EPSON RC+ 7.0 (Ver.7.4)

*SPEL+ Language Reference*

Rev.9
FOREWORD

Thank you for purchasing our robot products. This manual contains the information necessary for the correct use of the EPSON RC+ 7.0 software. 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 contact the supplier of your region 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, Windows logo, Visual Basic, and Visual C++ are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Pentium is a trademark of Intel Corporation. Other brand and product names are trademarks or registered trademarks of the respective holders.

TRADEMARK NOTATION IN THIS MANUAL

Microsoft® Windows® 7 Operating system
Microsoft® Windows® 8 Operating system
Microsoft® Windows® 10 Operating system
Throughout this manual, Windows 7, Windows 8, and Windows 10 refer to above respective operating systems. In some cases, Windows refers generically to Windows 7, Windows 8, and Windows 10.

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.

MANUFACTURER

SEIKO EPSON CORPORATION

CONTACT INFORMATION

Contact information is described in “SUPPLIERS” in the first pages of the following manual:

Robot System  Safety and Installation  Read this manual first
SAFETY PRECAUTIONS

Installation of robots and robotic equipment should only be performed by qualified personnel in accordance with national and local codes. Please carefully read this manual and other related manuals when using this software. Keep this manual in a handy location for easy access at all times.

<table>
<thead>
<tr>
<th>WARNING</th>
<th>This symbol indicates that a danger of possible serious injury or death exists if the associated instructions are not followed properly.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUTION</td>
<td>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.</td>
</tr>
</tbody>
</table>
## Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Category</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Management Commands</td>
<td>1</td>
</tr>
<tr>
<td>Robot Control Commands</td>
<td>2</td>
</tr>
<tr>
<td>Torque Commands</td>
<td>7</td>
</tr>
<tr>
<td>Input / Output Commands</td>
<td>7</td>
</tr>
<tr>
<td>Point Management Commands</td>
<td>9</td>
</tr>
<tr>
<td>Coordinate Change Commands</td>
<td>10</td>
</tr>
<tr>
<td>Program Control Commands</td>
<td>11</td>
</tr>
<tr>
<td>Program Execution Commands</td>
<td>12</td>
</tr>
<tr>
<td>Pseudo Statements</td>
<td>12</td>
</tr>
<tr>
<td>File Management Commands</td>
<td>13</td>
</tr>
<tr>
<td>Fieldbus Commands</td>
<td>14</td>
</tr>
<tr>
<td>Numeric Value Commands</td>
<td>14</td>
</tr>
<tr>
<td>String Commands</td>
<td>15</td>
</tr>
<tr>
<td>Logical Operators</td>
<td>15</td>
</tr>
<tr>
<td>Variable Commands</td>
<td>16</td>
</tr>
<tr>
<td>Security Commands</td>
<td>16</td>
</tr>
<tr>
<td>Conveyor Tracking Commands</td>
<td>16</td>
</tr>
<tr>
<td>Force Sensing Commands</td>
<td>17</td>
</tr>
<tr>
<td>DB Commands</td>
<td>17</td>
</tr>
<tr>
<td>PG Commands</td>
<td>17</td>
</tr>
<tr>
<td>Collision Detection Commands</td>
<td>17</td>
</tr>
<tr>
<td>Parts Consumption Commands</td>
<td>18</td>
</tr>
<tr>
<td>Simlator Commands</td>
<td>19</td>
</tr>
</tbody>
</table>

### SPEL+ Language Reference

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
</tr>
</tbody>
</table>

### SPEL+ Error Messages

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>791</td>
</tr>
</tbody>
</table>

### Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>922</td>
</tr>
</tbody>
</table>

### Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>932</td>
</tr>
<tr>
<td>932</td>
</tr>
<tr>
<td>942</td>
</tr>
<tr>
<td>953</td>
</tr>
</tbody>
</table>
## Appendix C: Commands of EPSON RC+7.0

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-1: List of Commands Added EPSON RC+4.0 or Later</td>
<td>964</td>
</tr>
<tr>
<td>C-2: List of Commands Added for Each Version of EPSON RC+ 7.0</td>
<td>967</td>
</tr>
<tr>
<td>C-3: Deletion Commands (Sort by Version)</td>
<td>973</td>
</tr>
</tbody>
</table>
# Summary of SPEL+ Commands

The following is a summary of SPEL+ commands.

## System Management Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>Resets the controller.</td>
</tr>
<tr>
<td>SysConfig</td>
<td>Displays controller setup.</td>
</tr>
<tr>
<td>SysErr</td>
<td>Returns the latest error status or warning status.</td>
</tr>
<tr>
<td>Date</td>
<td>Sets the system date.</td>
</tr>
<tr>
<td>Time</td>
<td>Sets system time.</td>
</tr>
<tr>
<td>Date$</td>
<td>Returns the system date as a string.</td>
</tr>
<tr>
<td>Time$</td>
<td>Returns system time as a string.</td>
</tr>
<tr>
<td>Hour</td>
<td>Displays / returns controller operation time.</td>
</tr>
<tr>
<td>Stat</td>
<td>Returns controller status bits.</td>
</tr>
<tr>
<td>CtrlInfo</td>
<td>Returns controller information.</td>
</tr>
<tr>
<td>RobotInfo</td>
<td>Returns robot information.</td>
</tr>
<tr>
<td>RobotInfo$</td>
<td>Returns robot text information.</td>
</tr>
<tr>
<td>TaskInfo</td>
<td>Returns task information.</td>
</tr>
<tr>
<td>TaskInfo$</td>
<td>Returns task text information.</td>
</tr>
<tr>
<td>DispDev</td>
<td>Sets the current display device.</td>
</tr>
<tr>
<td>EStopOn</td>
<td>Return the Emergency Stop status.</td>
</tr>
<tr>
<td>CtrlDev</td>
<td>Returns the current control device number.</td>
</tr>
<tr>
<td>Cls</td>
<td>Clears the EPSON RC+ 6.0 Run, Operator, or Command window text area. Clears the TP print panel.</td>
</tr>
<tr>
<td>Toff</td>
<td>Turns off execution line display on the LCD.</td>
</tr>
<tr>
<td>Ton</td>
<td>Specifies a task which shows an execution line on the LCD.</td>
</tr>
<tr>
<td>SafetyOn</td>
<td>Return the Safety Door open status.</td>
</tr>
<tr>
<td>Eval</td>
<td>Executes a Command window statement from a program and returns the error status.</td>
</tr>
<tr>
<td>ShutDown</td>
<td>Shuts down EPSON RC+ and optionally shuts down or restarts Windows.</td>
</tr>
<tr>
<td>TeachOn</td>
<td>Returns the Teach mode status.</td>
</tr>
<tr>
<td>WindowsStatus</td>
<td>Returns the Windows startup status.</td>
</tr>
</tbody>
</table>
## Robot Control Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIO_TrackingSet</td>
<td>Sets the distance tracking function.</td>
</tr>
<tr>
<td>AIO_TrackingStart</td>
<td>Starts the distance tracking function.</td>
</tr>
<tr>
<td>AIO_TrackingEnd</td>
<td>Ends the distance tracking function.</td>
</tr>
<tr>
<td>AIO_TrackingOn Function</td>
<td>Returns the status of the distance tracking function.</td>
</tr>
<tr>
<td>AtHome</td>
<td>Returns if the current robot orientation is Home position or not.</td>
</tr>
<tr>
<td>Calib</td>
<td>Replaces the current arm posture pulse values with the current CalPls values.</td>
</tr>
<tr>
<td>CalPls</td>
<td>Specifies and displays the position and orientation pulse values for calibration.</td>
</tr>
<tr>
<td>Hofs</td>
<td>Returns the offset pulses used for software zero point correction.</td>
</tr>
<tr>
<td>MCal</td>
<td>Executes machine calibration for robots with incremental encoders.</td>
</tr>
<tr>
<td>MCalComplete</td>
<td>Returns status of MCal.</td>
</tr>
<tr>
<td>MCordr</td>
<td>Specifies and displays the moving joint order for machine calibration Mcal. Required only for robots with incremental encoders.</td>
</tr>
<tr>
<td>Power</td>
<td>Sets / returns servo power mode.</td>
</tr>
<tr>
<td>Motor</td>
<td>Sets / returns motor status.</td>
</tr>
<tr>
<td>MHour Function</td>
<td>Returns the accumulated MOTOR ON time of the robot motors.</td>
</tr>
<tr>
<td>SFree</td>
<td>Removes servo power from the specified servo axis.</td>
</tr>
<tr>
<td>SLock</td>
<td>Restores servo power to the specified servo axis.</td>
</tr>
<tr>
<td>SyncRobots</td>
<td>Start the reserved robot motion.</td>
</tr>
<tr>
<td>Jump</td>
<td>Jumps to a point using point to point motion.</td>
</tr>
<tr>
<td>Jump3</td>
<td>Jumps to a point using 3D gate motion.</td>
</tr>
<tr>
<td>Jump3CP</td>
<td>Jumps to a point using 3D motion in continuous path.</td>
</tr>
<tr>
<td>JumpTLZ</td>
<td>Jumps to a point using 3D gate motion.</td>
</tr>
<tr>
<td>Arch</td>
<td>Sets / returns arch parameters for Jump motion.</td>
</tr>
<tr>
<td>LimZ</td>
<td>Sets the upper Z limit for the Jump command.</td>
</tr>
<tr>
<td>LimZMargin</td>
<td>Sets / returns the margin for error detection when the operation starts at the position higher than LimZ value.</td>
</tr>
<tr>
<td>Sense</td>
<td>Sets / returns the condition to stop the manipulator above the target coordinate when Sense is specified by Jump command.</td>
</tr>
<tr>
<td>JS</td>
<td>Returns status of Sense operation.</td>
</tr>
<tr>
<td>JT</td>
<td>Returns the status of the most recent Jump command for the current robot.</td>
</tr>
<tr>
<td>Go</td>
<td>Moves the robot to a point using point to point motion.</td>
</tr>
<tr>
<td>Pass</td>
<td>Executes simultaneous four joint Point to Point motion, passing near but not through the specified points.</td>
</tr>
<tr>
<td>Pulse</td>
<td>Moves the robot to a position defined in pulses.</td>
</tr>
<tr>
<td>BGo</td>
<td>Executes Point to Point relative motion, in the selected local coordinate system.</td>
</tr>
<tr>
<td>BMove</td>
<td>Executes linear interpolation relative motion, in the selected local coordinate system.</td>
</tr>
</tbody>
</table>
### Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TGo</strong></td>
<td>Executes Point to Point relative motion, in the current tool coordinate system.</td>
</tr>
<tr>
<td><strong>TMove</strong></td>
<td>Executes linear interpolation relative motion, in the selected tool coordinate system.</td>
</tr>
<tr>
<td><strong>Till</strong></td>
<td>Specifies motion stop when input occurs.</td>
</tr>
<tr>
<td><strong>TillOn</strong></td>
<td>Returns the current Till status.</td>
</tr>
<tr>
<td><strong>!...!</strong></td>
<td>Process statements during motion.</td>
</tr>
<tr>
<td><strong>Speed</strong></td>
<td>Sets / returns speed for point to point motion commands.</td>
</tr>
<tr>
<td><strong>Accel</strong></td>
<td>Sets / returns acceleration and deceleration for point to point motion.</td>
</tr>
<tr>
<td><strong>SpeedFactor</strong></td>
<td>Sets / returns speed for point to point motion commands.</td>
</tr>
<tr>
<td><strong>Inertia</strong></td>
<td>Specifies or displays the inertia settings for the robot arm.</td>
</tr>
<tr>
<td><strong>Weight</strong></td>
<td>Specifies or displays the weight settings for the robot arm.</td>
</tr>
<tr>
<td><strong>Arc</strong></td>
<td>Moves the arm using circular interpolation.</td>
</tr>
<tr>
<td><strong>Arc3</strong></td>
<td>Moves the arm in 3D using circular interpolation.</td>
</tr>
<tr>
<td><strong>Move</strong></td>
<td>Moves the robot using linear interpolation.</td>
</tr>
<tr>
<td><strong>Curve</strong></td>
<td>Defines the data and points required to move the arm along a curved path. Many data points can be defined in the path to improve precision of the path.</td>
</tr>
<tr>
<td><strong>CVMove</strong></td>
<td>Performs the continuous spline path motion defined by the Curve instruction.</td>
</tr>
<tr>
<td><strong>SpeedS</strong></td>
<td>Sets / returns speed for linear motion commands.</td>
</tr>
<tr>
<td><strong>AccelS</strong></td>
<td>Sets / returns acceleration and deceleration for linear motion.</td>
</tr>
<tr>
<td><strong>SpeedR</strong></td>
<td>Sets / returns speed for tool rotation.</td>
</tr>
<tr>
<td><strong>AccelR</strong></td>
<td>Sets / returns acceleration and deceleration for tool rotation.</td>
</tr>
<tr>
<td><strong>AccelMax</strong></td>
<td>Returns maximum acceleration value limit available for Accel.</td>
</tr>
<tr>
<td><strong>Brake</strong></td>
<td>Turns brake on or off for specified joint of the current robot.</td>
</tr>
<tr>
<td><strong>Home</strong></td>
<td>Moves robot to user defined home position.</td>
</tr>
<tr>
<td><strong>HomeClr</strong></td>
<td>Clears the home position definition.</td>
</tr>
<tr>
<td><strong>HomeDef</strong></td>
<td>Returns status of home position definition.</td>
</tr>
<tr>
<td><strong>HomeSet</strong></td>
<td>Sets user defined home position.</td>
</tr>
<tr>
<td><strong>Hordr</strong></td>
<td>Sets motion order for Home command.</td>
</tr>
<tr>
<td><strong>InPos</strong></td>
<td>Checks if robot is in position (not moving).</td>
</tr>
<tr>
<td><strong>CurPos</strong></td>
<td>Returns current position while moving.</td>
</tr>
<tr>
<td><strong>TCPSpeed</strong></td>
<td>Returns calculated current tool center point velocity.</td>
</tr>
<tr>
<td><strong>Pallet</strong></td>
<td>Defines a pallet or returns a pallet point.</td>
</tr>
<tr>
<td><strong>PalletClr</strong></td>
<td>Clears a pallet definition.</td>
</tr>
<tr>
<td><strong>Fine</strong></td>
<td>Specifies and displays the positioning error limits. (Unit: pulse)</td>
</tr>
<tr>
<td><strong>FineDist</strong></td>
<td>Specifies and displays the positioning error limits (Unit: mm)</td>
</tr>
<tr>
<td><strong>FineStatus Function</strong></td>
<td>Returns whether Fine or FineDist is used by the integer.</td>
</tr>
</tbody>
</table>
### Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>QP</strong></td>
<td>Sets / returns Quick Pause status.</td>
</tr>
<tr>
<td><strong>QPDecelR</strong></td>
<td>Sets the deceleration speed of quick pause for the change of tool orientation during the CP motion.</td>
</tr>
<tr>
<td><strong>QPDecelS</strong></td>
<td>Sets the deceleration speed of quick pause in the CP motion.</td>
</tr>
<tr>
<td><strong>CP</strong></td>
<td>Sets CP (Continuous Path) motion mode.</td>
</tr>
<tr>
<td><strong>Box</strong></td>
<td>Specifies and displays the approach check area.</td>
</tr>
<tr>
<td><strong>BoxClr</strong></td>
<td>Clears the definition of approach check area.</td>
</tr>
<tr>
<td><strong>BoxDef</strong></td>
<td>Returns whether Box has been defined or not.</td>
</tr>
<tr>
<td><strong>Plane</strong></td>
<td>Specifies and displays the approach check plane.</td>
</tr>
<tr>
<td><strong>PlaneClr</strong></td>
<td>Clears (undefines) a Plane definition.</td>
</tr>
<tr>
<td><strong>PlaneDef</strong></td>
<td>Returns the setting of the approach check plane.</td>
</tr>
<tr>
<td><strong>InsideBox</strong></td>
<td>Returns the check status of the approach check area.</td>
</tr>
<tr>
<td><strong>InsidePlane</strong></td>
<td>Returns the check status of the approach check plane.</td>
</tr>
<tr>
<td><strong>GetRobotInsideBox</strong></td>
<td>Returns a robot which is in the approach check area.</td>
</tr>
<tr>
<td><strong>GetRobotInsidePlane</strong></td>
<td>Returns a robot which is in the approach check plane.</td>
</tr>
<tr>
<td><strong>Find</strong></td>
<td>Specifies or displays the condition to store coordinates during motion.</td>
</tr>
<tr>
<td><strong>FindPos</strong></td>
<td>Returns a robot point stored by Find during a motion command.</td>
</tr>
<tr>
<td><strong>PosFound</strong></td>
<td>Returns status of Find operation.</td>
</tr>
<tr>
<td><strong>WaitPos</strong></td>
<td>Waits for robot to decelerate and stop at position before executing the next statement while path motion is active.</td>
</tr>
<tr>
<td><strong>Robot</strong></td>
<td>Selects the current robot. (Returns the robot number by Robot Function)</td>
</tr>
<tr>
<td><strong>RobotModel$</strong></td>
<td>Returns the robot model name.</td>
</tr>
<tr>
<td><strong>RobotName$</strong></td>
<td>Returns the robot name.</td>
</tr>
<tr>
<td><strong>RobotSerial$</strong></td>
<td>Returns the robot serial number.</td>
</tr>
<tr>
<td><strong>RobotType</strong></td>
<td>Returns the robot type.</td>
</tr>
<tr>
<td><strong>TargetOK</strong></td>
<td>Returns a status indicating whether or not the PTP (Point to Point) motion from the current position to a target position is possible.</td>
</tr>
<tr>
<td><strong>JRange</strong></td>
<td>Sets / returns joint limits for one joint.</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>Sets limits for all joints.</td>
</tr>
<tr>
<td><strong>XYLim</strong></td>
<td>Sets or displays the permissible XY motion range limits for the robot.</td>
</tr>
<tr>
<td><strong>XYLimClr</strong></td>
<td>Clears the XYLim definition.</td>
</tr>
<tr>
<td><strong>XYLimDef</strong></td>
<td>Returns whether XYLim has been defined or not.</td>
</tr>
<tr>
<td><strong>XY</strong></td>
<td>Returns a point from individual coordinates that can be used in a point expression.</td>
</tr>
</tbody>
</table>
### Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dist</td>
<td>Returns the distance between two robot points.</td>
</tr>
<tr>
<td>PTPBoost</td>
<td>Specifies or displays the acceleration, deceleration and speed algorithmic boost parameter for small distance PTP (point to point) motion.</td>
</tr>
<tr>
<td>PTPBoostOK</td>
<td>Returns whether or not the PTP (Point to Point) motion from a current position to a target position is a small travel distance.</td>
</tr>
<tr>
<td>PTPTime</td>
<td>Returns the estimated time for a point to point motion command without executing it.</td>
</tr>
<tr>
<td>CX</td>
<td>Sets / returns the X axis coordinate of a point.</td>
</tr>
<tr>
<td>CY</td>
<td>Sets / returns the Y axis coordinate of a point.</td>
</tr>
<tr>
<td>CZ</td>
<td>Sets / returns the Z axis coordinate of a point.</td>
</tr>
<tr>
<td>CU</td>
<td>Sets / returns the U axis coordinate of a point.</td>
</tr>
<tr>
<td>CV</td>
<td>Sets / returns the V axis coordinate of a point.</td>
</tr>
<tr>
<td>CW</td>
<td>Sets / returns the W axis coordinate of a point.</td>
</tr>
<tr>
<td>CR</td>
<td>Sets / returns the R axis coordinate of a point.</td>
</tr>
<tr>
<td>CS</td>
<td>Sets / returns the S axis coordinate of a point.</td>
</tr>
<tr>
<td>CT</td>
<td>Sets / returns the T axis coordinate of a point.</td>
</tr>
<tr>
<td>Pls</td>
<td>Returns the pulse value of one joint.</td>
</tr>
<tr>
<td>Agl</td>
<td>Returns joint angle at current position.</td>
</tr>
<tr>
<td>PAgl</td>
<td>Return a joint value from a specified point.</td>
</tr>
<tr>
<td>JA</td>
<td>Returns a robot point specified in joint angles.</td>
</tr>
<tr>
<td>AglToPls</td>
<td>Converts robot angles to pulses.</td>
</tr>
<tr>
<td>DegToRad</td>
<td>Converts degrees to radians.</td>
</tr>
<tr>
<td>RadToDeg</td>
<td>Converts radians to degrees.</td>
</tr>
<tr>
<td>Joint</td>
<td>Displays the current position for the robot in joint coordinates.</td>
</tr>
<tr>
<td>JTran</td>
<td>Perform a relative move of one joint.</td>
</tr>
<tr>
<td>PTran</td>
<td>Perform a relative move of one joint in pulses.</td>
</tr>
<tr>
<td>RealPls</td>
<td>Returns the pulse value of the specified joint.</td>
</tr>
<tr>
<td>RealPos</td>
<td>Returns the current position of the specified robot.</td>
</tr>
<tr>
<td>RealAccel Function</td>
<td>Returns the Accel value automatically adjusted by OLAccel.</td>
</tr>
<tr>
<td>PPls</td>
<td>Return the pulse position of a specified joint value from a specified point.</td>
</tr>
<tr>
<td>LJM Function</td>
<td>Returns the point data with the orientation flags converted to enable least joint motion when moving to a specified point based on the reference point.</td>
</tr>
<tr>
<td>AutoLJM</td>
<td>Sets the Auto LJM</td>
</tr>
<tr>
<td>AutoLJM Function</td>
<td>Returns the state of the Auto LJM</td>
</tr>
<tr>
<td>AutoOrientationFlag</td>
<td>Changes orientation flag of N6-A1000**</td>
</tr>
<tr>
<td>AutoOrientationFlag Function</td>
<td>Returns the state of the AutoOrientationFlag</td>
</tr>
<tr>
<td>AvoidSingularity</td>
<td>Sets the Singularity avoiding function</td>
</tr>
<tr>
<td>AvoidSingularity Function</td>
<td>Returns the state of the Singularity avoiding function</td>
</tr>
<tr>
<td>SingularityAngle</td>
<td>Sets the singularity neighborhood angle for the singularity avoiding function</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SingularityAngle Function</td>
<td>Returns the singularity neighborhood angle for the singularity avoiding function</td>
</tr>
<tr>
<td>SingularitySpeed</td>
<td>Sets the singularity neighborhood speed for the singularity avoiding function</td>
</tr>
<tr>
<td>SingularitySpeed Function</td>
<td>Returns the singularity neighborhood speed for the singularity avoiding function</td>
</tr>
<tr>
<td>SingularityDist</td>
<td>Sets the singularity neighborhood distance necessary for the singularity avoiding function</td>
</tr>
<tr>
<td>SingularityDist Function</td>
<td>Returns the singularity neighborhood distance necessary for the singularity avoiding function</td>
</tr>
<tr>
<td>AbortMotion</td>
<td>Aborts a motion command and puts the running task in error status.</td>
</tr>
<tr>
<td>Align Function</td>
<td>Returns point data converted to align robot orientation with the nearest coordinate axis in local coordinate system.</td>
</tr>
<tr>
<td>AlignECP Function</td>
<td>Returns point data converted to align robot orientation with a nearest coordinate axis in ECP coordinate system.</td>
</tr>
<tr>
<td>SoftCP</td>
<td>Sets / returns SoftCP motion mode.</td>
</tr>
<tr>
<td>SoftCP Function</td>
<td>Returns the status of SoftCP motion mode.</td>
</tr>
<tr>
<td>Here</td>
<td>Teach a robot point at the current position.</td>
</tr>
<tr>
<td>Where</td>
<td>Displays current robot position data.</td>
</tr>
<tr>
<td>PerformMode</td>
<td>Sets the mode of the robot.</td>
</tr>
<tr>
<td>PerformMode Function</td>
<td>Returns the robot performance mode number.</td>
</tr>
<tr>
<td>VSD</td>
<td>Sets the variable speed CP motion of SCARA robots.</td>
</tr>
<tr>
<td>VSD Function</td>
<td>Returns the variable speed CP motion setting of SCARA robots.</td>
</tr>
<tr>
<td>CP_Offset</td>
<td>Sets the offset time to start the subsequent motion command when executing CP On.</td>
</tr>
<tr>
<td>CP_Offset Function</td>
<td>Returns the offset time to start the subsequent motion command when executing CP On.</td>
</tr>
<tr>
<td>AvgSpeedClear</td>
<td>Clears and initializes the average of the joint speed.</td>
</tr>
<tr>
<td>AvgSpeed</td>
<td>Displays the average of the joint speed.</td>
</tr>
<tr>
<td>AvgSpeed Function</td>
<td>Returns the average value of the joint speed.</td>
</tr>
<tr>
<td>PeakSpeedClear</td>
<td>Clears and initializes the peak speed for one or more joints.</td>
</tr>
<tr>
<td>PeakSpeed</td>
<td>Displays the peak speed values for the specified joint.</td>
</tr>
<tr>
<td>PeakSpeed Function</td>
<td>Returns the peak speed for the specified joint.</td>
</tr>
</tbody>
</table>
### Torque Commands

