE_ZFORCE, -10, FORCE_LESS
Till Force
Jump P1
Speeds 1
Move P2 Till
```

You can combine other conditions with Force in the Till command:

```
Till Sw(1) = On Or Force
```

You can combine other force/torque conditions by calling Force\_SetTrigger more than once. It's a good idea to clear all triggers first before setting them.

```
Force_ClearTrigger
Force_SetTrigger FORCE_ZFORCE, -10, FORCE_LESS
Force_SetTrigger FORCE_XFORCE, 5, FORCE_GREATER
```

### Stopping Motion along X or Y axes

Use a trigger on the XForce, XTorque, YForce, YTorque axes to stop the robot during Z axis motion. You need to align the force sensor by rotating the robot's U axis. The X and Y axes of the force sensor are marked on the transducer.

For example:

' Set the force trigger to fire when torque or force on X axis is less than -10

```
Force_ClearTrigger
Force_SetTrigger FORCE_XFORCE, -10, FORCE_LESS
Force_SetTrigger FORCE_XTORQUE, -10, FORCE_LESS
Till Force
Jump P1
Speeds 1
Move P2 Till
```

## 18 Real-Time I/O

This function does not work with EPSON RC+ 6.0 Ver.6.2.0.

You can use this function with the controller (Serial number 01 / 02-02001 or later).

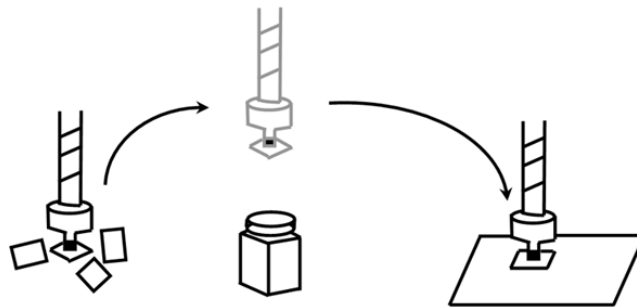
In the case of 6-axis robot, this function can not be used.

### 18.1 Overview

Real-time I/O is a feature that allows you to input trigger signals into the R-I/O connector of the robot controller so that you can latch and acquire the robot position at high speed while it is operating.

An example of an application using real-time I/O is "Picture on the fly": This synchronizes the robot position detection and the vision position detection, and performs part pickup, alignment, and assembly without stopping the robot.

With the real-time I/O feature, you can reduce the robot stop time for vision image acquisition that is necessary for traditional vision applications.



### 18.2 Specifications

#### R-I/O Connector

The RC620 robot controller has an R-I/O connector that is used to connect the real-time I/O trigger input signals. An R-I/O input is a special input interface that monitors the signals at higher speed than the standard I/O inputs. There are two trigger input signals on each of the RC620 Control Unit and external Drive Unit. For example, set the transmission type sensor so that it reacts when the robot passes the camera acquisition point and use the R-I/O connector so that the R-I/O input is detected at the moment shutter clicked.

For the details of the hardware (connection connector, connection circuit), refer to the manual *RC620 Robot Controller - R-I/O*.

#### Real-time I/O commands

There are special commands provided for use the real-time I/O. The following are basic descriptions of these commands.

For more details, refer to the manual *SPEL+ Language Reference*.

#### LatchEnable

This command is used to enable or disable the latch function of the robot position information with the real-time I/O. When LatchEnable On executes, it enables the robot position latch function using the trigger input signals connected to the R-I/O connector. After the latch is enabled, the robot current position information is latched when the first trigger input is detected. To repeatedly latch the robot position, execute LatchEnable Off and then execute LatchEnable On again. To use the

command repeatedly, it requires 60 ms minimum interval for each command processing time but it is not necessary to consider the command executing time.

### SetLatch

Specifies the real-time input port that you connected the trigger input signal to, and the input logic. The table below shows the port numbers you can specify. Specify the port number that the robot using R-I/O is connected. If the other ports are specified, an error will occur. One robot cannot wait for the trigger signals from multiple ports.

		Point	Port number
Control Unit	INPUT	2 points	24,25
Drive Unit 1	INPUT	2 points	56,57
Drive Unit 2	INPUT	2 points	280,281

Executing SetLatch needs approximately 40 msec for processing.

### LatchState Function

This function returns the position latch status.

After it confirms that the latch has been done, it acquires the position information using the LatchPos Function.

### LatchPos Function

This function returns the robot position information latched by the trigger input.

Executing the LatchPos Function needs approximately 15 msec for processing.

### RobotPos Vision Sequence Property

Set the RobotPos sequence property to set the robot coordinates of the image acquisition position to calculate the part position when you use a mobile camera system. The system can calculate the correct part position by using the position acquired by LatchPos Function in this property.

For the details, refer to the manual *Vision Guide 6.0 Properties & Results Reference*.

### Latch accuracy

The following is the theoretical sampling time used to latch the position information.

		Sampling time [μsec]
Control Unit	4-axis robot	32
	6-axis robot	32
Drive Unit	4-axis robot	32
	6-axis robot	21

You can get a rough idea of latch accuracy from the robot speed (parts moving speed) at the latch trigger input and the sampling time. For real accuracy, you must have a margin on the required accuracy because time delay and variation in the hardware may affect. The latch accuracy will improve as the robot moves slower at the trigger input.

$$\text{Latched position accuracy [mm]} = \text{Robot speed [mm/sec]} \times \text{Sampling time [sec]}$$

## 18.3 Usage

### 1. Basic example

The following program is a sample to connect any trigger signal to the R-I/O connector of the controller, latch the robot position information while it is operating at the trigger input, and show the latched position information.