**TC**
- Returns the torque control mode setting and current mode.

**TCSpeed**
- Specifies the speed limit in the torque control.

**TCLim**
- Specifies the torque limit of each joint for the torque control mode.

**RealTorque**
- Returns the current torque instruction value of the specified joint.

**ATCLR**
- Clears and initializes the average torque for one or more joints.

**ATRQ**
- Displays the average torque for the specified joint.

**PTCLR**
- Clears and initializes the peak torque for one or more joints.

**PTRQ**
- Displays the peak torque for the specified joint.

**OLAccel**
- Sets up the automatic adjustment of acceleration/deceleration that is adjusted.

**OLRate**
- Display overload rating for one or all joints for the current robot.

**LimitTorque**
- Sets / returns the upper torque value in High power mode.

**LimitTorque Function**
- Returns the LimitTorque setting value.

**LimitTorqueLP**
- Sets / returns the upper limit torque value in Low power mode.

**LimitTorqueLP Function**
- Returns the LimitTorqueLP setting value.

**LimitTorqueStop**
- Specifies /returns whether or not to stop the robot when torque reaches the upper limit in High power mode.

**LimitTorqueStop Function**
- Returns the LimitTorqueStop setting value.

**LimitTorqueStopLP**
- Specifies / returns whether or not to stop the robot when torque reaches the upper limit in Low power mode.

**LimitTorqueStopLP Function**
- Returns the LimitTorqueStopLP setting value.

### Input / Output Commands

**On**
- Turns an output on.

**Off**
- Turns an output off.

**Oport**
- Reads status of one output bit.

**Sw**
- Returns status of input.

**In**
- Reads 8 bits of inputs.

**InW**
- Returns the status of the specified input word port.

**InBCD**
- Reads 8 bits of inputs in BCD format.

**Out**
- Sets / returns 8 bits of outputs.

**OutW**
- Simultaneously sets 16 output bits.

**OpBCD**
- Simultaneously sets 8 output bits using BCD format.

**MemOn**
- Turns a memory bit on.

**MemOff**
- Turns a memory bit off.

**MemSw**
- Returns status of memory bit.

**MemIn**
- Reads 8 bits of memory I/O.

**MemOut**
- Sets / returns 8 memory bits.

**MemInW**
- Returns the status of the specified memory I/O word port.
  - Each word port contains 16 memory I/O bits.

**MemOutW**
- Simultaneously sets 16 memory I/O bits.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Wait</strong></td>
<td>Wait for condition or time.</td>
</tr>
<tr>
<td><strong>TMOOut</strong></td>
<td>Sets default time out for Wait statement.</td>
</tr>
<tr>
<td><strong>Tw</strong></td>
<td>Returns the status of the Wait condition and Wait timer interval.</td>
</tr>
<tr>
<td><strong>Input</strong></td>
<td>Receives input data from the display device and stored in a variable(s).</td>
</tr>
<tr>
<td><strong>InReal</strong></td>
<td>Reads an input data of 2 words (32 bits) as a floating-point data (IEEE754 compliant) of 32 bits.</td>
</tr>
<tr>
<td><strong>Print</strong></td>
<td>Display characters on current display window.</td>
</tr>
<tr>
<td><strong>Line Input</strong></td>
<td>Input a string from the current display window.</td>
</tr>
<tr>
<td><strong>Input #</strong></td>
<td>Allows string or numeric data to be received from a file, communications port, or database and stored in one or more variables.</td>
</tr>
<tr>
<td><strong>Print #</strong></td>
<td>Outputs data to the specified file, communications port, database, or device.</td>
</tr>
<tr>
<td><strong>Line Input #</strong></td>
<td>Reads data of one line from a file, communication port, database, or the device.</td>
</tr>
<tr>
<td><strong>Lof</strong></td>
<td>Checks whether the specified RS-232 or TCP/IP port has any lines of data in its buffer.</td>
</tr>
<tr>
<td><strong>SetIn</strong></td>
<td>For Virtual IO, sets specified input port (8 bits) to the specified value.</td>
</tr>
<tr>
<td><strong>SetInW</strong></td>
<td>For Virtual IO, sets specified input word (16 bits) to the specified value.</td>
</tr>
<tr>
<td><strong>SetSw</strong></td>
<td>For Virtual IO, sets specified input bit to the specified value.</td>
</tr>
<tr>
<td><strong>IOLabel$</strong></td>
<td>Returns the I/O label for a specified input or output bit, byte, or word.</td>
</tr>
<tr>
<td><strong>IONumber</strong></td>
<td>Returns the I/O number of the specified I/O label.</td>
</tr>
<tr>
<td><strong>IODEf</strong></td>
<td>Returns whether the specified I/O label is defined.</td>
</tr>
<tr>
<td><strong>OpenCom</strong></td>
<td>Open an RS-232 communication port.</td>
</tr>
<tr>
<td><strong>OpenCom Function</strong></td>
<td>Acquires the task number that executes OpenCom.</td>
</tr>
<tr>
<td><strong>CloseCom</strong></td>
<td>Close the RS-232C port that has been opened with OpenCom.</td>
</tr>
<tr>
<td><strong>SetCom</strong></td>
<td>Sets or displays parameters for RS-232C port.</td>
</tr>
<tr>
<td><strong>ChkCom</strong></td>
<td>Returns number of characters in the reception buffer of a communication port</td>
</tr>
<tr>
<td><strong>OpenNet</strong></td>
<td>Open a TCP/IP network port.</td>
</tr>
<tr>
<td><strong>OpenNet Function</strong></td>
<td>Acquires the task number that executes OpenNet.</td>
</tr>
<tr>
<td><strong>OutReal</strong></td>
<td>Output the output data of real value as the floating-point data (IEEE754 compliant) of 32 bits to the output port 2 words (32 bits).</td>
</tr>
<tr>
<td><strong>CloseNet</strong></td>
<td>Close the TCP/IP port previously opened with OpenNet.</td>
</tr>
<tr>
<td><strong>SetNet</strong></td>
<td>Sets parameters for a TCP/IP port.</td>
</tr>
<tr>
<td><strong>ChkNet</strong></td>
<td>Returns number of characters in the reception buffer of a network port.</td>
</tr>
</tbody>
</table>
### Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WaitNet</td>
<td>Wait for TCP/IP port connection to be established.</td>
</tr>
<tr>
<td>Read</td>
<td>Reads characters from a file or communications port.</td>
</tr>
<tr>
<td>ReadBin</td>
<td>Reads binary data from a file or communications port.</td>
</tr>
<tr>
<td>Write</td>
<td>Writes characters to a file or communication port without end of line terminator.</td>
</tr>
<tr>
<td>WriteBin</td>
<td>Writes binary data to a file or communications port.</td>
</tr>
<tr>
<td>InputBox</td>
<td>Displays a prompt in a dialog box, waits for the operator to input text or choose a button, and returns the contents of the box.</td>
</tr>
<tr>
<td>MsgBox</td>
<td>Displays a message in a dialog box and waits for the operator to choose a button.</td>
</tr>
<tr>
<td>RunDialog</td>
<td>Runs an EPSON RC+ dialog from a SPEL+ program.</td>
</tr>
<tr>
<td>LatchEnable</td>
<td>Enable / Disable the latch function for the robot position by the R-I/O input.</td>
</tr>
<tr>
<td>LatchState Function</td>
<td>Returns the latch state of robot position using the R-I/O.</td>
</tr>
<tr>
<td>LatchPos Function</td>
<td>Returns the robot position latched using the R-I/O input signal.</td>
</tr>
<tr>
<td>SetLatch</td>
<td>Sets the latch function of the robot position using the R-I/O input.</td>
</tr>
<tr>
<td>AIO_In Function</td>
<td>Reads analog value form analog I/O input channel.</td>
</tr>
<tr>
<td>AIO_InW Function</td>
<td>Reads one word input data from analog I/O input channel.</td>
</tr>
<tr>
<td>AIO_Out</td>
<td>Output analog value on the analog I/O output channel.</td>
</tr>
<tr>
<td>AIO_Out Function</td>
<td>Returns the output state of analog I/O output channel.</td>
</tr>
<tr>
<td>AIO_OutW</td>
<td>Output the one word data to analog I/O output channel.</td>
</tr>
<tr>
<td>AIO_OutW Function</td>
<td>Returns the output state by one word of analog I/O output channel.</td>
</tr>
<tr>
<td>AIO_Set</td>
<td>Output the speed information to analog I/O output channel.</td>
</tr>
<tr>
<td>AIO_Set Function</td>
<td>Returns setting information of robot speed output which is set on optional analog I/O output channel.</td>
</tr>
</tbody>
</table>

### Point Management Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClearPoints</td>
<td>Clears all point data in memory.</td>
</tr>
<tr>
<td>LoadPoints</td>
<td>Loads point data from a file in memory.</td>
</tr>
<tr>
<td>SavePoints</td>
<td>Saves point data to a file in memory.</td>
</tr>
<tr>
<td>ImportPoints</td>
<td>Imports a point file into the current project for the specified robot.</td>
</tr>
<tr>
<td>ExportPoints</td>
<td>Exports a point file to the specified path in the PC.</td>
</tr>
<tr>
<td>P#</td>
<td>Defines a specified point.</td>
</tr>
<tr>
<td>PDef</td>
<td>Returns the definition status of a specified point.</td>
</tr>
<tr>
<td>PDel</td>
<td>Deletes specified position data.</td>
</tr>
<tr>
<td>PLabel</td>
<td>Defines a label for a specified point.</td>
</tr>
<tr>
<td>PLabel$</td>
<td>Returns the point label associated with a point number.</td>
</tr>
<tr>
<td>PNumber</td>
<td>Returns the point number associated with a point label.</td>
</tr>
<tr>
<td>PLIST</td>
<td>Displays point data in memory for the current robot.</td>
</tr>
<tr>
<td>PLocal</td>
<td>Sets the local attribute for a point.</td>
</tr>
<tr>
<td>PDescription</td>
<td>Define a description of specified point data.</td>
</tr>
<tr>
<td>PDescription$</td>
<td>Returns description of point that defined to the specified point number.</td>
</tr>
</tbody>
</table>
### Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WorkQue_Add</strong></td>
<td>Adds the work queue data (point data and user data) to the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_AutoRemove</strong></td>
<td>Sets the auto delete function to the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_AutoRemove Function</strong></td>
<td>Returns the state of the auto delete function set to the work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Get Function</strong></td>
<td>Returns the point data from the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Len Function</strong></td>
<td>Returns the number of the valid work queue data registered to the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_List</strong></td>
<td>Displays the work queue data list (point data and user data) of the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Reject</strong></td>
<td>Sets and displays the minimum distance for double registration prevention of the point data to the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Reject Function</strong></td>
<td>Returns the distance of the double registration prevention set to the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Remoove</strong></td>
<td>Deletes the work queue data (point data and user data) from the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Sort</strong></td>
<td>Sets and displays the Sort type of the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_Sort Function</strong></td>
<td>Returns the Sort type of the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_UserData</strong></td>
<td>Resets and displays the user data (real number) registered to the specified work queue.</td>
</tr>
<tr>
<td><strong>WorkQue_UserData Function</strong></td>
<td>Returns the user data (real number) registered to the specified work queue.</td>
</tr>
</tbody>
</table>

### Coordinate Change Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Arm</strong></td>
<td>Sets / returns current arm.</td>
</tr>
<tr>
<td><strong>ArmSet</strong></td>
<td>Defines an arm.</td>
</tr>
<tr>
<td><strong>ArmDef</strong></td>
<td>Returns status of arm definition.</td>
</tr>
<tr>
<td><strong>ArmClr</strong></td>
<td>Clears an arm definition.</td>
</tr>
<tr>
<td><strong>Tool</strong></td>
<td>Sets / returns the current tool number.</td>
</tr>
<tr>
<td><strong>TLSet</strong></td>
<td>Defines or displays a tool coordinate system.</td>
</tr>
<tr>
<td><strong>TLDInt</strong></td>
<td>Returns status of tool definition.</td>
</tr>
<tr>
<td><strong>TLCInt</strong></td>
<td>Clears a tool definition.</td>
</tr>
<tr>
<td><strong>ECP</strong></td>
<td>Sets / returns the current ECP number.</td>
</tr>
<tr>
<td><strong>ECPSInt</strong></td>
<td>Defines or displays an external control point.</td>
</tr>
<tr>
<td><strong>ECPCDef</strong></td>
<td>Returns status of ECP definition.</td>
</tr>
<tr>
<td><strong>ECPClr</strong></td>
<td>Clears an ECP definition.</td>
</tr>
<tr>
<td><strong>Base</strong></td>
<td>Defines and displays the base coordinate system.</td>
</tr>
<tr>
<td><strong>Local</strong></td>
<td>Define a local coordinate system.</td>
</tr>
<tr>
<td><strong>LocalDef</strong></td>
<td>Returns status of local definition.</td>
</tr>
<tr>
<td><strong>LocalClr</strong></td>
<td>Clears (undefines) a local coordinate system.</td>
</tr>
</tbody>
</table>
Summary of SPEL+ Commands

Elbow
Sets / returns elbow orientation of a point.