```

Function Main
  Motor On
  Power High

  Speed 50; Accel 50, 50
  SpeedS 500; AccelS 5000

  Go P0                                ' Start point
  SetLatch SETLATCH_PORT_CU_0, SETLATCH_TRIGGERMODE_LEADINGEDGE
  LatchEnable On                        ' Enable the latch
  Move P1                               ' Start the operation, trigger input while operating

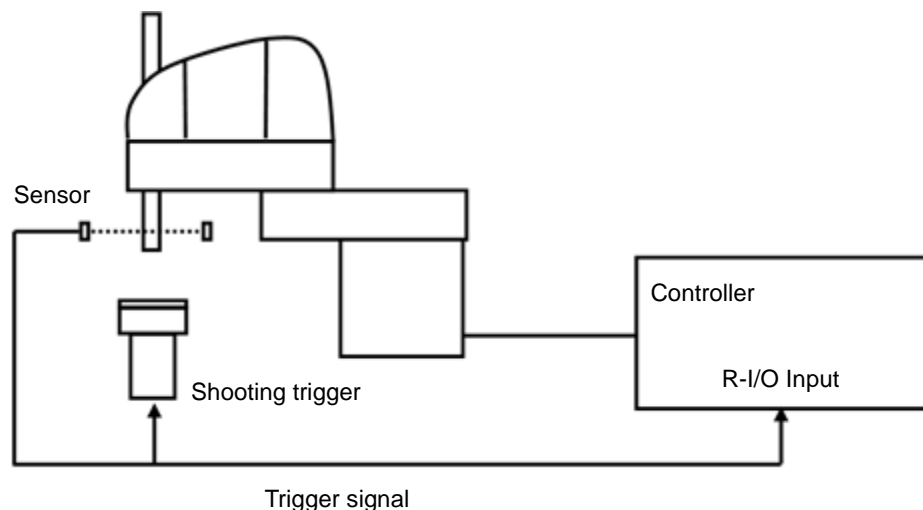
  Wait LatchState = True                ' Confirmed the latch is complete
  P3 = LatchPos                         ' Acquire the latched position
  LatchEnable Off                       ' Disable the latch

  Print P3                             ' Show the latched position
End

```

### 2. Example with Vision system

This is an example that uses the robot end effector to handle parts, passes above the external fixed upward camera acquisition point without stopping, and assembles the parts with an appropriate position correction.

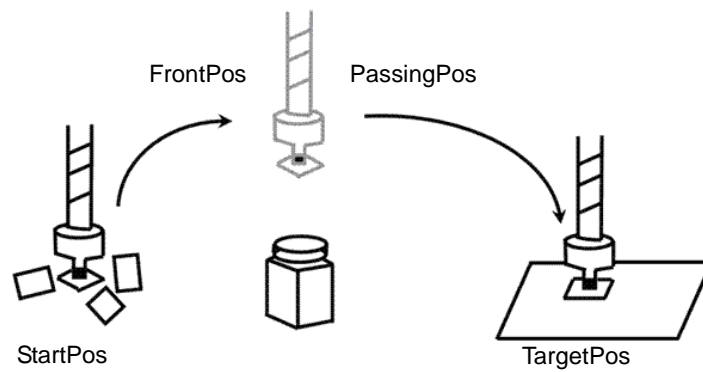


This system has a transmission type sensor that outputs the trigger signal when the robot end effector handles a part and passes the camera acquisition point. Then, it connects the sensor output with both the R-I/O and the camera trigger input for external tuning and synchronizes the latched robot position information and the camera image. It calculates the part position error and offsets the position comparing the robot position information from the camera image to the robot position information from the real-time I/O.

In this case, the robot vision system must be calibrated.

For details on the camera trigger signal connection and the vision calibration, refer to the manual *Vision Guide 6.0*.

The following program is a sample.



```

' Variable that stores the position correction amount
Global Double g_RevPosX_i, g_RevPosY_i, g_RevPosU_i

Function Main
  Robot 1
  Motor On
  Power High

  Speed 100
  Accel 100, 100

  Jump InitPos                                     ' Move to the initial position
  Wait 1.0

  SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
                                                    ' Set the latch condition

  MemOff 0
  Xqt PictureOnFly_Camera                         ' Start the shooting task

  Jump StartPos C0
  Wait 0.5                                         ' Move to the part feed point

  LatchEnable On                                  ' Start waiting the latch

  MemOn 0                                          ' Enable the shooting

  Jump FrontPos C0 CP                             ' Move close to the camera
  Go PassingPos CP                               ' Pass over the camera

  Go TargetPos :Z(-70) CP                         ' Move over the assembly point

  Wait MemSw(1) = On                             ' Wait until the image processing is complete
  Jump TargetPos +X(g_RevPosX_i) +Y(g_RevPosY_i) +U(g_RevPosU_i) C0 LimZ(-70)
                                                    ' Move to the assembly point

  Wait 0.5

  Jump InitPos                                    ' Move to the initial point
  Wait 0.5

  Motor Off

Fend

```

```

' Acquire the part image
Function PictureOnFly_Camera

    ' Vision Result variable
    Double WorkX, WorkY, WorkU, Angle          ' Part position and angle
    Integer AcqStat                            ' Strobe shooting completion flag
    Boolean Found                              ' Detected condition

    ' Variable for calculating the position correction
    Double Xtmp, Ytmp, Xofs, Yofs
    Double Xrot, Yrot
    Double dAngle, rtmp, rtmp2

    Wait MemSw(0) = On
    MemOff 1                                    ' Clear the acquire completion flag
    MemOff 0                                    ' Clear the acquire start flag
    AcqStat = 0                                ' Clear the strobe acquire flag

    VRun PictureOnFly_i

    Wait LatchState = True                      ' Wait for the trigger
    ShutterPos = LatchPos                      ' Store the trigger position

    Do Until AcqStat = 3                        ' Wait a strobe
        VGet PictureOnFly_i.AcquireState, AcqStat
        Wait 0.01
    Loop

    ' Set the sequence robot position
    VSet PictureOnFly_i, ShutterPos

    ' Process the acquired image
    VGet PictureOnFly_i.Geom01.RobotXYU, Found, WorkX, WorkY, WorkU
    VGet PictureOnFly_i.Geom01.Angle, Angle

    ' Calculate the part offset
    Xtmp = WorkX - CX(ShutterPos)
    Ytmp = WorkY - CY(ShutterPos)
    rtmp = Sqr((Xtmp ** 2) + (Ytmp ** 2))

    ' Calculate the part rotation
    Xrot = rtmp * Cos(DegToRad(Angle))
    Yrot = rtmp * Sin(DegToRad(Angle))

    ' Calculate the offset
    dAngle = CU(StopPos) - CU(ShutterPos)
    Xofs = Xrot * Cos(DegToRad(dAngle)) + Yrot * Sin(DegToRad(dAngle))
    Yofs = Xrot * -Sin(DegToRad(dAngle)) + Yrot * Cos(DegToRad(dAngle))