Hand
Sets / returns hand orientation of a point.

Wrist
Sets / returns wrist orientation of a point.

J4Flag
Sets / returns the J4Flag setting of a point.

J6Flag
Sets / returns the J6Flag orientation of a point.

J1Flag
Sets / returns the J1Flag setting of a point.

J2Flag
Sets / returns the J2Flag orientation of a point.

J1Angle
Returns the J1Angle attribute of a point.

J4Angle
Returns the J4Angle attribute of a point.

VxCalib
Creates the calibration data.

VxTrans
Converts the pixel coordinates to the robot coordinates and returns the converted the point data.

VxCalInfo
Returns the calibration completion status / calibration data.

VxCalDelete
Deletes the calibration data.

VxCalSave
Saves the calibration data to the file.

VxCalLoad
Loads the calibration data from the file.

Program Control Commands

Function
Declare a function.

For...Next
Executes one or more statements for a specific count.

GoSub
Execute a subroutine.

Return
Returns from a subroutine.

GoTo
Branch unconditionally to a line number or label.

Call
Call a user function.

If..Then..Else..EndIf
Conditional statement execution.

Else
Used with the If instruction to allow statements to be executed when the condition used with the If instruction is False. Else is an option for the If/Then instruction.

Select ... Send
Executes one of several groups of statements, depending on the value of an expression.

Do...Loop
Do...Loop construct.

Declare
Declares an external function in a dynamic link library (DLL).

Trap
Specify a trap handler.

OnErr
Defines an error handler.

Era
Returns the robot joint number for last error.

Erf$
Returns the function name for last error.

Erl
Returns the line number of error.

Err
Returns the error number.

Ert
Returns the task number of error.

Errb
Returns the robot number of error.

ErrMsg$
Returns the error message.

Signal
Sends a signal to tasks executing WaitSig.

SyncLock
Synchronizes tasks using a mutual exclusion lock.

SynUnlock
Unlocks a sync ID that was previously locked with SyncLock.

WaitSig
Waits for a signal from another task.
Summary of SPEL+ Commands

**ErrorOn**
Returns the error status of the controller.

**Error**
Generates a user error.

**EResume**
Resumes execution after an error-handling routine is finished.

**PauseOn**
Returns the pause status.

**Exit**
Exits a loop construct or function.

---

Program Execution Commands

**Xqt**
Execute a task.

**Pause**
Pause all tasks that have pause enabled.

**Cont**
Resumes the controller after a Pause statement has been executed and continues the execution of all tasks.

**Halt**
Suspend a task.

**Quit**
Quits a task.

**Resume**
Resume a task in the halt state.

**MyTask**
Returns current task.

**TaskDone**
Returns the completion status of a task.

**TaskState**
Returns the current state of a task.

**TaskWait**
Waits to for a task to terminate.

**Restart**
Re starts the current main program group.

**Recover**
Executes safeguard position recovery and returns status.

**RecoverPos**
Returns the position where a robot was in when safeguard was open.

**StartMain**
Executes the main function from a background task.

---

Pseudo Statements

**#define**
Defines a macro.

**#ifdef ... #endif**
Conditional compile.

**#ifndef ... #endif**
Conditional compile.

**#include**
Include a file.

**#undef**
Undefines an identifier previously defined with #define.
### File Management Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChDir</td>
<td>Changes and displays the current directory.</td>
</tr>
<tr>
<td>ChDisk</td>
<td>Sets the object disk for file operations.</td>
</tr>
<tr>
<td>MkDir</td>
<td>Creates a subdirectory on a controller disk drive.</td>
</tr>
<tr>
<td>RmDir</td>
<td>Removes an empty subdirectory from a controller disk drive.</td>
</tr>
<tr>
<td>RenDir</td>
<td>Rename a directory.</td>
</tr>
<tr>
<td>FileDateTime$</td>
<td>Returns the date and time of a file.</td>
</tr>
<tr>
<td>FileExists</td>
<td>Checks if a file exists.</td>
</tr>
<tr>
<td>FileLen</td>
<td>Returns the length of a file.</td>
</tr>
<tr>
<td>FolderExists</td>
<td>Checks if a folder exists.</td>
</tr>
<tr>
<td>Del</td>
<td>Deletes one or more files.</td>
</tr>
<tr>
<td>Copy</td>
<td>Copies a file to another location.</td>
</tr>
<tr>
<td>Rename</td>
<td>Renames a file.</td>
</tr>
<tr>
<td>AOpen</td>
<td>Opens file in the appending mode.</td>
</tr>
<tr>
<td>BOpen</td>
<td>Opens file in binary mode.</td>
</tr>
<tr>
<td>ROpen</td>
<td>Opens a file for reading.</td>
</tr>
<tr>
<td>UOpen</td>
<td>Opens a file for read/write access.</td>
</tr>
<tr>
<td>WOpen</td>
<td>Opens a file for writing.</td>
</tr>
<tr>
<td>Input #</td>
<td>Allows string or numeric data to be received from a file, communications port, or database and stored in one or more variables.</td>
</tr>
<tr>
<td>Print #</td>
<td>Outputs data to the specified file, communications port, database, or device.</td>
</tr>
<tr>
<td>Line Input #</td>
<td>Reads data of one line from a file, communication port, database, or the device.</td>
</tr>
<tr>
<td>Read</td>
<td>Reads characters from a file or communications port.</td>
</tr>
<tr>
<td>ReadBin</td>
<td>Reads binary data from a file or communications port.</td>
</tr>
<tr>
<td>Write</td>
<td>Writes characters to a file or communication port without end of line terminator.</td>
</tr>
<tr>
<td>WriteBin</td>
<td>Writes binary data to a file or communications port.</td>
</tr>
<tr>
<td>Seek</td>
<td>Changes position of file pointer for a specified file.</td>
</tr>
<tr>
<td>Close</td>
<td>Closes a file.</td>
</tr>
<tr>
<td>Eof</td>
<td>Returns end of file status.</td>
</tr>
<tr>
<td>ChDrive</td>
<td>Changes the current disk drive for file operations.</td>
</tr>
<tr>
<td>CurDir$</td>
<td>Returns a string representing the current directory.</td>
</tr>
<tr>
<td>CurDrive$</td>
<td>Returns a string representing the current directory.</td>
</tr>
<tr>
<td>CurDisk$</td>
<td>Returns a string representing the current disk.</td>
</tr>
<tr>
<td>Flush</td>
<td>Writes a file's buffer into the file.</td>
</tr>
</tbody>
</table>
## Summary of SPEL+ Commands

### Fieldbus Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FbusIO_GetBusStatus</td>
<td>Returns the status of the specified Fieldbus.</td>
</tr>
<tr>
<td>FbusIO_GetDeviceStatus</td>
<td>Returns the status of the specified Fieldbus device.</td>
</tr>
<tr>
<td>FbusIO_SendMsg</td>
<td>Sends an explicit message to a Fieldbus device and returns the reply.</td>
</tr>
</tbody>
</table>

### Numeric Value Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctr</td>
<td>Return the value of a counter.</td>
</tr>
<tr>
<td>CTReset</td>
<td>Resets a counter.</td>
</tr>
<tr>
<td>ElapsedTime</td>
<td>Measures a takt time.</td>
</tr>
<tr>
<td>ResetElapsedTime</td>
<td>Resets and starts a takt time measurement timer.</td>
</tr>
<tr>
<td>Tmr</td>
<td>Returns the value of a timer.</td>
</tr>
<tr>
<td>TmReset</td>
<td>Resets a timer to 0.</td>
</tr>
<tr>
<td>Sin</td>
<td>Returns the sine of an angle.</td>
</tr>
<tr>
<td>Cos</td>
<td>Returns cosine of an angle.</td>
</tr>
<tr>
<td>Tan</td>
<td>Returns the tangent of an angle.</td>
</tr>
<tr>
<td>Acos</td>
<td>Returns arccosine.</td>
</tr>
<tr>
<td>Asin</td>
<td>Returns arcsine.</td>
</tr>
<tr>
<td>Atan</td>
<td>Returns arctangent.</td>
</tr>
<tr>
<td>Atan2</td>
<td>Returns arctangent based on X, Y position.</td>
</tr>
<tr>
<td>Sqr</td>
<td>Returns the square root of a number.</td>
</tr>
<tr>
<td>Abs</td>
<td>Returns the absolute value of a number.</td>
</tr>
<tr>
<td>Sgn</td>
<td>Returns the sign of a number.</td>
</tr>
<tr>
<td>Int</td>
<td>Converts a real number to an integer.</td>
</tr>
<tr>
<td>BClr</td>
<td>Clears one bit in a number and return the new value.</td>
</tr>
<tr>
<td>BSet</td>
<td>Sets a bit in a number and returns the new value.</td>
</tr>
<tr>
<td>BTst</td>
<td>Returns the status of 1 bit in a number.</td>
</tr>
<tr>
<td>BClr64</td>
<td>Clears one bit in a number and return the new value.</td>
</tr>
<tr>
<td>BSet64</td>
<td>Sets a bit in a number and returns the new value.</td>
</tr>
<tr>
<td>BTst64</td>
<td>Returns the status of 1 bit in a number.</td>
</tr>
<tr>
<td>Fix</td>
<td>Returns the integer portion of a real number.</td>
</tr>
<tr>
<td>Hex</td>
<td>Returns a string representing a specified number in hexadecimal format.</td>
</tr>
<tr>
<td>Randomize</td>
<td>Initializes the random-number generator.</td>
</tr>
<tr>
<td>Redim</td>
<td>Redimension an array at run-time.</td>
</tr>
<tr>
<td>Rnd</td>
<td>Return a random number.</td>
</tr>
<tr>
<td>UBound</td>
<td>Returns the largest available subscript for the indicated dimension of an array.</td>
</tr>
</tbody>
</table>
### String Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asc</td>
<td>Returns the ASCII value of a character.</td>
</tr>
<tr>
<td>Chr$</td>
<td>Returns the character of a numeric ASCII value.</td>
</tr>
<tr>
<td>Left$</td>
<td>Returns a substring from the left side of a string.</td>
</tr>
<tr>
<td>Mid$</td>
<td>Returns a substring.</td>
</tr>
<tr>
<td>Right$</td>
<td>Returns a substring from the right side of a string.</td>
</tr>
<tr>
<td>Len</td>
<td>Returns the length of a string.</td>
</tr>
<tr>
<td>LSet$</td>
<td>Returns a string padded with trailing spaces.</td>
</tr>
<tr>
<td>RSet$</td>
<td>Returns a string padded with leading spaces.</td>
</tr>
<tr>
<td>Space$</td>
<td>Returns a string containing space characters.</td>
</tr>
<tr>
<td>Str$</td>
<td>Converts a number to a string.</td>
</tr>
<tr>
<td>Val</td>
<td>Converts a numeric string to a number.</td>
</tr>
<tr>
<td>LCase$</td>
<td>Converts a string to lower case.</td>
</tr>
<tr>
<td>UCase$</td>
<td>Converts a string to upper case.</td>
</tr>
<tr>
<td>LTrim$</td>
<td>Removes spaces from beginning of string.</td>
</tr>
<tr>
<td>RTrim$</td>
<td>Removes spaces from end of string.</td>
</tr>
<tr>
<td>Trim$</td>
<td>Removes spaces from beginning and end of string.</td>
</tr>
<tr>
<td>ParseStr</td>
<td>Parse a string and return array of tokens.</td>
</tr>
<tr>
<td>FmtStr</td>
<td>Format a number or string.</td>
</tr>
<tr>
<td>FmtStr$</td>
<td>Format a number or string.</td>
</tr>
<tr>
<td>InStr</td>
<td>Returns position of one string within another.</td>
</tr>
<tr>
<td>Tab$</td>
<td>Returns a string containing the specified number of tabs characters.</td>
</tr>
</tbody>
</table>

### Logical Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>And</td>
<td>Performs logical and bitwise AND operation.</td>
</tr>
<tr>
<td>Or</td>
<td>Or operator.</td>
</tr>
<tr>
<td>LShift</td>
<td>Shifts bits to the left.</td>
</tr>
<tr>
<td>LShift64</td>
<td>Shifts bits to the left.</td>
</tr>
<tr>
<td>Mod</td>
<td>Modulus operator.</td>
</tr>
<tr>
<td>Not</td>
<td>Not operator.</td>
</tr>
<tr>
<td>RShift</td>
<td>Shifts bits to the right.</td>
</tr>
<tr>
<td>RShift64</td>
<td>Shifts bits to the right.</td>
</tr>
<tr>
<td>Xor</td>
<td>Exclusive Or operator.</td>
</tr>
<tr>
<td>Mask</td>
<td>Performs bitwise AND operation in Wait statements.</td>
</tr>
</tbody>
</table>
Summary of SPEL+ Commands

Variable commands

<table>
<thead>
<tr>
<th>Variable Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boolean</td>
<td>Declares Boolean variables.</td>
</tr>
<tr>
<td>Byte</td>
<td>Declares byte variables.</td>
</tr>
<tr>
<td>Double</td>
<td>Declares double variables.</td>
</tr>
<tr>
<td>Global</td>
<td>Declares global variables.</td>
</tr>
<tr>
<td>Int32</td>
<td>Declares 4-byte integer variables.</td>
</tr>
<tr>
<td>Integer</td>
<td>Declares 2-byte integer variables.</td>
</tr>
<tr>
<td>Long</td>
<td>Declares long integer variables.</td>
</tr>
<tr>
<td>Int64</td>
<td>Declares 8-byte integer variables.</td>
</tr>
<tr>
<td>Real</td>
<td>Declares real variables.</td>
</tr>
<tr>
<td>Short</td>
<td>Declares 2-byte integer variables.</td>
</tr>
<tr>
<td>String</td>
<td>Declares string variables.</td>
</tr>
<tr>
<td>UByte</td>
<td>Declares unsigned integer variables.</td>
</tr>
<tr>
<td>UInt32</td>
<td>Declares unsigned 4-byte integer variables.</td>
</tr>
<tr>
<td>UShort</td>
<td>Declares unsigned 2-byte integer variables.</td>
</tr>
<tr>
<td>UInt64</td>
<td>Declares unsigned 8-byte integer variables.</td>
</tr>
</tbody>
</table>

Security Commands

- **GetCurrentUser**$ Returns the current EPSON RC+ user.
- **Login** Log into EPSON RC+ 6.0 as another user.