    ' Calculate the position offset
    g_RevPosX_i = Xrot - Xofs
    g_RevPosY_i = Yrot - Yofs
    g_RevPosU_i = dAngle - Angle
    If(g_RevPosU_i > 180) Then
        g_RevPosU_i = g_RevPosU_i - 360
    ElseIf(g_RevPosU_i < -180) Then
        g_RevPosU_i = g_RevPosU_i + 360
    EndIf

    MemOn 1

Fend

```



## 19 Additional Axis

### 19.1 Overview

You can attach up to two additional drive axes (per manipulator) which can operate in association with the manipulator. The position data of the additional axis is saved with the robot point data. The additional axis can move simultaneously with the manipulator by motion commands and you can design an application using a traveling axis (manipulator on the straight axis) with simple programming.



If you want to operate the manipulator and drive axis separately, you need to define the additional axis as another manipulator using the multi-manipulators feature.



■ When you use the additional axis as traveling axis and mount a manipulator(s) on the axis, the reaction force of manipulator(s) is put on the traveling axis. Therefore, you should limit the acceleration/deceleration speed with the Accel setting so that it is within the allowable inertia of traveling axis. In addition, the manipulator may swing widely at the positioning and possibly break the additional axis.

### 19.2 Specification

#### Types of additional axis

The supported additional axes are the servo axis (X5 series Single-axis robot) controlled by the servo driver of the controller and the PG axis, controlled by the pulse generator board. However, note that the PG axis has some limitations.

##### Limitations of a PG additional axis

- a. Synchronizes with the manipulator to start motion but not to finish.
- b. Does not support Path motion with CP On and Pass. Stops for every motion.
- c. Does not go through the CvMove series of points
- d. Calibration is necessary using the MCAL command. Cannot operate robots until the calibration is complete.

#### Number of additional axis

Up to two additional axes are available for each of the SCARA robot series (including RS series), Cartesian coordinate robot, 6-axis robot, and JOINT type robot. However, the number of axes you can add is determined by how many axes are available with your controller.

#### Position data management

The additional axes are allocated to Joint #8 and #9 for all robot types. The position data are shown in the S and T coordinate values of point data of the manipulator to which you add the additional axes.

The additional axis as Joint #8 is called the additional S axis and Joint #9 is the additional T axis.

The coordinate values of additional axes are saved with the robot point data but don't have any effect on the robot coordinate system.

**How to operate**

The additional axis can move simultaneously with the manipulator (synchronous start / stop). However, if you use the PG axis, it doesn't synchronize with manipulator to finish and operate by the different acceleration/deceleration speed from the manipulator. See below for the details of motion commands.

Also, you can operate the additional axis and manipulator separately by proper management of the point data. However, you cannot operate separately both of them in arbitrary timing. In this case, use the multi-manipulators function and set the drive axis as another manipulator.

**Commands' specification**

Pulse, Go, BGo, TGo, Pass

The additional axis can operate in association with the manipulator motion. However, if you use the PG axis, it synchronizes only to start the motion and a motion command completes when both the manipulator and axis finish each motion. In addition, if the PG additional axis has a travel distance, the Path motion with CP On and Pass are prohibited and the axis moves with CP Off automatically.

Move, BMove, Tmove

The additional axis can operate in association with the manipulator motion. However, if you use the PG axis, it synchronizes only to start the motion and a motion command completes when both the manipulator and axis finish each motion. In addition, if the PG additional axis has a travel distance, the Path motion with CP On is prohibited and the axis moves with CP Off automatically.

Arc, Arc3

The additional axis can operate in association with the manipulator motion. It doesn't go through the specified midPoint and directly goes to the end point. If you use the PG axis, it synchronizes only to start the motion and a motion command completes when both the manipulator and axis finish each motion. In addition, if the PG additional axis has a travel distance, the Path motion with CP On is prohibited and the axis moves with CP Off automatically.

CVMove

The additional axis can operate in association with the manipulator motion. If you use a servo axis for the additional axis, for each of the S and T axis it creates a curve going through the S and T coordinates specified by a series of point data. However, if you use the PG axis for the additional axis, it doesn't go through the series of points and directly goes to the end point. Also, it synchronizes only to start the motion and a motion command completes when both the manipulator and axis finish each motion. In addition, if the PG additional axis has a travel distance, the Path motion with CP On is prohibited and the axis moves with CP Off automatically.

Jump

The additional axis executes PTP motion in association with the manipulator horizontal motion. However, if you use the PG axis, it synchronizes only to start the motion and a motion command completes when both the manipulator and axis finish each motion. In addition, if the PG additional axis has a travel distance, the Path motion with CP On is prohibited and the axis moves with CP Off automatically.

Jump3, Jump3CP

The additional axis can operate in association with the manipulator depart / span / approach motion. However, if you use the PG axis, it synchronizes only to start the motion and a motion command completes when both the manipulator and axis finish each motion. In addition, if the PG additional axis has a travel distance, the Path motion with CP On and Pass are prohibited and the axis moves with CP Off automatically.

JTran, PTran

The additional axis can operate separately by specifying as Joint #8, #9.

Example:

```
> JTran 8, 90      'Move the additional S axis by 90 mm
> PTran 9, 10000   'Move the additional T axis by 10000 pulse
```

## 19.3 Usage

### Additional axis configuration

For the instruction of configuring the additional axis, refer to 9.2 *Configuration of Additional Axes*.

If you use the PG axis for the additional axis, you need to set the PG parameters. For the details of PG parameters, refer to the *Robot Controller RC620 option: PG Motion System manual*.

### Point data usage

This example specifies the position data of manipulator and additional ST axes and substitutes them to the point data.

```
P1 = XY(10, 20, 30, 40) :ST(10, 20)      ' SCARA robot
P1 = XY(10, 20, 30, 40, 50, 60) :ST(10, 20) ' 6-axis robot
```

This example specifies the position data of manipulator and additional ST axes and executes a PTP motion.

```
Go XY(10, 20, 30, 40) :ST(10, 20)
Go XY(10, 20, 30, 40, 50, 60) :ST(10, 20)
```

This example specifies the position data of additional ST axes individually

```
P1 = XY(10, 20, 30, 40) :S(10) :T(20)
P1 = XY(10, 20, 30, 40) :S(10)
P1 = XY(10, 20, 30, 40) :T(20)
```

This example omits the robot position assignment XY() and specifies only the additional axis position. Then, the point data is defined so that the manipulator doesn't move (undefined).

```
P1 = ST(10, 20)
Go P1      'Only additional axis moves and
           the manipulator remains at the current position.
```

This example operates only the additional axis.

```
Go ST(10, 20)      'Only the additional axis moves.
```

This example omits the additional axis position assignment ST() and specifies only the manipulator position. Then, the point data is defined so that the additional axis doesn't move (undefined).

```
P1 = XY(10, 20, 30, 40)
Go P1      'Only the manipulator moves and
           the additional axis remains at the current position.
```

This example operates the manipulator only.

```
Go XY(10, 20, 30, 40)      'Only the manipulator moves.
```

This example calculates the additional axis coordinate value using a point operator expression.

```
P1 = XY(10, 20, 30, 40, 50, 60) :ST(10, 20)
P2 = P1 + S(10) + T(20)      'Add the offset amount to the additional ST axes for P1.
```

Note that you cannot use the point operator for undefined points.

```
P1 = XY(10, 20, 30, 40, 50, 60)
```

```
P2 = P1 + S(10) + T(20)
```

```
` Error (ST are undefined for P1 and you cannot use  
the point operator)
```

```
P1 = XY(10, 20, 30, 40, 50, 60) +ST(10, 20)    ` Error
```

```
P1 = XY(10, 20, 30, 40, 50, 60) +S(10) +T(20)  ` Error
```

```
Go ST(10, 20) + X(10)
```

```
` Error (XY are undefined and you cannot use the point operator)
```

This example shows the additional ST axes coordinate values retrieved from the point data.