Conveyor Tracking Commands

- **Cnv_AbortTrack** Aborts tracking motion to a conveyor queue point.
- **Cnv_Accel Function** Returns acceleration and deceleration for the conveyor.
- **Cnv_Accel** Sets acceleration and deceleration for the conveyor.
- **Cnv_Downstream Function** Returns the downstream limit for the specified conveyor.
- **Cnv_Downstream** Sets the downstream limit for the specified conveyor.
- **Cnv_Fine Function** Returns the current Cnv_Fine setting.
- **Cnv_Fine** Sets the value of Cnv_Fine for one conveyor.
- **Cnv_Flag Function** Returns the tracking state of the robot.
- **Cnv_Mode Function** Returns the setting mode value of the conveyor.
- **Cnv_Mode** Sets the setting mode value of the conveyor.
- **Cnv_Name$ Function** Returns the name of the specified conveyor.
- **Cnv_Number Function** Returns the number of a conveyor specified by name.
- **Cnv_OffsetAngle** Sets the offset value for the conveyor queue data.
- **Cnv_OffsetAngle Function** Returns the offset value of the conveyor queue data.
- **Cnv_Point Function** Returns a robot point in the specified conveyor's coordinate system derived from sensor coordinates.
- **Cnv_PosErr Function** Returns deviation in current tracking position compared to tracking target.
- **Cnv_Pulse Function** Returns the current position of a conveyor in pulses.
- **Cnv_QueueAdd** Adds a robot point to a conveyor queue.
- **Cnv_QueueGet Function** Returns a point from the specified conveyor's queue.
- **Cnv_QueueLen Function** Returns the number of items in the specified conveyor's queue.
- **Cnv_QueueList** Displays a list of items in the specified conveyor's queue.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cnv_QueMove</td>
<td>Moves data from upstream conveyor queue to downstream conveyor queue.</td>
</tr>
<tr>
<td>Cnv_QueReject</td>
<td>Sets and displays the queue reject distance for a conveyor.</td>
</tr>
<tr>
<td>Cnv_QueReject Function</td>
<td>Returns the current part reject distance for a conveyor.</td>
</tr>
<tr>
<td>Cnv_QueRemove</td>
<td>Removes items from a conveyor queue.</td>
</tr>
<tr>
<td>Cnv_QueUserData</td>
<td>Sets and displays user data associated with a queue entry.</td>
</tr>
<tr>
<td>Cnv_QueUserData Function</td>
<td>Returns the user data value associated with an item in a conveyor queue.</td>
</tr>
<tr>
<td>Cnv_RobotConveyor Function</td>
<td>Returns the conveyor being tracked by a robot.</td>
</tr>
<tr>
<td>Cnv_Speed Function</td>
<td>Returns the current speed of a conveyor.</td>
</tr>
<tr>
<td>Cnv_Trigger</td>
<td>Latches current conveyor position for the next Cnv_QueAdd statement.</td>
</tr>
<tr>
<td>Cnv_Upstream Function</td>
<td>Returns the upstream limit for the specified conveyor.</td>
</tr>
<tr>
<td>Cnv_Upstream</td>
<td>Sets the upstream limit for the specified conveyor.</td>
</tr>
</tbody>
</table>

**Force Sensing Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Force_Calibrate</td>
<td>Sets zero offsets for all axes for the current force sensor.</td>
</tr>
<tr>
<td>Force_ClearTrigger</td>
<td>Clears all trigger conditions for the current force sensor.</td>
</tr>
<tr>
<td>Force_GetForces</td>
<td>Returns the forces and torques for all force sensor axes in an array.</td>
</tr>
<tr>
<td>Force_GetForce Function</td>
<td>Returns the force for a specified axis.</td>
</tr>
<tr>
<td>Force_Sensor</td>
<td>Sets the current force sensor for the current task.</td>
</tr>
<tr>
<td>Force_Sensor Function</td>
<td>Returns the current force sensor for the current task.</td>
</tr>
<tr>
<td>Force_SetTrigger</td>
<td>Sets the force trigger for the Till command.</td>
</tr>
</tbody>
</table>

**DB Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CloseDB</td>
<td>Close the database that has been opened with the OpenDB command and releases the file number.</td>
</tr>
<tr>
<td>DeleteDB</td>
<td>Deletes data from the table in the opened database.</td>
</tr>
<tr>
<td>OpenDB</td>
<td>Opens a database or Excel workbook.</td>
</tr>
<tr>
<td>SelectDB</td>
<td>Searches the data in the table in an opened database.</td>
</tr>
<tr>
<td>UpdateDB</td>
<td>Updates data of the table in the opened database.</td>
</tr>
</tbody>
</table>

**PG Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG_FastStop</td>
<td>Stops the PG axes immediately.</td>
</tr>
<tr>
<td>PG_LSpeed</td>
<td>Sets the pulse speed of the time when the PG axis starts accelerating and fishishes decelerating.</td>
</tr>
<tr>
<td>PG_Scan</td>
<td>Starts the continuous spinning motion of the PG robot axes.</td>
</tr>
<tr>
<td>PG_SlowStop</td>
<td>Stops slowly the PG axis spinning continuously.</td>
</tr>
</tbody>
</table>

**Collision Detection Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CollisionDetect</td>
<td>Enables or disables the collision detection.</td>
</tr>
<tr>
<td>CollisionDetect Function</td>
<td>Returns the setting value of CollisionDetect command.</td>
</tr>
</tbody>
</table>
## Parts Consumption Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HealthCalcPeriod</td>
<td>Set the calculation period of parts consumption commands.</td>
</tr>
<tr>
<td>HealthCalcPeriod Function</td>
<td>Returns the calculation period of parts consumption commands.</td>
</tr>
<tr>
<td>HealthCtrlAlarmOn Function</td>
<td>Returns the status of the parts consumption alarm for the specified Controller parts.</td>
</tr>
<tr>
<td>HealthCtrlInfo</td>
<td>Displays the remaining months before the recommended replacement time for the specified Controller parts.</td>
</tr>
<tr>
<td>HealthCtrlInfo Function</td>
<td>Returns the remaining months before the recommended replacement time for the specified Controller parts.</td>
</tr>
<tr>
<td>HealthCtrlRateOffset</td>
<td>Sets the offset for the consumption rate of the specified parts.</td>
</tr>
<tr>
<td>HealthCtrlReset</td>
<td>Clears the consumption rate for the specified Controller parts.</td>
</tr>
<tr>
<td>HealthCtrlWarningEnable</td>
<td>Sets enable or disable the parts consumption alarm notification of the Controller parts.</td>
</tr>
<tr>
<td>HealthCtrlWarningEnable Function</td>
<td>Returns enable or disable the parts consumption alarm notification of the controller part.</td>
</tr>
<tr>
<td>HealthRateCtrlInfo Function</td>
<td>Returns the consumption rate of the specified Controller parts.</td>
</tr>
<tr>
<td>HealthRateRBInfo Function</td>
<td>Returns the consumption rate for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBAlarmOn Function</td>
<td>Returns the status of the parts consumption alarm for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBAnalysis</td>
<td>Displays the analysis result regarding the parts consumption (remaining months before the recommended parts replacement time) for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBAnalysis Function</td>
<td>Returns the analysis result regarding the parts consumption (remaining months before the recommended parts replacement time) for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBDistance</td>
<td>Displays the driving amount of the specified joint.</td>
</tr>
<tr>
<td>HealthRBDistance Function</td>
<td>Returns the driving amount of the specified joint.</td>
</tr>
<tr>
<td>HealthRBInfo</td>
<td>Displays the remaining months before the recommended replacement time for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBInfo Function</td>
<td>Returns the remaining months before the recommended replacement time for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBRateOffset</td>
<td>Sets the offset for the consumption rate of the specified parts.</td>
</tr>
<tr>
<td>HealthRBRateOffset Function</td>
<td>Sets the offset for the consumption rate of the specified parts.</td>
</tr>
<tr>
<td>HealthRBReset</td>
<td>Clears the consumption rate for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBSpeed</td>
<td>Displays the average speed of the specified joint.</td>
</tr>
<tr>
<td>HealthRBSpeed Function</td>
<td>Returns the average of the absolute speed of the specified joint.</td>
</tr>
<tr>
<td>HealthRBStart</td>
<td>Starts analysis of the parts consumption for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBStop</td>
<td>Stops analysis of the parts consumption for the specified robot parts.</td>
</tr>
<tr>
<td>HealthRBTRQ</td>
<td>Displays the torque value of the specified joint.</td>
</tr>
<tr>
<td>HealthRBTRQ Function</td>
<td>Returns the torque value of the specified joint.</td>
</tr>
</tbody>
</table>
### Summary of SPEL+ Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HealthRBWarningEnable</strong></td>
<td>Sets enable or disable the parts consumption alarm notification of the robot parts.</td>
</tr>
<tr>
<td><strong>HealthRBWarningEnable Function</strong></td>
<td>Returns enable or disable the parts consumption alarm notification of the robot parts.</td>
</tr>
</tbody>
</table>

### Simulator Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SimSet</strong></td>
<td>Sets the object settings, operations, and robot motions of simulator.</td>
</tr>
<tr>
<td><strong>SimGet</strong></td>
<td>Acquires the setting values of simulator object.</td>
</tr>
</tbody>
</table>
This section describes each SPEL\(^{+}\) command as follows:

- **Syntax**: Syntax describes the format used for each command. For some commands, there is more than one syntax shown, along with a number that is referenced in the command description. Parameters are shown in italics.

- **Parameters**: Describes each of the parameters for this command.

- **Return Values**: Describes any values that the command returns.

- **Description**: Gives details about how the command works.

- **Note**: Gives additional information that may be important about this command.

- **See Also**: Shows other commands that are related to this command. Refer to the Table of Contents for the page number of the related commands.

- **Example**: Gives one or more examples of using this command.
### Operators

The following table shows the operators for the SPEL+ language.

<table>
<thead>
<tr>
<th>Keyword or Symbol</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>A+B</td>
<td>Addition</td>
</tr>
<tr>
<td>-</td>
<td>A-B</td>
<td>Subtraction</td>
</tr>
<tr>
<td>*</td>
<td>A*B</td>
<td>Multiplication</td>
</tr>
<tr>
<td>/</td>
<td>A/B</td>
<td>Division</td>
</tr>
<tr>
<td>**</td>
<td>A**B</td>
<td>Exponentiation</td>
</tr>
<tr>
<td>=</td>
<td>A=B</td>
<td>Equal</td>
</tr>
<tr>
<td>&gt;</td>
<td>A&gt;B</td>
<td>Greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>A&lt;B</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>A&gt;=B</td>
<td>Greater than or equal</td>
</tr>
<tr>
<td>&lt;=</td>
<td>A&lt;=B</td>
<td>Less or than equal</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>A&lt;&gt;B</td>
<td>Not equal</td>
</tr>
<tr>
<td>And</td>
<td>A And B</td>
<td>Performs logical and bitwise AND operation.</td>
</tr>
<tr>
<td>Mod</td>
<td>A Mod B</td>
<td>Returns the remainder obtained by dividing a numeric expression by another numeric expression.</td>
</tr>
<tr>
<td>Not</td>
<td>Not A</td>
<td>Performs logical or bitwise negation of the operand.</td>
</tr>
<tr>
<td>Or</td>
<td>A Or B</td>
<td>Performs the bitwise Or operation on the values of the operands.</td>
</tr>
<tr>
<td>Xor</td>
<td>A Xor B</td>
<td>Performs the bitwise Xor operation on the values of the operand.</td>
</tr>
</tbody>
</table>

### Priority Order of the Operators

The operators are processed in programs in the following order.

<table>
<thead>
<tr>
<th>Priority level</th>
<th>Operator</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>()</td>
<td>( A+B )</td>
<td>Brackets</td>
</tr>
<tr>
<td>2</td>
<td>**</td>
<td>A**B</td>
<td>Exponentiation</td>
</tr>
<tr>
<td>3</td>
<td>*</td>
<td>A*B</td>
<td>Multiplication</td>
</tr>
<tr>
<td></td>
<td>/</td>
<td>A/B</td>
<td>Division</td>
</tr>
<tr>
<td>4</td>
<td>Mod</td>
<td>A Mod B</td>
<td>Returns the remainder obtained by dividing a numeric expression by another numeric expression.</td>
</tr>
<tr>
<td>5</td>
<td>+</td>
<td>A+B</td>
<td>Addition</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>A-B</td>
<td>Subtraction</td>
</tr>
<tr>
<td>6</td>
<td>=</td>
<td>A=B</td>
<td>Equal</td>
</tr>
<tr>
<td></td>
<td>&lt;&gt;</td>
<td>A&lt;&gt;B</td>
<td>Not equal</td>
</tr>
<tr>
<td></td>
<td>&lt;</td>
<td>A&lt;B</td>
<td>Less than</td>
</tr>
<tr>
<td></td>
<td>&gt;</td>
<td>A&gt;B</td>
<td>Greater than</td>
</tr>
<tr>
<td></td>
<td>&lt;=</td>
<td>A&lt;=B</td>
<td>Less or than equal</td>
</tr>
<tr>
<td></td>
<td>&gt;=</td>
<td>A&gt;=B</td>
<td>Greater than or equal</td>
</tr>
<tr>
<td>7</td>
<td>Not</td>
<td>Not A</td>
<td>Performs logical or bitwise negation of the operand.</td>
</tr>
<tr>
<td>8</td>
<td>And</td>
<td>A And B</td>
<td>Performs logical and bitwise AND operation.</td>
</tr>
<tr>
<td>9</td>
<td>Or</td>
<td>A Or B</td>
<td>Performs the bitwise Or operation on the values of the operands.</td>
</tr>
<tr>
<td>10</td>
<td>Xor</td>
<td>A Xor B</td>
<td>Performs the bitwise Xor operation on the values of the operand.</td>
</tr>
</tbody>
</table>
Parallel Processing

Processes input/output statements in parallel with motion.

Syntax

```
motion cmd !statements!
```

Parameters

- **motion cmd**: Any valid motion command included in the following list: Arc, Arc3, Go, Jump, Jump3, Jump3CP, Move, BGo, BMove, TGo, TMove.
- **statements**: Any valid parallel processing I/O statement(s) which can be executed during motion. (See the table below)

Description

Parallel processing commands are attached to motion commands to allow I/O statements to execute simultaneously with the beginning of motion travel. This means that I/O can execute while the arm is moving rather than always waiting for arm travel to stop and then executing I/O. There is even a facility to define when within the motion that the I/O should begin execution. (See the “Dn” parameter described in the table below.)

The table below shows all valid parallel processing statements. Each of these statements may be used as single statements or grouped together to allow multiple I/O statements to execute during one motion statement.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dn</td>
<td>Used to specify %travel before the next parallel statement is executed. “n” is a percentage between 0 and 100 which represents the position within the motion where the parallel processing statements should begin. Statements which follow the Dn parameter will begin execution after n% of the motion travel has been completed. When used with the Jump, Jump3, and Jump3CP commands, %travel does not include the depart and approach motion. To execute statements after the depart motion has completed, include D0 (zero) at the beginning of the statement. “Dn” may appear a maximum of 16 times in a parallel processing statement.</td>
</tr>
<tr>
<td>On / Off n</td>
<td>Turn Output bit number “n” on or off.</td>
</tr>
<tr>
<td>MemOn / MemOff n</td>
<td>Turns memory I/O bit number “n” on or off.</td>
</tr>
<tr>
<td>Out p,d</td>
<td>Outputs data “d” to output port “p”.</td>
</tr>
<tr>
<td>OpBCD p,q</td>
<td>Outputs data “d” to memory I/O port “p”.</td>
</tr>
<tr>
<td>OutW p,d</td>
<td>Outputs data “d” to memory I/O port “p”.</td>
</tr>
<tr>
<td>Signal s</td>
<td>Generates synchronizing signal.</td>
</tr>
<tr>
<td>Wait t</td>
<td>Delays for “t” seconds prior to execution of the next parallel processing statement.</td>
</tr>
<tr>
<td>WaitSig s</td>
<td>Waits for signal “s” before processing next statement.</td>
</tr>
<tr>
<td>Wait Sw(n) = j</td>
<td>Delays execution of next parallel processing statement until the input bit “n” is equal to the condition defined by “j”. (On or Off)</td>
</tr>
<tr>
<td>Wait MemSw(n) = j</td>
<td>Delays execution of the next parallel processing statement until the memory I/O bit “n” is equal to the condition defined by “j”. (On or Off)</td>
</tr>
<tr>
<td>Wait other conditions</td>
<td>Wait other than the above two patterns is available. Refer to Wait Statement for details.</td>
</tr>
<tr>
<td>Print</td>
<td>Prints data to the display device.</td>
</tr>
<tr>
<td>Print #</td>
<td>Prints data to the specified communications port.</td>
</tr>
<tr>
<td>External functions</td>
<td>Executes the external functions declared with Declare statement.</td>
</tr>
</tbody>
</table>
Notes

When Motion is Completed before All I/O Commands are Complete
If, after completing the motion for a specific motion command, all parallel processing statement execution has not been completed, subsequent program execution is delayed until all parallel processing statements execution has been completed. This situation is most likely to occur with short moves with many I/O commands to execute in parallel.

When the Till statement is used to stop the arm before completing the intended motion
If Till is used to stop the arm at an intermediate travel position, the system considers that the motion is completed. The next statement execution is delayed until the execution of all parallel processing statements has been completed.

When the AbortMotion statement or Trap is used to stop the arm before completing the motion
After the arm stops at an intermediate travel position, D statement cannot be executed.

Specifying “n” near 100% can cause path motion to decelerate
If a large value of “n” is used during CP motion, the robot may decelerate to finish the current motion. This is because the position specified would normally be during deceleration if CP was not being used. To avoid deceleration, consider placing the processing statement after the motion command. For example, in the example below, the On 1 statement is moved from parallel processing during the jump to P1 to after the jump.

```plaintext
CP On
Jump P1 !D96; On 1!
Go P2
```

The Jump statement and Parallel Processing
It should be noted that execution of parallel processing statements which are used with the Jump statement begins after the rising motion has completed and ends at the start of falling motion.

The Here statement and Parallel Processing
You cannot use both of the Here statement and parallel processing in one motion command like this:

```plaintext
Go Here :Z(0) ! D10; MemOn 1 !
```
Be sure to change the program like this:

```plaintext
P999 = Here
Go P999 Here :Z(0) ! D10; MemOn 1 !
```

See Also
Arc, Arc3, Go, Jump, Jump3, Jump3CP, Move, BGo, BMove, TGo, TMove

I...I Parallel Processing Example
The following examples show various ways to use the parallel processing feature with Motion Commands:

Parallel processing with the Jump command causes output bit 1 to turn on at the end of the Z joint rising travel and when the 1st, 2nd, and 4th axes begin to move. Then output bit 1 is turned off again after 50% of the Jump motion travel has completed.

```plaintext
Function test
    Jump P1 !D0; On 1; D50; Off 1!
Fend
```

Parallel processing with the Move command causes output bit 5 to turn on when the joints have completed 10% of their move to the point P1. Then 0.5 seconds later turn output bit 5 off.

```plaintext
Function test2
    Move P1 !D10; On 5; Wait 0.5; Off 5!
Fend
```
#define

Defines identifier to be replaced by specified replacement string.