```
Print CS(P1), CT(P1)
```

### **Pallet motion**

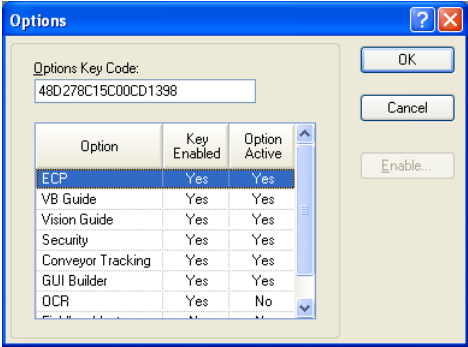
When you specify a pallet with the point data including the position data of additional axis, the position data of additional axis is also calculated by the pallet operator. If you use the additional axis as traveling axis, you can define a wide range pallet than for a single manipulator.

Also, if you want to use the additional axis not as traveling axis and exclude the additional axis position from the pallet operator, define the pallet with the point data that clears the additional axis position data.

## 20. Installing Controller Options

When you purchase options with your system, the options have already been installed on your system before shipment. Of course, you can purchase options separately.

To see what options are enabled on your system, select Setup | Options. The following dialog will be displayed.



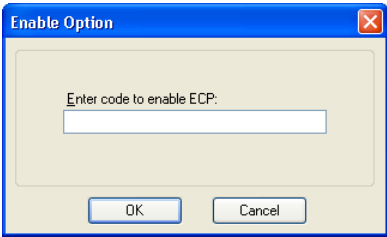
Item	Description
<b>Option</b>	Name of the option.
<b>Key Enabled</b>	Indicates that the option is enabled in the controller.
<b>Option Active</b>	Indicates that the option is currently active in RC+.

### To enable an option on site

1. Copy and paste or write down the Options Key Code. You can view this from the Setup | Options dialog.
2. Call your distributor to purchase the enable key code for the desired option.
3. You will receive a code to enable the option from your distributor.
4. Select the option to enable on the grid, and then click the Enable button.
5. Enter in the code you received from your distributor.



The key code is case sensitive.



*Enabling an option*

## 21. Software License Agreement

THIS IS A CONTRACT. CAREFULLY READ ALL THE TERMS AND CONDITIONS CONTAINED IN THIS AGREEMENT. INSTALLING THE SOFTWARE (EPSON RC+ 6.0) INDICATES YOUR ACCEPTANCE OF THESE TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO THESE TERMS AND CONDITIONS, SIMPLY DO NOT INSTALL OR USE THE EPSON RC+ 6.0 SOFTWARE.

### **LICENSE**

SEIKO EPSON CORPORATION (the "Licensor") hereby grants to you (the "Licensee") a non-exclusive and transferable right to use the EPSON RC+ 6.0 Software program and documentation (the "Licensed Materials"). You may only use this software on one (1) CPU. You may not use, copy, or modify the Licensed Materials, in whole or in part, except as expressly provided for by this agreement.

### **OWNERSHIP**

As the Licensee you own the magnetic or other physical media on which the Licensed Materials are recorded or fixed, but the Licensor retains sole and exclusive title to and ownership of the Licensed Materials recorded on the original disk and all subsequent copies regardless of the form or media in or on which the original and other copies may exist. By paying the fee required for this license, you do not become the owner of the Licensed Materials, but are entitled to use the Licensed Materials according to the terms of this agreement. You acknowledge that the Licensed Materials are not your property and understand that giving away or selling copies of the Licensed Materials is theft.

### **TERM**

This license is effective until terminated. You may terminate this license by destroying the Licensed Materials together with any backup copies that may have been made. This license will also terminate if you fail to comply with any term or conditions of this Agreement. You agree upon such termination to destroy the Licensed Materials together with any copies which may have been made.

### **BACKUP AND TRANSFER**

The Licensed Materials are copyrighted and contain proprietary information and trade secrets of the Licensor. Unauthorized copying, modifying or reproducing of the Licensed Materials, even if modified, merged or included with other software, is expressly forbidden. You may make one (1) archival copy of the Licensed Materials for the sole purpose of backing up the Licensed Materials. In no event does the limited copying or reproduction permitted hereunder include the right to decompile, disassemble or electronically transfer the Licensed Materials, or translate the Licensed Materials into another language. You may sell your license rights in the software to another party. If you sell your license rights in the Licensed Materials you must at the same time transfer the documentation and the backup copy or destroy the backup copy. You cannot sell your license rights in the Licensed Materials to another party unless that party also agrees to the terms and conditions of this agreement.