Syntax

```
#define identifier [ ( parameter [, parameter ] ) ] string
```

Parameters

- **identifier**: Keyword defined by user which is an abbreviation for the *string* parameter. Rules for identifiers are as follows:
  - The first character must be alphabetic while the characters which follow may be alphanumeric or an underscore (_).
  - Spaces or tab characters are not allowed as part of the *identifier*.

- **parameter**: Normally used to specify a variable (or multiple variables) which may be used by the replacement string. This provides for a dynamic define mechanism which can be used like a macro. A maximum of up to 8 parameters may be used with the #define command. However, each parameter must be separated by a comma and the parameter list must be enclosed within parenthesis.

- **string**: This is the replacement string which replaces the identifier when the program is compiled. Rules regarding replacement strings are as follows:
  - Spaces or tabs are allowed in replacement strings.
  - Identifiers used with other #define statements cannot be used as replacement strings.
  - If the comment symbol (') is included, the characters following the comment symbol will be treated as a comment and will not be included in the replacement string.
  - The replacement string may be omitted. In this case the specified identifier is replaced by "nothing" or the null string. This actually deletes the identifier from the program.

Description

The #define instruction causes a replacement to occur within a program for the specified identifier. Each time the specified identifier is found the identifier is replaced with the replacement string prior to compilation. However, the source code will remain with the identifier rather than the replacement string. This allows code to become easier to read in many cases by using meaningful identifier names rather than long difficult to read strings of code.

The defined identifier can be used for conditional compiling by combining with the #ifdef or #ifndef commands.

If a parameter is specified, the new identifier can be used like a macro.

Note

*Using #define for variable declaration or label substitutions will cause an error:*

It should be noted that usage of the #define instruction for variable declaration will cause an error.

See Also

- #ifdef, #ifndef
#define Example

' Uncomment next line for Debug mode.
' #define DEBUG

Input #1, A$
#ifdef DEBUG
    Print "A$ = ", A$
#endif
Print "The End"

#define SHOWVAL(x) Print "var = ", x

Integer a
a = 25
SHOWVAL(a)
#ifdef...#else...#endif

Provides conditional compiling capabilities.

Syntax

```
#ifdef identifier
..put selected source code for conditional compile here.
#else
..put selected source code for false condition here.
#endif
```

Parameters

`identifier`  Keyword defined by the user which when defined allows the source code defined between #ifdef and #else or #endif to be compiled. Thus the identifier acts as the condition for the conditional compile.

Description

#ifdef...#else...#endif allows for the conditional compiling of selected source code. The condition as to whether or not the compile will occur is determined based on the identifier. #ifdef first checks if the specified identifier is currently defined by #define. The #else statement is optional.

If defined, and the #else statement is not used, the statements between #ifdef and #endif are compiled. Otherwise, if #else is used, then the statements between #ifdef and #else are compiled.

If not defined, and the #else statement is not used, the statements between #ifdef and #endif are skipped without being compiled. Otherwise, if #else is used, then the statements between #else and #endif are compiled.

See Also

#define, ifndef

#ifdef Example

A section of code from a sample program using #ifdef is shown below. In the example below, the printing of the value of the variable A$ will be executed depending on the presence or absence of the definition of the #define DEBUG pseudo instruction. If the #define DEBUG pseudo instruction was used earlier in this source, the Print A$ line will be compiled and later executed when the program is run. However, the printing of the string "The End" will occur regardless of the #define DEBUG pseudo instruction.

```
' Uncomment next line for Debug mode.
' #define DEBUG

Input #1, A$
#ifdef DEBUG
    Print "A$ = ", A$
#endif
Print "The End"
```
#ifndef...#endif

Provides conditional compiling capabilities.

Syntax

```c
#ifndef  identifier
   // Put selected source code for conditional compile here.
#else
   // ...put selected source code for true condition here.]
#endif
```

Parameters

- `identifier`: Keyword defined by the user which when not defined allows the source code defined between `#ifndef` and `#else` or `#endif` to be compiled. Thus the identifier acts as the condition for the conditional compile.

Description

This instruction is called the "if not defined" instruction. `#ifndef...#else...#endif` allow for the conditional compiling of selected source code. The `#else` statement is optional.

- If defined, and the `#else` statement is not used, the statements between `#ifndef` and `#endif` are not compiled. Otherwise, if `#else` is used, then the statements between `#else` and `#endif` are compiled.

- If not defined, and the `#else` statement is not used, the statements between `#ifndef` and `#endif` are compiled. Otherwise, if `#else` is used, then the statements between `#else` and `#endif` are not compiled.

Note

**Difference between `#ifdef` and `#ifndef`**

The fundamental difference between `#ifdef` and `#ifndef` is that the `#ifdef` instruction compiles the specified source code if the identifier is defined. The `#ifndef` instruction compiles the specified source code if the identifier is not defined.

See Also

- `#define`, `#ifdef`

 ifndef Example

A section of code from a sample program using `#ifndef` is shown below. In the example below, the printing of the value of the variable $A$ will be executed depending on the presence or absence of the definition of the `#define NODELAY` pseudo instruction. If the `#define NODELAY` pseudo instruction was used earlier in this source, the Wait 1 line will **not be compiled** along with the rest of the source for this program when it is compiled. (i.e. submitted for running.) If the `#define NODELAY` pseudo instruction was not used (i.e. `NODELAY` is not defined) earlier in this source, the Wait 1 line will **be compiled** and later executed when the program is run. The printing of the string "The End" will occur regardless of the `#define NODELAY` pseudo instruction.

```c
' Comment out next line to force delays.
#define NODELAY 1

Input #1, A$
#ifndef NODELAY
 Wait 1
#else
#endif
Print "The End"
```
#include

Includes the specified file into the file where the #include statement is used.

Syntax

```
#include "fileName.INC"
```

Parameters

- `fileName` must be the name of an include file in the current project. All include files have the ".inc" extension. The filename specifies the file which will be included in the current file.

Description

#include inserts the contents of the specified include file with the current file where the #include statement is used.

Include files are used to contain #define statements and global variable declarations.

The #include statement must be used outside of any function definitions.

An include file may contain a secondary include file. For example, FILE2 may be included within FILE1, and FILE3 may be included within FILE2. This is called nesting.

See Also

#define, ifdef, ifndef

#include Example

Include File (Defs.inc)

```
#define DEBUG 1
#define MAX_PART_COUNT 20
```

Program File (main.prg)

```
#include "defs.inc"

Function main
    Integer i

    Integer Parts(MAX_PART_COUNT)

Fend
```
#undef

Undefines an identifier previously defined with #define.

Syntax
#undef identifier

Parameters
identifier  Keyword used in a previous #define statement.

See Also
#define, ifndef, ifndef
AbortMotion Statement

AbortMotion Statement

Aborts a motion command and puts the running task in error status.
This command is for the experienced user and you need to understand the command specification before use.

Syntax

AbortMotion {robotNumber | All}

Parameters

robotNumber Robot number that you want to stop the motion for.
All Aborts motion for all robots.

Description

Depending on the robot status when AbortMotion is executed, the result is different as follows.
In each case, hook an error and handle the error processing with OnErr to continue the processing.
Error 2999 can use the constant ERROR_DOINGMOTION.
Error 2998 can use the constant ERROR_NOMOTION.

Write a program not to execute AbortMotion more than twice before executing the continuous execution (Cont).

When the robot is executing the motion command
The robot promptly pauses the arm motion immediately and cancels the remaining motions.
Error 2999 (ERROR_DOINGMOTION) occurs in the task which was running the motion command for the robot.
For the following motion commands, the robot directly moves to the next position from the point where it was paused.

When the robot has been paused immediately
When AbortMotion is executed, the remaining motion is canceled.
Error 2999 (ERROR_DOINGMOTION) occurs in the task which was running the motion command for the robot when specifying the Cont statement.
For the following motion commands, the robot directly moves to the next position from the point where it was paused.

When the robot is in WaitRecover status (Safeguard Open)
When AbortMotion is executed, the remaining motion is canceled.
The following motions can be selected with the Recover command flags.

When executing “Recover robotNumber, WithMove”, the robot motors turn on and the recovery motion is executed.
When Cont is executed, error 2999 (ERROR_DOINGMOTION) occurs in the task which was running the motion command for the robot.
For the following motion commands, the robot directly moves to the next position from the point where it was paused.

When executing “Recover robotNumber, WithoutMove”, the robot motors turn on.
When Cont is executed, error 2999 (ERROR_DOINGMOTION) occurs in the task which was running the motion command for the robot.
For the following motion commands, the robot directly moves to the next position from the point where it was paused, without the recovery motion.
When the robot is executing commands other than motion commands
Error 2998 (ERROR_NOMOTION) occurs in the task which was previously running the motion command for the robot. When the task is waiting with Wait or Input commands, the task is aborted promptly and error 2998 occurs.
When executing a motion command with CP On and a program has no more motion commands, error 2998 occurs even if the robot is running.

When the robot is not running from a program (task)
An error occurs.

See Also
OnErr, Recover, Till

AbortMotion Statement Example
When memory I/O #0 turns on, AbortMotion is executed and the robot goes back to the home position.

```
Function main
  Motor On
  Xqt sub, NoEmgAbort
  OnErr GoTo errhandle

  Go P0
  Wait Sw(1)
  Go P1

  Quit sub
  Exit Function

errstart:
  Home
  Quit sub
  Exit Function

errhandle:
  Print Err
  If Err = ERROR_DOINGMOTION Then
    Print "Robot is moving" ' Executing Go P0 or Go P1
    ER esume errstart
  ElseIf Err = ERROR_NOMOTION Then
    Print "Robot is not moving " ' Executes Wait Sw(1)
    ER esume errstart
  EndIf

  Print "Error Stop" ' Other error occurs
  Quit All
Fend

Function sub
  MemOff 0
  Wait MemSw(0)
  AbortMotion 1
  MemOff 0
Fend
```
Abs Function

Returns the absolute value of a number.

Syntax

Abs(number)

Parameters

number Any valid numeric expression.

Return Values

The absolute value of a number.

Description

The absolute value of a number is its unsigned magnitude. For example, Abs(-1) and Abs(1) both return 1.

See Also

Atan, Atan2, Cos, Int, Mod, Not, Sgn, Sin, Sqr, Str$, Tan, Val

Abs Function Example

The following examples are done from the command window using the Print instruction.

```
> print abs(1)
1
> print abs(-1)
1
> print abs(-3.54)
3.54
> 
```
Accel Statement

Sets (or displays) the acceleration and deceleration rates for the point to point motion instructions Go, Jump and Pulse.

Syntax

(1) **Accel**  *accel*, *decel* [, *departAccel*, *departDecel*, *approAccel*, *approDecel*]

(2) **Accel**

Parameters

- **accel**: Integer expression 1 or more representing a percentage of maximum acceleration rate.
- **decel**: Integer expression 1 or more representing a percentage of the maximum deceleration rate.
- **departAccel**: Depart acceleration for Jump. Valid Entries are 1 or more. Optional. Available only with Jump command.
- **departDecel**: Depart deceleration for Jump. Valid Entries are 1 or more. Optional. Available only with Jump command.
- **approAccel**: Approach acceleration for Jump. Valid Entries are 1 or more. Optional. Available only with Jump command.
- **approDecel**: Approach deceleration for Jump. Valid Entries are 1 or more. Optional. Available only with Jump command.

Return Values

When parameters are omitted, the current Accel parameters are displayed.

Description

Accel specifies the acceleration and deceleration for all Point to Point type motions. This includes motion caused by the Go, Jump and Pulse robot motion instructions.

Each acceleration and deceleration parameter defined by the Accel instruction may be an integer value 1 or more. This number represents a percentage of the maximum acceleration (or deceleration) allowed. Usually, the maximum value is 100. However, some robots allow setting larger than 100. Use AccelMax function to get the maximum value available for Accel.

The Accel instruction can be used to set new acceleration and deceleration values or simply to print the current values. When the Accel instruction is used to set new accel and decel values, the first 2 parameters (*accel* and *decel*) in the Accel instruction are required.

The optional *departAccel*, *departDecel*, *approAccel*, and *approDecel* parameters are effective for the Jump instruction only and specify acceleration and deceleration values for the depart motion at the beginning of Jump and the approach motion at the end of Jump.

The Accel value initializes to the default values (low acceleration) when any one of the following conditions occurs:

```
Controller Startup
Motor On
SFree, SLock, Brake
Reset, Reset Error
Stop button or QuitAll stops tasks
```
Notes

Executing the Accel command in Low Power Mode (Power Low)

If Accel is executed when the robot is in low power mode (Power Low), the new values are stored, but the current values are limited to low values.
The current acceleration values are in effect when Power is set to High, and Teach mode is OFF.

Accel vs. AccelS

It is important to note that the Accel instruction does not set the acceleration and deceleration rates for straight line and arc motion. The AccelS instruction is used to set the acceleration and deceleration rates for the straight line and arc type moves.

Accel setting larger than 100

Usually, the maximum value is 100. However, some robots allow setting larger than 100.
In general use, Accel setting 100 is the optimum setting that maintains the balance of acceleration and vibration when positioning. However, you may require an operation with high acceleration to shorten the cycle time by decreasing the vibration at positioning. In this case, set the Accel to larger than 100. Except in some operation conditions, the cycle time may not change by setting Accel to larger than 100.

See Also

AccelR, AccelS, Go, Jump, Jump3, Power, Pulse, Speed, TGo

Accel Statement Example

The following example shows a simple motion program where the acceleration (Accel) and speed (Speed) is set using predefined variables.

<Example 1>

Function acctest
  Integer slow, accslow, decslow, fast, accfast, decfast

  slow = 20  'set slow speed variable
  fast = 100 'set high speed variable
  accslow = 20 'set slow acceleration variable
  decslow = 20 'set slow deceleration variable
  accfast = 100 'set fast acceleration variable
  decfast = 100 'set fast deceleration variable

  Accel accslow, decslow
  Speed slow
  Jump pick
  On gripper
  Accel accfast, decfast
  Speed fast
  Jump place
  .
  .
  Fend
<Example 2>
Set the Z joint downward deceleration to be slow to allow a gentle placement of the part when using the Jump instruction. This means we must set the Zdnd parameter low when setting the Accel values.

>Accel 100,100,100,100,100,35

>Accel
100  100
100  100
100   35

>
Accel Function

Returns specified acceleration value.

Syntax

```
Accel(paramNumber)
```

Parameters

- `paramNumber`: Integer expression which can have the following values:
  1: acceleration specification value
  2: deceleration specification value
  3: depart acceleration specification value for Jump
  4: depart deceleration specification value for Jump
  5: approach acceleration specification value for Jump
  6: approach deceleration specification value for Jump

Return Values

Integer 1% or more

See Also

- Accel Statement

Accel Function Example

This example uses the Accel function in a program:

```plaintext
Integer currAccel, currDecel

' Get current accel and decel
currAccel = Accel(1)
currDecel = Accel(2)
Accel 50, 50
SRVJump pick
' Restore previous settings
Accel currAccel, currDecel
```
AccelMax Function

Returns maximum acceleration value limit available for Accel.

Syntax

\texttt{AccelMax(maxValueNumber)}

Parameters

\textit{maxValueNumber} Integer expression which can have the following values:

1: acceleration maximum value
2: deceleration maximum value
3: depart acceleration maximum value for Jump
4: depart deceleration maximum value for Jump
5: approach acceleration maximum value for Jump
6: approach deceleration maximum value for Jump

Return Values

Integer 1% or more

See Also

Accel

AccelMax Function Example

This example uses the AccelMax function in a program:

\begin{verbatim}
' Get maximum accel and decel
Print \texttt{AccelMax}(1), \texttt{AccelMax}(2)
\end{verbatim}
AccelR Statement

Sets or displays the acceleration and deceleration values for tool rotation control of CP motion.

Syntax

(1) **AccelR**  \( \text{accel} [, \text{decel}] \)
(2) **AccelR**

Parameters

\( \text{accel} \) Real expression in degrees / second\(^2\) (0.1 to 5000).

\( \text{decel} \) Real expression in degrees / second\(^2\) (0.1 to 5000).

Valid entries range of the parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>VT6L</td>
<td>0.1 to 1000</td>
</tr>
<tr>
<td>C4, C8, C12, N2, N6 T series, G series, RS series LS series, LS-B series X5</td>
<td>0.1 to 5000</td>
</tr>
</tbody>
</table>

Return Values

When parameters are omitted, the current AccelR settings are displayed.

Description

AccelR is effective when the ROT modifier is used in the Move, Arc, Arc3, BMove, TMove, and Jump3CP motion commands.

The AccelR value initializes to the default values when any one of the following conditions occurs:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

See Also

Arc, Arc3, BMove, Jump3CP, Power, SpeedR, TMove

**AccelR Statement Example**

\[ \text{AccelR 360, 200} \]
AccelR Function

Returns specified tool rotation acceleration value.

Syntax

```
AccelR(paramNumber)
```

Parameters

- `paramNumber`  
  Integer expression which can have the following values:
  1: acceleration specification value
  2: deceleration specification value

Return Values

- Real value in degrees / second²

See Also

- AccelR Statement

AccelR Function Example

```
Real currAccelR, currDecelR

' Get current accel and decel
currAccelR = AccelR(1)
currDecelR = AccelR(2)
```
AccelS Statement

Sets the acceleration and deceleration rates for the Straight Line and Continuous Path robot motion instructions such as Move, Arc, Arc3, Jump3, CVMove, etc.

Syntax

(1) **AccelS**  \( \text{accel} [, \text{decel}] [, \text{departAccel}, \text{departDecel}, \text{approAccel}, \text{approDecel}] \)

(2) **AccelS**

Parameters

- **accel**: Real expression represented in mm/sec\(^2\) units to define acceleration and deceleration values for straight line and continuous path motion. If **decel** is omitted, then **accel** is used to specify both the acceleration and deceleration rates.

- **decel**: Optional. Real expression represented in mm/sec\(^2\) units to define the deceleration value.

- **departAccel**: Optional. Real expression for depart acceleration value for Jump3, Jump3CP.

- **departDecel**: Optional. Real expression for depart deceleration value for Jump3, Jump3CP.

- **approAccel**: Optional. Real expression for approach acceleration value for Jump3, Jump3CP.

- **approDecel**: Optional. Real expression for approach deceleration value for Jump3, Jump3CP.

Valid entries range of the parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>(mm/sec(^2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{accel} )</td>
<td>0.1 to 5000</td>
</tr>
<tr>
<td>( \text{decel} )</td>
<td>0.1 to 10000</td>
</tr>
<tr>
<td>( \text{departAccel} )</td>
<td>0.1 to 15000</td>
</tr>
<tr>
<td>( \text{departDecel} )</td>
<td>0.1 to 25000</td>
</tr>
<tr>
<td>( \text{approAccel} )</td>
<td>0.1 to 35000</td>
</tr>
<tr>
<td>( \text{approDecel} )</td>
<td>0.1 to 35000</td>
</tr>
</tbody>
</table>

Return Values

Displays Accel and Decel values when used without parameters

Description

AccelS specifies the acceleration and deceleration for all interpolated type motions including linear and curved interpolations. This includes motion caused by the Move and Arc motion instructions.

The AccelS value initializes to the default values when any one of the following conditions occurs:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks
Notes

Executing the AccelS command in Low Power Mode (Power Low):

If AccelS is executed when the robot is in low power mode (Power Low), the new values are stored, but the current values are limited to low values.

The current acceleration values are in effect when Power is set to High, and Teach mode is OFF.

Accel vs. AccelS:

It is important to note that the AccelS instruction does not set the acceleration and deceleration rates for point to point type motion. (i.e. motions initiated by the Go, Jump, and Pulse instructions.) The Accel instruction is used to set the acceleration and deceleration rates for Point to Point type motion.

Upper limit value

The AccelS upper limit value varies depending on Weight setting and the position of the spline unit. For details, refer to the Manipulator manuals.

See Also

Accel, Arc, Arc3, Jump3, Jump3CP, Power, Move, TMove, SpeedS

AccelS Statement Example

The following example shows a simple motion program where the straight line/continuous path acceleration (AccelS) and straight line/continuous path speed (SpeedS) are set using predefined variables.

```
Function acctest
    Integer slow, accslow, fast, accfast

    slow = 20        'set slow speed variable
    fast = 100       'set high speed variable
    accslow = 200    'set slow acceleration variable
    accfast = 5000   'set fast acceleration variable

    AccelS accslow
    SpeedS slow
    Move P1
    On 1
    AccelS accfast
    SpeedS fast
    Jump P2
    .
    .

Fend
```
AccelS Function

Returns acceleration or deceleration for CP motion commands.

Syntax

\[ \text{AccelS}(\text{paramNumber}) \]

Parameters

\( \text{paramNumber} \) Integer expression which can have the following values:

1: acceleration value
2: deceleration value
3: depart acceleration value for Jump3, Jump3CP
4: depart deceleration value for Jump3, Jump3CP
5: approach acceleration value for Jump3, Jump3CP
6: approach deceleration value for Jump3, Jump3CP

Return Values

Real value from 0 to 5000 mm/sec/sec

See Also

AccelS Statement, Arc3, SpeedS, Jump3, Jump3CP

AccelS Function Example

```
Real savAccelS

savAccelS = AccelS(1)
```
Acos Function

Returns the arccosine of a numeric expression.

Syntax

Acos(number)

Parameters

number  Numeric expression representing the cosine of an angle.

Return Values

Real value, in radians, representing the arccosine of the parameter number.

Description

Acos returns the arccosine of the numeric expression. Values range is from -1 to 1. The value returned by Acos will range from 0 to PI radians. If number is < -1 or > 1, an error occurs.

To convert from radians to degrees, use the RadToDeg function.

See Also

Abs, Asin, Atan, Atan2, Cos, DegToRad, RadToDeg, Sgn, Sin, Tan, Val

Acos Function Example

Function acostest
    Double x

    x = Cos(DegToRad(30))
    Print "Acos of ", x, " is ", Acos(x)
Fend
Agl Function

Returns the joint angle for the selected rotational joint, or position for the selected linear joint.

Syntax

\[
\text{Agl}(\text{jointNumber})
\]

Parameters

\textit{jointNumber} \quad \text{Integer expression representing the joint number. Values are from 1 to the number of joints on the robot. The additional S axis is 8 and T axis is 9.}

Return Values

The joint angle for selected rotational joint or position for selected linear joints.

Description

The Agl function is used to get the joint angle for the selected rotational joint or position for the selected linear joint.

If the selected joint is rotational, Agl returns the current angle, as measured from the selected joint's 0 position, in degrees. The returned value is a real number.

If the selected joint is a linear joint, Agl returns the current position, as measured from the selected joint's 0 position, in mm. The returned value is a real number.

If an auxiliary arm is selected with the Arm statement, Agl returns the angle (or position) from the standard arm's 0 pulse position to the selected arm.

See Also

PAgl, Pls, PPls

Agl Function Example

The following examples are done from the command window using the Print instruction.

\[
> \text{print agl}(1), \text{agl}(2) \\
17.234 \quad 85.355
\]
AglToPls Function

Converts robot angles to pulses.

Syntax

\[
\text{AglToPls}(j1, j2, j3, j4[, j5[, j6[, j7[, j8, j9]]]])
\]

Parameters

\( j1 \) - \( j6 \)  
Real expressions representing joint angles.

\( j7 \)  
Real expression representing the joint #7 angle. For the Joint type 7-axis robot.

\( j8 \)  
Real expression representing the additional S axis angle.

\( j9 \)  
Real expression representing the additional T axis angle.

Return Values

A robot point whose location is determined by joint angles converted to pulses.

Description

Use AglToPls to create a point from joint angles.

Note

Assignment to point can cause part of the joint position to be lost.

In certain cases, when the result of AglToPls is assigned to a point data variable, the arm moves to a joint position that is different from the joint position specified by AglToPls.

For example:

\[
P1 = \text{AglToPls}(0, 0, 0, 90, 0, 0) \\
\text{Go P1} \; \text{moves to AglToPls}(0, 0, 0, 0, 0, 90) \text{ joint position}
\]

Similarly, when the AglToPls function is used as a parameter in a CP motion command, the arm may move to a different joint position from the joint position specified by AglToPls.

\[
\text{Move AglToPls}(0, 0, 0, 90, 0, 0) \; \text{moves to AglToPls}(0, 0, 0, 0, 0, 90) \text{ joint position}
\]

When using the AglToPls function as a parameter in a PTP motion command, this problem does not occur.

See Also

Agl, JA, Pls

AglToPls Function Example

\[
\text{Go AglToPls}(0, 0, 0, 90, 0, 0)
\]
**AIO_In Function**

Reads analog value form optional analog I/O input channel.

**Syntax**

\[ \text{AIO_In(Channel Number)} \]

**Parameters**

- Channel Number: Specify the channel number of the analog I/O.

**Return Values**

Return the analog input value of the analog I/O channel which specified in channel number in real number. Return value range differs depending on the input range configuration of the analog I/O board.

**Description**

InFunction

**See Also**

- AIO_InWFunctio, AIO_Out, AIO_OutW, AIO_OutFunction, AIO_OutWFunctio, AIO_Set, Wait

**AIO_In Function Example**

```plaintext
Function main
    Real var1
    var1 = AIO_In(2)  'Acquires input state of analog channel input 2
    If var1 > 5.0 Then
        Go P1
        Go P2
        'Execute other motion command here
        '
        '
    Else
        Print "Error in initialization!"
        Print "Sensory Inputs not ready for cycle start"
        Print "Please check analog inputs 2."
    EndIf
Endfend
```
AIO_InW Function

Reads analog value from optional analog I/O input channel.

Syntax
AIO_InW(Channel Number)

Parameters
Channel Number Specify the channel number of the analog I/O.

Return Values
Returns the input states (long integers from 0 to 65535) of specified analog I/O channel.

The following table shows input voltage (current) and return value of each input channel according to input range configuration of analog I/O board.

<table>
<thead>
<tr>
<th>Input Data</th>
<th>Input Range Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hexadecimal</td>
<td>Decimal</td>
</tr>
<tr>
<td>0xFFFF</td>
<td>65535</td>
</tr>
<tr>
<td>0x8001</td>
<td>32769</td>
</tr>
<tr>
<td>0x8000</td>
<td>32768</td>
</tr>
<tr>
<td>0x0000</td>
<td>0</td>
</tr>
</tbody>
</table>

See Also
AIO_InFunction, AIO_Out, AIO_OutW, AIO_OutFunction, AIO_OutWFunction, AIO_Set, Wait

AIO_In Function Example

```plaintext
Long word0

word0 = AIO_InW(1)
```
**AIO_Out**

Output analog value from the optional analog I/O output channel.

**Syntax**

\[
\text{AIO\_Out}\ Channel\ Number,\ Outputdata\ [,\ Forced]\n\]

**Parameters**

- **Channel Number**: Specify the channel number of the analog I/O.
- **Output data**: Specify the real number of Real type which indicates output voltage [V] or current value [mA] in formula or value.
- **Forced**: Optional. Usually omitted.

**Description**

Output the Real value indicating specified voltage [V] or current [mA] to analog output port which specified on channel port. Set the voltage output range of analog output port or selection of voltage and current output by the switch on the board. If setting a value which out of range of analog I/O port, output the border value (maximum and minimum value) which is not out of the range.

AIO_Out command becomes an error if outputting the speed information by specified channel. Stop the speed information output and execute the AIO_Out command.

**Note**

**Forced Flag**

Specify the flag if outputting the analog I/O when operating emergency stop or opening the Safety Door by NoPause task and NoEmgAbort task (special task specified NoPause or NoEmgAbort to start when executing Xqt).

Need to be careful about the system design since analog I/O output changes when operating emergency stop or opening the Safety Door.

**See Also**

AIO_In, AIO_InFunction, AIO_OutW, AIO_OutFunction, AIO_OutWFunction, AIO_Set

**AIO_Out Example**

Output 7.0 [V] from the analog I/O channel 1.

\[
\text{AIO\_Out}\ 1,\ 7.0
\]
AIO_Out Function

Returns analog value in real number which is outputting in optional analog I/O output channel.

Syntax
AIO_Out(Channel Number)

Parameters
Channel Number Specify the channel number of the analog I/O.

Return Values
Returns specified analog I/O channel voltage and current output state in real number. Unit of voltage output is [V] and current output is [mA]. This function is available when outputting the speed information of the robot on specified channel.

See Also
AIO_In, AIO_InFunction, AIO_Out, AIO_OutW, AIO_OutWFunction, AIO_Set, Wait

AIO_Out Function Example

Real rdata01
rdata01 = AIO_Out(1)
AIO_OutW

Output 16 bits analog value from optional analog I/O output channel.

Syntax

\texttt{AIO\_OutW Channel Number, Output data [, Forced]} \\

Parameters

- **Channel Number**: Specify the channel number of the analog I/O.
- **Output data**: Specify the output data (Integer expression from 0 to 65535) in formula or value.
- **Forced**: Optional. Usually omitted.

Description

Output to analog I/O channel specified by channel number.

For the output data, specify integer expression from 0 to 65535 in formula or value.

Output voltage (current) is as follows according to output range configuration which is set by the switch on the board.

<table>
<thead>
<tr>
<th>Output Data</th>
<th>Output Range Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hexadecimal</td>
<td>Decimal</td>
</tr>
<tr>
<td>0xFFF</td>
<td>65535</td>
</tr>
<tr>
<td>0x8001</td>
<td>32769</td>
</tr>
<tr>
<td>0x8000</td>
<td>32768</td>
</tr>
<tr>
<td>0x0000</td>
<td>0</td>
</tr>
</tbody>
</table>

Note

**Forced Flag**

Specify the flag if outputting the analog I/O when operating emergency stop or opening the Safety Door by NoPause task, NoEmgAbort task (special task specified NoPause or NoEmgAbort to start when executing Xqt), and background task.

Need to be careful about the system design since analog I/O output changes when operating emergency stop or opening the Safety Door.

See Also

AIO_In, AIO_InFunction, AIO_Out, AIO_OutFunction, AIO_OutWFunction, AIO_Set, Wait

AIO_OutW Example

\texttt{AIO\_OutW 1, &H8000}
AIO_OutW Function

Returns output analog value in Long integers from 0 to 65535 which is output on optional analog I/O channel.

Syntax

AIO_OutW(Channel Number)

Parameters

Channel Number Specify the channel number of the analog I/O.

Return Values

Returns the output state of specified analog I/O channel in Long integers from 0 to 65535.

The following table shows output voltage (current) and return value of each output channel according to output range configuration of analog I/O board.

<table>
<thead>
<tr>
<th>Output Data</th>
<th>Output Range Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hexadecimal</td>
<td>Decimal</td>
</tr>
<tr>
<td>0xFFFF</td>
<td>65535</td>
</tr>
<tr>
<td>0x8001</td>
<td>32769</td>
</tr>
<tr>
<td>0x8000</td>
<td>32768</td>
</tr>
<tr>
<td>0x0000</td>
<td>0</td>
</tr>
</tbody>
</table>

This function is available when outputting the speed information of the robot on specified channel.

See Also

AIO_In, AIO_InFunction, AIO_Out, AIO_OutW, AIO_OutFunction, AIO_Set, Wait

AIO_OutW Function Example

```c
Long word0

word0 = AIO_OutW(1)
```
AIO_Set

Output the speed information of the robot to optional analog I/O output channel.

Syntax

(1) AIO_Set channelNumber, On, {RefTCPSpeed | RealTCPSpeed | RefECPSpeed | RealECPSpeed }, MaximumOutputSpeed [, MinimumOutputSpeed]
(2) AIO_Set Channel Number, Off
(3) AIO_Set [Channel Number]

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Number</td>
<td>Specify the channel number of the analog I/O.</td>
</tr>
<tr>
<td>On</td>
<td>Specify the output data (Integer expression from 0 to 65535) in formula or value.</td>
</tr>
<tr>
<td>Off</td>
<td>Finish analog output of the speed information and initializes to output “0”.</td>
</tr>
<tr>
<td>RefTCPSpeed</td>
<td>Output the commanded speed of TCP which is currently selected.</td>
</tr>
<tr>
<td>RealTCPSpeed</td>
<td>Output the actual speed of TCP which is currently selected.</td>
</tr>
<tr>
<td>RefECPSpeed</td>
<td>Output the commanded speed of ECP which is currently selected.</td>
</tr>
<tr>
<td>RealECPSpeed</td>
<td>Output the actual speed of ECP which is currently selected.</td>
</tr>
<tr>
<td>MaximumOutputSpeed</td>
<td>Specify the Real type real number (unit [mm/s]) indicating speed when outputting the maximum value of the output range in formula or value.</td>
</tr>
<tr>
<td>MinimumOutputSpeed</td>
<td>Specify the Real type real number (unit [mm/s]) indicating speed when outputting the minimum value of the output range in formula or value. Value is “0” [0mm/s] when omitting.</td>
</tr>
</tbody>
</table>

Description

Perform real-time output the speed of TCP (tool center point) or ECP (external control point) by analog voltage or current to analog I/O channel specified by channel number. Set the selection of analog voltage or current and output range configuration by a switch and jumper on the analog I/O board.

The robot speed corresponding to minimum and maximum value of the output range is determined by liner interpolation depending on specified minimum output speed and maximum output speed as shown in the figure below.

![Diagram showing speed output range](image)

If specifying the commanded speed (RefTCPSpeed or RefECPSpeed), output the ideal speed waveform based on the applying command value on the robot.
If specifying the actual speed (RealTCPSpeed and RealECPSpeed), output the calculated speed waveform based on the actual robot move.
If specifying the TCP (RefTCP\text{Speed} or RealTCP\text{Speed}), output the center point speed of currently selected tool (default: Tool 0).
If specifying the ECP (RefECPS\text{Speed} or RealECPS\text{Speed}), output the speed of external control point (ECP) which is currently selected. If ECP is not selected (when ECP = 0), output the minimum output.

If only channel number is specified, display the output configuration information of the specified analog channel I/O. If all argument is omitted, display the output configuration information of all analog channel I/O.

**See Also**
AIO\_In, AIO\_InFunction, AIO\_Out, AIO\_OutFunction, AIO\_Out, AIO\_OutWFunction, AIO\_Set, Wait

**AIO\_Set Example**

Set actual speed output of TCP of robot 1 and tool 1 to analog output channel.
Perform analog output the robot operating speed and disable the speed output configuration.

```
Robot 1
Tool 1
Motor On
Power High
SpeedS 2000
AccelS 5000
AIO\_Set 1, On, RealTCP\text{Speed}, 2000.0, 0.0
Move P1
AIO\_Set 1, Off
```
AIO_Set Function

Returns the configuration information of the robot speed output which is set in optional analog I/O output channel.

Syntax
AIO_Set(channelNumber, Index)

Parameters
- Channel number: Specify the channel number of the analog I/O.
- Index: Specify the index of acquiring configuration information in integer.

Return Values
The following table shows the information that is available from the AIO_Set function:

<table>
<thead>
<tr>
<th>Index</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>On(1)/ Off(0)</td>
</tr>
<tr>
<td>2</td>
<td>RefTCPSpeed(0)/ RealTCPSpeed(1)/ RefECPSpeed(2)/ RealECPSpeed(3)</td>
</tr>
<tr>
<td>3</td>
<td>Maximum output speed [mm/sec]</td>
</tr>
<tr>
<td>4</td>
<td>Minimum output speed [mm/sec]</td>
</tr>
</tbody>
</table>

See Also
AIO_In, AIO_InFunction, AIO_Out, AIO_OutW, AIO_OutFunction, AIO_OutWFunction, AIO_Set, Wait

AIO_Set Function Example
Print "Analog Ch#1 speed output is: ", AIO_Set(1, 1)
AIO_TrackingSet

Sets the distance tracking function.