**PROTECTION AND SECURITY**

You agree not to deliver or otherwise make available the Licensed Materials or any part thereof, including, without limitation, the object code, to any person other than the Licensor or its employees, except for purposes specifically related to your use of the Licensed Materials on one (1) CPU, without the prior written consent of the Licensor. You agree to use your best efforts and take all reasonable steps to safeguard the Licensed Materials to ensure that no unauthorized person shall have access thereto and that no unauthorized copy, publication, disclosure or distribution thereof, in whole or in part, in any form, shall be made. You recognize that the Licensed Materials contain valuable confidential information and trade secrets.

**LIMITED WARRANTY**

The only warranty the Licensor makes to you in connection with this license to use the Licensed Materials is that the media on which the Licensed Materials are recorded will be replaced without charge, as long as the original diskette(s) are returned to the licensor, with satisfactory proof of date of purchase, within ninety (90) days of purchase. This warranty is limited to the Licensee and is not transferable. The foregoing warranty does not extend to any Licensed Materials that have been damaged as a result of accident, misuse or abuse.

EXCEPT FOR THE LIMITED WARRANTY DESCRIBED ABOVE, THESE LICENSED MATERIALS ARE PROVIDED "AS IS". THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE LICENSED MATERIALS IS ASSUMED BY YOU. THE LICENSOR DOES NOT WARRANT, GUARANTEE OR MAKE ANY REPRESENTATIONS REGARDING THE USE OF, OR THE RESULTS OBTAINED WITH THE LICENSED MATERIALS IN TERMS OF CORRECTNESS AND RELIABILITY OR LEGALITY. THE ABOVE IS THE ONLY WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. INCLUDING, WITHOUT LIMITATION, LOSS OF PROFITS OR INABILITY TO USE THE LICENSED MATERIALS, EVEN IF THE LICENSOR OR SUCH OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. YOU AGREE THAT YOUR EXCLUSIVE REMEDIES, AND THE LICENSOR'S OR SUCH OTHER PARTY'S ENTIRE LIABILITY WITH RESPECT TO THE LICENSED MATERIALS SHALL BE AS SET FORTH HEREIN, AND IN NO EVENT SHALL THE LICENSOR'S OR SUCH OTHER PARTY'S LIABILITY FOR ANY DAMAGES OR LOSS TO YOU OR ANY OTHER PARTY EXCEED THE LICENSED FEE PAID FOR THE LICENSED MATERIALS.

**ENTIRE AGREEMENT**

This Agreement represents the entire agreement between Licensor and Licensee with respect to the License and use of the Licensed Materials.

**SEVERABILITY**

If any provision or a portion of this agreement is determined to be invalid under any applicable law, it shall be deemed omitted and the remaining provisions and partial provisions of this agreement shall continue in full force and effect.

**NOTICE**

Any notice or other communication relating to this license agreement to be sent to the Licensor must be mailed by certified mail to your distributor.





## Appendix A: Automatic Processing of Project Import

### Project for EPSON RC+ 5.\*

When projects created in EPSON RC+ 5.\* are imported, all project files are copied to the new EPSON RC+ 6.0 project directory. In addition, the following processes are executed automatically:

- Point file update

#### Point File Update

For EPSON RC+ 5.\*, the .PTS files are updated automatically to the EPSON RC+ 6.0 .PTS file version.

### Project for EPSON RC+ 3.\* / 4.\*

When projects created in EPSON RC+ 3.\* / 4.\* are imported, the following processes are executed automatically:

- User program conversion
- Point file conversion
- I/O label file conversion
- User error label file conversion
- Vision Guide conversion

#### User Program Conversion

The tables below show the syntax conversions from EPSON RC+ 3.\* / 4.\* to EPSON RC+ 6.0.

Project Type	EPSON RC+ 4.*	EPSON RC+ 6.0
Syntax	While	Do While
	Wend	Loop
	Trap...Call	Trap...Xqt

Project Type	EPSON RC+ 3.*	EPSON RC+ 6.0
Syntax	While	Do While
	Wend	Loop
	Trap...Call	Trap...Xqt