**Syntax**

(1) AIO_TrackingSet  
channelNumber, Conversion coefficient of measured value and distance, Measured value at 0mm, Lower limit of available range for tracking, Upper limit of available range for tracking, [Robot motions out of the available range for tracking], Axis to execute the distance tracking function

(2) AIO_TrackingSet  
channelNumber

**Parameters**

**Channel Number**  
Integer expression from 1 to 8 representing the channel number of analog I/O which the distance sensor to be used is connected.

**Conversion coefficient of measured value and distance**  
Convert the measured value (V, mA) of distance sensor to distance (mm). Specify the coefficient in read number between −500 to 500 excepting 0. (Unit: mm/V, mm/mA)

**Measured value at 0mm**  
Specify the voltage or current value when the distance is 0mm (in case of displacement meter: amount of displacement). (Unit: V, mA)  
Set the value within the input range setting of the analog I/O board.

<table>
<thead>
<tr>
<th>Input range setting</th>
<th>Minimum value</th>
<th>Maximum value</th>
</tr>
</thead>
<tbody>
<tr>
<td>±10.24 V</td>
<td>−10.24 V</td>
<td>10.24 V</td>
</tr>
<tr>
<td>±5.12 V</td>
<td>−5.12 V</td>
<td>5.12 V</td>
</tr>
<tr>
<td>0-5.12 V</td>
<td>0 V</td>
<td>5.12 V</td>
</tr>
<tr>
<td>0-10.24 V</td>
<td>0 V</td>
<td>10.24 V</td>
</tr>
<tr>
<td>0-24 mA</td>
<td>0 mA</td>
<td>24 mA</td>
</tr>
</tbody>
</table>

**Lower limit of available range for tracking**  
Lower limit of the available range for tracking is the same as the lower limit of of the allowable displacement amount when executing the distance tracking function. Specify the limit between −300 to 300 in real number. (Unit: mm)  
Be sure to specify a larger value than the lower limit of the measurable range of the distance sensor.  
For lower limit of the available range for tracking, specify a smaller value than its upper limit.

**Upper limit of available range for tracking**  
Upper limit of the available range for tracking is the same as the upper limit of of the allowable displacement amount when executing the distance tracking function. Specify the limit between −300 to 300 in real number. (Unit: mm)  
Be sure to specify a smaller value than the upper limit of the measurable range of the distance sensor.  
For upper limit of the available range for tracking, specify a larger value than its lower limit.
Robot motions out of the available range for tracking

When the robot is out of the available range for tracking (between the upper and lower limits as described in previous page), specify 0 or 1 to stop/continue the robot motion.
The value can be omitted. If omitted, “0” is set.
Constants are as follows:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIOTRACK_ERRSTOP</td>
<td>0</td>
<td>Robot stops due to an error outside of the available range for tracking.</td>
</tr>
<tr>
<td>AIOTRACK_CONTINUE</td>
<td>1</td>
<td>Robot continues motion outside of the available range for tracking.</td>
</tr>
</tbody>
</table>

Axis to execute the distance tracking function

Specify an axis (integer value from 0 to 5) to execute the distance tracking function. Specify the axis which is matched with the measured direction of the distance sensor to be used.
The value can be omitted. If omitted, “2” is set.
Constants are as follows:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIOTRACK_TOOL_X</td>
<td>0</td>
<td>Tool coordinate X axis</td>
</tr>
<tr>
<td>AIOTRACK_TOOL_Y</td>
<td>1</td>
<td>Tool coordinate Y axis</td>
</tr>
<tr>
<td>AIOTRACK_TOOL_Z</td>
<td>2</td>
<td>Tool coordinate Z axis</td>
</tr>
<tr>
<td>AIOTRACK_ECP_X</td>
<td>3</td>
<td>ECP coordinate X axis</td>
</tr>
<tr>
<td>AIOTRACK_ECP_Y</td>
<td>4</td>
<td>ECP coordinate Y axis</td>
</tr>
<tr>
<td>AIOTRACK_ECP_Z</td>
<td>5</td>
<td>ECP coordinate Z axis</td>
</tr>
</tbody>
</table>

Values: 3 to 5 can be specified when the external control point (ECP) option is enabled.

Return Values

Syntax (2) shows the current set value on the console.

The following is a correspondence table of the above mentioned parameter names and parameter names displayed on the console.

<table>
<thead>
<tr>
<th>Parameter names</th>
<th>Names displayed on the console</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conversion coefficient of measured value and distance</td>
<td>ScaleFactor</td>
</tr>
<tr>
<td>Measured value at 0mm</td>
<td>RefVoltage</td>
</tr>
<tr>
<td>Lower limit of available range for tracking</td>
<td>ThresholdMin</td>
</tr>
<tr>
<td>Upper limit of available range for tracking</td>
<td>ThresholdMax</td>
</tr>
<tr>
<td>Robot motions out of the available range for tracking</td>
<td>OutOfRangeMode</td>
</tr>
<tr>
<td>Axis to execute the distance tracking function</td>
<td>TrackingAxis</td>
</tr>
</tbody>
</table>

Displayed examples are as follows:
Ex 1: When channel #1 is set
   Ch1:
      ScaleFactor 1.000[V/mm or mA/mm]
      RefVoltage 0.000 [V or mA]
      ThresholdMin −10.000[mm]
      ThresholdMax 10.000[mm]
      OutOfRangeMode AIOTRACK_ERRSTOP
      TrackingAxis AIOTRACK_TOOL_Z

Ex2: When channel #1 is not set
   Ch1: Undefined
Description

AIO_TrackingSet sets parameters for the distance tracking function. Parameters to be set are determined by the distance sensor or the working environment. After booting the controller, AIO_TrackingSet must be executed before executing AIO_TrackingStart. Set parameters keep values until the robot controller is turned OFF or rebooted.

Detailed descriptions for parameters are as follows:

Conversion coefficient of measured value and distance:
When the distance sensor indicates displacement: +2mm per +1V, conversion coefficient is 2. At this time, +2mm is the displacement to direction where the distance becomes longer. Depending on the displacement meter, the voltage is set to positive to the direction where the distance becomes shorter. In this case, the conversion coefficient will be negative.

Measured value at 0mm:
For distance sensor, especially the displacement meter, voltage or current value at distance: 0mm differs depending on the products. Also, some of products can set any value for voltage or current value at distance: 0mm by user setting. Specify values depending on the using distance sensor. If the output voltage is of distance sensor is 0V when the distance (or displacement) is 0mm, this parameter is “0”.

Upper/lower limit of available range for tracking:
Set the upper and lower limits depending on the variations allowed by applications. Set values must be within the measurable range of the distance sensor. The measurable range of the distance sensor differs depending on each sensor and user settings. Be sure to set the limits before executing the distance tracking function. If this parameter is set outside the measurable range of the distance sensor, the distance tracking function cannot work properly and the robot may move unintentionally.

Robot motions out of the available range for tracking:
The following figures indicates the motion trajectory of the robot when the distance tracking function is executed to Z direction in Tool (when the “Robot motions out of the available range for tracking” parameter is set to “0” or “1”).

P1: Start point of the distance tracking function
P2: Target point

The figures indicate an object which will move outside of the measurable range at point A and return inside the range at point B.

Set the measured value (displacement) in Tool Z direction at P1 (start point of the function) as a reference value. The distance tracking function controls the robot so that the measured value always becomes the reference value. Therefore, when the robot moves from P1 to P2, the measured values between P1 and point A will be constant.

When the robot is arrived at point A, it stops due to an error if the parameter is set to “0”. If the parameter is set to “1”, the robot keeps moving to P2 from point A. However, the distance tracking function is disabled while the robot is out of the available range. When the robot moved to point B, the function is enabled since the robot is within the available range. The robot moves as with the motion from P1 to point A so that the measured value will be constant.
When the parameter is set to “1” and the robot moves outside of the range, the robot moves on the trajectory from the start point (P1) to the target point (P2) with CP motion. As shown in the figures below, the trajectory between A and B (outside of the available range) will become parallel to its of P1-P2. When the robot arrived to point B, the robot returned to inside the available range. Therefore, the robot is controlled based on the measured value and may move suddenly.

**Tool Z direction**

- 0: Stop the robot motion due to out of the range
- 1: Continue the robot motion even out of the range

<table>
<thead>
<tr>
<th>CAUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>If each parameter is not set correctly, the robot may move unintentionally when AIO_TrackingStart is executed.</td>
</tr>
<tr>
<td>Be sure to set properly depending on the using device and working environment.</td>
</tr>
<tr>
<td>If the robot moves abnormally, immediately hold down the emergency button.</td>
</tr>
</tbody>
</table>

**See Also**
AIO_TrackingStart, AIO_TrackingEnd, AIO_TrackingOn Function
AIO_TrackingSet Function Example
The following is an example program which moves the robot by using the distance tracking function. (P1: Start point, P2: End point)

<table>
<thead>
<tr>
<th>CAUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ The parameters set in the example are reference values. Please note that the operation may not be successful or the motion may be vibratory depending on the set parameters and some operating conditions. If the robot moves abnormally, immediately hold down the emergency button.</td>
</tr>
</tbody>
</table>

Function Main
Motor On
Power High
SpeedS 30
AccelS 300,300

Go P1
AIO_TrackingSet 1,1,0,-5,5,0,2
AIO_TrackingStart 1,5,5,5
Move P2
AIO_TrackingEnd
Motor Off
Fend

• Move to P1: start point
• Set the distance tracking function
• Start the distance tracking function
• Move to P2 with executing the distance tracking function
• End the distance tracking function
Starts the distance tracking function.

**Syntax**

```
AIO_TrackingStart channelNumber,ProportionalGain [,IntegralGain [,DifferentialGain]]
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Channel Number</td>
<td>Integer expression from 1 to 8 representing the channel number of analog I/O which the distance sensor to be used is connected.</td>
</tr>
<tr>
<td>ProportionalGain</td>
<td>Real value (less than 50 except 0) representing a proportional gain of the distance tracking function. Optimum value differs depending on the robot motion speed or workpiece shape. Therefore, the value needs to be set depending on the using environment.</td>
</tr>
<tr>
<td>IntegralGain</td>
<td>Real value (less than 100) representing an integral gain of the distance tracking function. Optional. If omitted, “0” is set. To increase accuracy of the distance tracking, adjust the integral gain.</td>
</tr>
<tr>
<td>DifferentialGain</td>
<td>Real value (less than 100) representing a differential gain of the distance tracking function. Optional. If omitted, “0” is set. To increase accuracy of the distance tracking, adjust the differential gain.</td>
</tr>
</tbody>
</table>

**Description**

The distance tracking function controls the robot so that a constant distance can be kept between the robot and the workpiece using the value measured by distance sensor which is connected to the analog I/O. Direction of the robot axis to be controlled is specified by the “Axis to execute the distance tracking function” parameter of AIO_TrackingSet. If the kept distance is set as “reference value”, the measured value by the distance sensor when executing the command will be the reference value.

Execute AIO_TrackingStart to start the distance tracking function and the function ends by executing AIO_TrackingEnd. The function is working until AIO_TrackingEnd is executed. If you do not use the function, execute AIO_TrackingEnd immediately to end the function.

If AIO_TrackingStart is executed before AIO_TrackingSet, an error occurs. Be sure to execute AIO_TrackingSet before executing AIO_TrackingStart.

The distance tracking function is available for SCARA robots (including RS series manipulators) and 6-Axis robots (including N series manipulators).

The robot can move while the function is working. However, the robot moves in CP motion only and PTP motion is not available.

If the robot passes singularity neighborhood while the distance tracking function is working, an error occurs.
The following commands cannot be used while the distance tracking function is executed.

<table>
<thead>
<tr>
<th>Command to turn OFF the motor</th>
<th>Motor off, SFree</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTP motion commands</td>
<td>BGo, Go, JTran, Jump, Jump3, Jump3CP, JumpTLZ, Pass, Ptran, Pulse, TGo</td>
</tr>
<tr>
<td>Force control commands</td>
<td>FCKeep, Motion commands with FC, FS#.Reset, FS.Reboot</td>
</tr>
<tr>
<td>Torque control command</td>
<td>TC</td>
</tr>
<tr>
<td>Conveyor tracking commands</td>
<td>Motion command + Crv_QueGet</td>
</tr>
<tr>
<td>VRT commands</td>
<td>VRT, VRT_CPMotion</td>
</tr>
<tr>
<td>Setting commands</td>
<td>AIO_TrackingSet, Arm, ArmSet, Base, Calib, CalPls, ECP, ECPSet, HofS, Inertia, MCal, Power, TLSet, Tool, Weight (For AIO_TrackingSet, ArmSet, ECPSet, and TLSet, an error occurs when changing the using number.)</td>
</tr>
<tr>
<td>Others</td>
<td>Brake, Here, Home, VCal, WaitPos</td>
</tr>
</tbody>
</table>

Settings for ProportionalGain, IntegralGain, and DifferentialGain

In ProportionalGain, the larger value you set, the faster the robot tracks. However, if the set value is too large, the robot moves too fast and may result in an error.
IntegralGain and DifferentialGain can be omitted. To increase the correction accuracy, the setting is required.
If the setting is not proper, the robot may move fast or vibrate.
For details on each gain setting, refer to the following manual.
EPSON RC+ User’s Guide: 19. Distance Tracking Function

- If too large value is set for ProportionalGain, IntegralGain, and DifferentialGain, the robot may move unintentionally.
  Please increase values of each parameter gradually. Changing the value to a larger one at one time is extremely hazardous and the robot may move unintentionally.
  If the robot moves abnormally, immediately hold down the emergency button.

See Also
AIO_TrackingSet, AIO_TrackingEnd, AIO_TrackingOn Function
AIO_TrackingStart

AIO_TrackingStart Statement Example
The following is an example program which moves the robot by using the distance tracking function. (P1: Start point, P2: relay point, P3: End point)

<table>
<thead>
<tr>
<th>CAUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>The parameters set in the example are reference values.</td>
</tr>
<tr>
<td>Please note that the operation may not be successful or the motion may be vibratory depending on the set parameters and some operating conditions.</td>
</tr>
<tr>
<td>If the robot moves abnormally, immediately hold down the emergency button.</td>
</tr>
</tbody>
</table>

Function Main
Motor On
Power High
SpeedS 30
AccelS 300,300

  Go P1
  AIO_TrackingSet 1,1,0,-5,5,0,2
  **AIO_TrackingStart** 1,1,0,0
  Move P2
  Move P3
  AIO_TrackingEnd
Motor Off
Fend

  ‘ Move to P1: start point
  ‘ Set the distance tracking function
  ‘ Start the distance tracking function
  ‘ Move to P2 with executing the distance tracking function
  ‘ Move to P3 with executing the distance tracking function
  ‘ End the distance tracking function
AIO_TrackingEnd

Ends the distance tracking function.

Syntax

AIO_TrackingEnd

Description

End the distance tracking function started by AIO_TrackingStart.

See Also

AIO_TrackingSet, AIO_TrackingStart, AIO_TrackingOn Function

AIO_TrackingEnd Statement Example

The following is an example program which moves the robot by using the distance tracking function. (P1: Start point, P2: relay point, P3: End point)

Function Main

Integer ChNo
Motor On
Power High
SpeedS 30
AccelS 300, 300
ChNo=1

Go P1
AIO_TrackingSet ChNo, 10, 0, -3, 3, 0, 2
AIO_TrackingStart ChNo, 1, 0, 0
Move P2

Move P3
AIO_TrackingEnd
Motor Off
Fend

CAUTION

The parameters set in the example are reference values.
Please note that the operation may not be successful or the motion may be vibratory depending on the set parameters and some operating conditions.
If the robot moves abnormally, immediately hold down the emergency button.
AIO_TrackingOnFunction

Returns whether the specified robot is executing the distance tracking function or not.

Syntax

AIO_TrackingOn(robotNumber)

Parameters

robotNumber  An integer expression representing a robot number which you want to acquire.

Return Values

True (−1) when the distance tracking function is executed, False(0) when it stopped.

See Also

AIO_TrackingSet, AIO_TrackingStart, AIO_TrackingEnd

AIO_TrackingOn Function Example

Function Main
    Integer i
    i = AIO_TrackingOn(1)
    print i
Fend

Example on command window

> print AIO_TrackingOn(1)
0
Align Function

Returns the point data converted to align the robot orientation (U, V, W) at the specified point in the tool coordinate system with the nearest or specified axis of the specified local coordinate system.

Syntax

(1) Align (Point[, localNumber[, axisNumber]])

Parameters

- **Point**: The point data.
- **localNumber**: The local coordinate system number to be a reference for the alignment of orientation. If omitted, the base coordinate system is used.
- **axisNumber**: Specify the axis number to align the robot orientation. If omitted, the robot orientation will be aligned to the nearest coordinate axis.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>COORD_X_PLUS</td>
<td>+X axis</td>
</tr>
<tr>
<td>COORD_Y_PLUS</td>
<td>+Y axis</td>
</tr>
<tr>
<td>COORD_Z_PLUS</td>
<td>+Z axis</td>
</tr>
<tr>
<td>COORD_X_MINUS</td>
<td>-X axis</td>
</tr>
<tr>
<td>COORD_Y_MINUS</td>
<td>-Y axis</td>
</tr>
<tr>
<td>COORD_Z_MINUS</td>
<td>-Z axis</td>
</tr>
</tbody>
</table>

Description

While operating the 6-axis robot (including N series), the robot orientation may have to be aligned with an axis of the specified local coordinate system without changing the tool coordinate system position (origin) defined with the point data.

Align Function converts the orientation data (U, V, W) of the specified point data and aligns with the nearest or specified axis of the specified local coordinate system.

For robots except for the 6-axis robots (including N series), it returns a specified point.

See Also

AlignECP Function, LJM Function

Align Function Example