	On \$, Off \$	MemOn, MemOff
	Sw(\$	MemSw(
	Sw \$(	MemSw(
	In(\$	MemIn(
	In \$(	MemIn(
	Out \$	MemOut
	Xqt !	Xqt
	Quit !	Quit
	Resume !	Resume
	Halt !	Halt

#### Point File Conversion

For EPSON RC+ 3.\*, the EPSON RC+ 6.0 .PTS files are generated automatically from the .PNT files and corresponding .DEF files.

Project Type	EPSON RC+ 3.*	EPSON RC+ 6.0
Point File	*.PNT file (Point file) *.DEF file (Point label file)	*.PTS

For EPSON RC+ 4.\*, the EPSON RC+ 6.0 .PTS files are generated automatically from the .PNT files.

Project Type	EPSON RC+ 4.*	EPSON RC+ 6.0
Point File	*.PNT file (Point file)	*.PTS

### I/O Label File Conversion

IOLabels.dat is generated automatically from the following three files.

Project Type	EPSON RC+ 3.* / 4.*	EPSON RC+ 6.0
I/O Label File	inplabel.txt outlabel.txt memlabel.txt	IOLabels.dat

### User Error Label File Conversion

Files are changed automatically as the user error numbers are changed.

Project Type	EPSON RC+ 3.* / 4.*	EPSON RC+ 6.0
User Error Label	30000 to 30999	8000 to 8999
User Error Label File	UserErrors.txt	UserErrors.dat

## Project Import for SPEL for Windows 2.\*

When projects created in SPEL for Windows 2.\* are imported, the following processes are executed automatically.

- User program conversion
- Point file conversion
- I/O label file conversion
- Global Preserve variable table conversion
- Global variable conversion
- Local variable conversion

### User Program Conversion

The table below shows the syntax conversions to EPSON RC+ 6.0.

Project Type	SPEL for Windows 2.*	EPSON RC+ 6.0
Syntax	While	Do While
	Wend	Loop
	Trap n...Call	Trap n...Xqt
	On \$, Off \$	MemOn, MemOff
	Sw(\$	MemSw(
	Sw \$(	MemSw(
	In(\$	MemIn(
	In \$(	MemIn(
	Out \$	MemOut
	Xqt !	Xqt
	Quit !	Quit
	Resume !	Resume
	Halt !	Halt
	Palet	Pallet
	Print"	Print "
	Date\$(0)	Date\$
	Time\$(0)	Time\$
	JS(0)	JS
	TW(0)	TW
	ZeroFlg(0)	ZeroFlg
	Entry	Global
	Config statement	SetCom statement
	Cooked	Line deleted
	SetRaw	Line deleted
	SelRB	Line deleted
	SelRB1	Line deleted
	Extern	Line deleted
	End	Quit All
	GetDate d\$	d\$ = Date\$
	GetTime t\$	t\$ = Time\$

### Point File Conversion

EPSON RC+ 6.0 .PTS files are generated automatically from the .PNT files and corresponding .DEF files.

Project Type	SPEL for Windows 2.*	EPSON RC+ 6.0
Point File	*.PNT file (Point file) *.DEF file (Point label)	*.PTS

### I/O Label File Conversion

Converts the I/O labels automatically.

Project Type	SPEL for Windows 2.*	EPSON RC+ 6.0
I/O Label File	<i>ProjectName</i> .IOL	IOLabels.dat

### Global Preserve Variable Table Conversion

Backup variable definitions created in the SPEL for Windows 2.\* Project Menu are converted into Global Preserve declaration statements in the first program file.

(Example)

If the SPEL for Windows 2.\* project defines an integer backup variable called “s\_iValue”, the following statement is generated in the first program of the project.

Global Preserve Integer s\_iValue

### Global Variable Conversion

Global variables (Entry / Extern) in SPEL for Windows 2.\* projects are converted to Global variables in EPSON RC+ 6.0.

Project Type	SPEL for Windows 2.*	EPSON RC+ 6.0
Global Variable (Command)	Entry / Extern command	Global command

### Local Variable Conversion

Local variables in SPEL for Windows 2.\* functions can be used throughout the entire file in which they are declared. These variables are converted to module variables or local variables in EPSON RC+ 6.0, depending on their scope.

If the variable is used in only one function, it is converted to a local variable in that function.

If the variable is used in more than one function, it is converted to a module variable.

## Appendix B: EPSON RC+ 6.0 Software

### EPSON RC+ 6.0 Software Installation

Make sure that the EPSON RC + 6.0 is installed by a user with Administrator right.

Insert the EPSON RC+ 6.0 setup DVD into the DVD drive and follow the menu to install the software.

### EPSON RC+ 6.0 Software Update

Make sure that the EPSON RC + 6.0 is updated by a user with Administrator right.

Insert the EPSON RC+ 6.0 setup DVD into the DVD drive and follow the menu to update the software. For the RC620 Controller without the DVD drive, connect the USB DVD drive and update the software.