```
Move Align(P0) ROT
P1 = Align(P0, 1)
Move P1 ROT
P2 = Align(P0, 1, 3)
Move P2 ROT
```
AlignECP Function

Returns the point data converted to align the robot orientation (U, V, W) at the specified point in the tool coordinate system with the nearest axis of the specified ECP coordinate system.

Syntax

(1) AlignECP (Point, ECPNumber)

Parameters

Point The point data.
ECPNumber The ECP coordinate system number to be a reference for the alignment of orientation.

Description

While operating the 6-axis robot (including N series), the robot orientation may have to be aligned with an axis of the specified local coordinate system without changing the tool coordinate system position (origin) defined with the point data.

AlignECP Function converts the orientation data (U,V,W) of the specified point data and aligns with the nearest axis of the specified local coordinate system.

For robots except for the 6-axis robots (including N series), it returns a specified point.

See Also
Align Function, LJM Function

AlignECP Function Example

Move AlignECP(P0) ROT
P1 = AlignECP(P0, 1)
Move P1 ROT
And Operator

Operator used to perform a logical or bitwise And of 2 expressions.

Syntax
\[
\text{result} = \text{expr1} \text{ And } \text{expr2}
\]

Parameters
- \(\text{expr1, expr2}\) For logical And, any valid expression which returns a Boolean result. For bitwise And, an integer expression.
- \(\text{result}\) For logical And, result is a Boolean value. For bitwise And, result is an integer.

Description
A logical And is used to combine the results of 2 or more expressions into 1 single Boolean result. The following table indicates the possible combinations.

<table>
<thead>
<tr>
<th>expr1</th>
<th>expr2</th>
<th>result</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>False</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

A bitwise And performs a bitwise comparison of identically positioned bits in two numeric expressions and sets the corresponding bit in result according to the following table:

<table>
<thead>
<tr>
<th>If bit in expr1 is</th>
<th>And bit in expr2 is</th>
<th>The result is</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

See Also
- LShift, Mask, Not, Or, RShift, Xor

And Operator Example

Function LogicalAnd(x As Integer, y As Integer)
  If x = 1 And y = 2 Then
    Print "The values are correct"
  EndIf
EndFunction

Function BitWiseAnd()
  If (Stat(0) And &H800000) = &H800000 Then
    Print "The enable switch is open"
  EndIf
EndFunction

>print 15 and 7
7
>
AOpen Statement

Opens file in the appending mode.

Syntax

\[
\text{AOpen} \text{ fileName As } \#\text{fileNumber} \\
\text{Close } \#\text{fileNumber}
\]

Parameters

fileName  String expression that specifies valid path and file name. If specifying only a file name, the file must be in the current directory. See ChDisk for details.

fileNumber  Integer expression representing values from 30 to 63.

Description

Opens the specified file and identifies it by the specified file number. This statement is used for appending data to the specified file. If the specified file is not found, create a new file. The specified fileNumber identifies the file while it is open and cannot be used to refer to a different file until the current file is closed. fileNumber is used by other file operations such as Print#, Write, Flush, and Close.

Use the Close statement to close the file and release the file number.

It is recommended that you use the FreeFile function to obtain the file number so that more than one task are not using the same number.

Notes

A network path is available.

File write buffering

File writing is buffered. The buffered data can be written with Flush statement. Also, when closing a file with Close statement, the buffered data can be written.

See Also

Close, Print #, BOpen, ROpen, UOpen, WOpen, FreeFile, Flush

AOpen Statement Example

```plaintext
Integer fileNum, i

FileNum = FreeFile
WOpen "TEST.TXT" As #fileNum
For i = 0 To 100
  Print #fileNum, i
Next I
Close #fileNum

....
....
....

FileNum = FreeFile
AOpen "TEST.TXT" As #FileNum
For i = 101 to 200
  Print #FileNum, i
Next i
Close #FileNum
```
Arc, Arc3 Statements

Arc moves the arm to the specified point using circular interpolation in the XY plane. Arc3 moves the arm to the specified point using circular interpolation in 3 dimensions. These two commands are available for SCARA robots (including RS series) and 6-axis robots (including N series).

Syntax

(1) Arc  
midPoint, endPoint [ROT] [CP] [ searchExpr ] [!...!] [SYNC]  
(2) Arc3  
midPoint, endPoint [ROT] [ECP][CP] [ searchExpr ] [!...!] [SYNC]

Parameters

midPoint  
Point expression. The middle point (taught previously by the user) which the arm travels through on its way from the current point to endPoint.

dropDown  
Point expression. The end point (taught previously by the user) which the arm travels to during the arc type motion. This is the final position at the end of the circular move.

ROT  
Optional. Decides the speed/acceleration/deceleration in favor of tool rotation.

ECP  
Optional. External control point motion. This parameter is valid when the ECP option is enabled.

CP  
Optional. Specifies continuous path motion.

searchExpr  
Optional. A Till or Find expression.

Till | Find  
Till Sw(expr) = {On | Off}  
Find Sw(expr) = {On | Off}

!...!  
Parallel processing statements may be used with the Arc statement. These are optional. (Please see the Parallel Processing description for more information.)

SYNC  
Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Arc and Arc3 are used to move the arm in a circular type motion from the current position to endPoint by way of midPoint. The system automatically calculates a curve based on the 3 points (current position, endPoint, and midPoint) and then moves along the curve until the point defined by endPoint is reached. The coordinates of midPoint and endPoint must be taught previously before executing the instruction. The coordinates cannot be specified in the statement itself.

Arc and Arc3 use the SpeedS speed value and AccelS acceleration and deceleration values. Refer to Using Arc3 with CP below on the relation between the speed/acceleration and the acceleration/deceleration. If, however, the ROT modifier parameter is used, Arc and Arc3 use the SpeedR speed value and AccelR acceleration and deceleration values. In this case SpeedS speed value and AccelS acceleration and deceleration value have no effect.

Usually, when the move distance is 0 and only the tool orientation is changed, an error will occur. However, by using the ROT parameter and giving priority to the acceleration and the deceleration of the tool rotation, it is possible to move without an error. When there is not an orientational change with the ROT modifier parameter and movement distance is not “0”, an error will occur.

Also, when the tool rotation is large as compared to move distance, and when the rotation speed exceeds the specified speed of the manipulator, an error will occur. In this case, please reduce the speed or append the ROT modifier parameter to give priority to the rotational speed/acceleration/deceleration.

When ECP is used (Arc3 only), the trajectory of the external control point corresponding to the ECP number specified by ECP instruction moves circular with respect to the tool coordinate system. In this case, the trajectory of tool center point does not follow a circular line.
Setting Speed and Acceleration for Arc Motion

SpeedS and AccelS are used to set speed and acceleration for the Arc and Arc3 instructions. SpeedS and AccelS allow the user to specify a velocity in mm/sec and acceleration in mm/sec².

Notes

Arc Instruction works in Horizontal Plane Only

The Arc path is a true arc in the Horizontal plane. The path is interpolated using the values for endPoint as its basis for Z and U. Use Arc3 for 3 dimensional arcs.

Range Verification for Arc Instruction

The Arc and Arc3 statements cannot compute a range verification of the trajectory prior to the arc motion. Therefore, even for target positions that are within an allowable range, en route the robot may attempt to traverse a path which has an invalid range, stopping with a severe shock which may damage the arm. To prevent this from occurring, be sure to perform range verifications by running the program at low speeds prior to running at faster speeds.

Suggested Motion to Setup for the Arc Move

Because the arc motion begins from the current position, it may be necessary to use the Go, Jump or other related motion command to bring the robot to the desired position prior to executing Arc or Arc3.

Using Arc, Arc3 with CP

The CP parameter causes the arm to move to the end point without decelerating or stopping at the point defined by endPoint. This is done to allow the user to string a series of motion instructions together to cause the arm to move along a continuous path while maintaining a specified speed throughout all the motion. The Arc and Arc3 instructions without CP always cause the arm to decelerate to a stop prior to reaching the end point.

Potential Errors

Changing Hand Attributes

Pay close attention to the HAND attributes of the points used with the Arc instruction. If the hand orientation changes (from Right Handed to Left Handed or vice-versa) during the circular interpolation move, an error will occur. This means the arm attribute (/L Lefty, or /R Righty) values must be the same for the current position, midPoint and endPoint points.

Attempt to Move Arm Outside Work Envelope

If the specified circular motion attempts to move the arm outside the work envelope of the arm, an error will occur.
See Also
!Parallel Processing!, AccelS, Move, SpeedS

Arc, Arc3 Statements Example
The diagram below shows arc motion which originated at the point P100 and then moves through P101 and ends up at P102. The following function would generate such an arc:

```
Function ArcTest
    Go P100
    Arc P101, P102
    Fend
```

Tip
When first trying to use the Arc instruction, it is suggested to try a simple arc with points directly in front of the robot in about the middle of the work envelope. Try to visualize the arc that would be generated and make sure that you are not teaching points in such a way that the robot arm would try to move outside the normal work envelope.
**Arch Statement**

Defines or displays the Arch parameters for use with the Jump, Jump3, Jump3CP instructions.

**Syntax**

1. `Arch archNumber, departDist, approDist`
2. `Arch archNumber`
3. `Arch`

**Parameters**

- `archNumber`: Integer expression representing the Arch number to define. Valid Arch numbers are from 0 to 6 making a total of 7 entries into the Arch table. (see default Arch Table below)
- `departDist`: The vertical distance moved (Z) at the beginning of the Jump move before beginning horizontal motion. (specified in millimeters)
  - For Jump3 and Jump3CP, it specifies the depart distance before a span motion. (specified in millimeters)
- `approDist`: The vertical distance required (as measured from the Z position of the point the arm is moving to) to move in a completely vertical fashion with all horizontal movement complete. (specified in millimeters)
  - For Jump3 and Jump3CP, it specifies the approach distance before a span motion. (specified in millimeters)

**Return Values**

Displays Arch Table when used without parameters.
The Arch table of the specified Arch number will be displayed when only the Arch number is specified.

**Description**

The primary purpose of the Arch instruction is to define values in the Arch Table which is required for use with the Jump motion instruction. The Arch motion is carried out per the parameters corresponding to the arch number selected in the Jump C modifier. (To completely understand the Arch instruction, the user must first understand the Jump instruction.)

The Arch definitions allow the user to "round corners" in the Z direction when using the Jump C instruction. While the Jump instruction specifies the point to move to (including the final Z joint position), the Arch table entries specify how much distance to move up before beginning horizontal motion (riseDist) and how much distance up from the final Z joint position to complete all horizontal motion (fallDist). (See the diagram below)

![Depart Distance](image)

Approach Distance
There are a total of 8 entries in the Arch Definition Table with 7 of them (0-6) being user definable. The 8th entry (Arch 7) is the default Arch which actually specifies no arch at all which is referred to as Gate Motion. (See Gate Motion diagram below) The Jump instruction used with the default Arch entry (Entry 8) causes the arm to do the following:

1) Begin the move with only Z-joint motion until it reaches the Z-Coordinate value specified by the LimZ command. (The upper Z value)
2) Next move horizontally to the target point position until the final X, Y and U positions are reached.
3) The Jump instruction is then completed by moving the arm down with only Z-joint motion until the target Z-joint position is reached.

**Gate Motion**
(Jump with Arch 7)

---

**Arch Table Default Values**

<table>
<thead>
<tr>
<th>Arch Number</th>
<th>Depart Distance</th>
<th>Approach Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>5</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>6</td>
<td>90</td>
<td>90</td>
</tr>
</tbody>
</table>

**Notes**

**Another Cause of Gate Motion**

When the specified value of the Rising Distance or Falling Distance is larger than the actual Z-joint distance which the robot must move to reach the target position, Gate Motion will occur. (i.e. no type Arch motion will occur.)

**Arch values are Maintained**

The Arch Table values are permanently saved and are not changed until either the user changes them.

**Caution for Arch motion**

Jump motion trajectory is comprised of vertical motion and horizontal motion. It is not a continuous path trajectory. The actual Jump trajectory of arch motion is not determined by Arch parameters alone. It also depends on motion and speed.

In a Jump trajectory, the depart distance increases and the approach distance decreases when the motion speed is set high. When the fall distance of the trajectory is shorter than the expected, lower the speed and/or the deceleration, or change the fall distance to be larger.

Always use care when optimizing Jump trajectory in your applications. Execute Jump with the desired motion and speed to verify the actual trajectory. When speed is lower, the trajectory will be lower. If Jump is executed with high speed to verify an arch motion trajectory, the end effector may crash into an obstacle with lower speed.
Arch Statement

Even if Jump commands with the same distance and speed are executed, the trajectory is affected by motion of the robot arms. As a general example, for a SCARA robot the vertical upward distance increases and the vertical downward distance decreases when the movement of the first arm is large. When the vertical fall distance decreases and the trajectory is shorter than the expected, lower the speed and/or the deceleration, or change the fall distance to be larger.

See Also
Jump, Jump3, JumpCP

Arch Statement Example
The following are examples of Arch settings done from the command window.

```plaintext
> arch 0, 15, 15
> arch 1, 25, 50
> jump p1 c1
> arch
    arch0 = 15.000 15.000
    arch1 = 25.000 50.000
    arch2 = 50.000 50.000
    arch3 = 60.000 60.000
    arch4 = 70.000 70.000
    arch5 = 80.000 80.000
    arch6 = 90.000 90.000
> 
```
Arch Function

Returns arch settings.

Syntax
Arch(archNumber, paramNumber)

Parameters

archNumber  Integer expression representing arch setting to retrieve parameter from (0 to 6).
paramNumber  1: depart distance
              2: approach distance

Return Values
Real number containing distance.

See Also
Arch Statement

Arch Function Example

Real archValues(6, 1)
Integer i

' Save current arch values
For i = 0 to 6
    archValues(i, 0) = Arch(i, 1)
    archValues(i, 1) = Arch(i, 2)
Next i
Arm Statement

Selects or displays the arm number to use.

Syntax

(1) Arm armNumber
(2) Arm

Parameters

armNumber Optional integer expression. Valid range is from 0 to 15. The user may select up to 16 different arms. Arm 0 is the standard (default) robot arm. Arm 1 to 15 are auxiliary arms defined by using the ArmSet instruction. When omitted, the current arm number is displayed.

Return Values

When the Arm instruction is executed without parameters, the system displays the current arm number.

Description

Allows the user to specify which arm to use for robot instructions. Arm allows each auxiliary arm to use common position data. If no auxiliary arms are installed, the standard arm (arm number 0) operates. Since at time of delivery the arm number is specified as “0”, it is not necessary to use the Arm instruction to select an arm. However, if auxiliary arms are used they must first defined with the ArmSet instruction.

The auxiliary arm configuration capability is provided to allow users to configure the proper robot parameters for their robots when the actual robot configuration is a little different than the standard robot. For example, if the user mounted a 2nd orientation joint to the 2nd robot link, the user will probably want to define the proper robot linkages for the new auxiliary arm which is formed. This will allow the auxiliary arm to function properly under the following conditions:

- Specifying that a single data point be moved through by 2 or more arms.
- Using Pallet
- Using Continuous Path motion
- Using relative position specifications
- Using Local coordinates

For SCARA robots (including RS series) with rotating joints used with a Cartesian coordinate system, joint angle calculations are based on the parameters defined by the ArmSet parameters. Therefore, this command is critical if any auxiliary arm or hand definition is required.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Notes

Arm 0

Arm 0 cannot be defined or changed by the user through the ArmSet instruction. It is reserved since it is used to define the standard robot configuration. When the user sets Arm to “0”, this means to use the standard robot arm parameters.

Arm Number Not Defined

Selecting auxiliary arm numbers that have not been defined by the ArmSet command will result in an error.
See Also
ArmClr, ArmSet, ECPSet, TLSet

Arm Statement Example
The following examples are potential auxiliary arm definitions using the ArmSet and Arm instructions. ArmSet defines the auxiliary arm and Arm defines which Arm to use as the current arm. (Arm 0 is the default robot arm and cannot be adjusted by the user.)

From the command window:

```plaintext
> ArmSet 1, 300, -12, -30, 300, 0
> ArmSet
    arm0 250 0 0 300 0
    arm1 300 -12 -30 300 0

> Arm 0
> Jump P1 'Jump to P1 using the Standard Arm Config
> Arm 1
> Jump P1 'Jump to P1 using auxiliary arm 1
```
Arm Function

Returns the current arm number for the current robot.

Syntax

    Arm

Return Values

    Integer containing the current arm number.

See Also

    Arm Statement

Arm Function Example

    Print "The current arm number is: ", Arm
ArmClr Statement

Clears (undefines) an arm definition.

Syntax

\[ \text{ArmClr } \text{armNumber} \]

Parameters

- \textit{armNumber} \hspace{1cm} \text{Integer expression representing which of 15 arms to clear (undefine). (Arm 0 is the default arm and cannot be cleared.)}

Description

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also

Arm, ArmSet, ECPSet, Local, LocalClr, Tool, TLSet

ArmClr Statement Example

\[ \text{ArmClr 1} \]
ArmDef Function

Returns arm definition status.

Syntax

\[
\text{ArmDef} \ (\text{armNumber})
\]

Parameters

\[\text{armNumber}\]

Integer expression representing which arm to return status for.

Return Values

True if the specified arm has been defined, otherwise False.

See Also

Arm, ArmClr, ArmSet, ECPSet, Local, LocalClr, Tool, TLClr, TLSet

ArmDef Function Example

```
Function DisplayArmDef(armNum As Integer)

    Integer i

    If ArmDef(armNum) = False Then
        Print "Arm ", ArmNum, " is not defined"
    Else
        Print "Arm ", armNum, " Definition:"
        For i = 1 to 5
            Print ArmSet(armNum, i)
        Next i
    EndIf

End
```

ArmSet Statement

Specifies and returns auxiliary arms.

Syntax
(1) ArmSet armNumber, link2Dist, joint2Offset, zOffset [, link1Dist] [, orientAngOffset]
(2) ArmSet armNumber
(3) ArmSet

Parameters

| armNumber | Integer expression: Valid range from 1 to 15. The user may define up to 15 different auxiliary arms. |

<table>
<thead>
<tr>
<th>paramName</th>
<th>SCARA Robots (including RS series)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Horizontal distance from joint #2 to orientation center (mm)</td>
</tr>
<tr>
<td>2</td>
<td>Joint #2 angle offset (degree)</td>
</tr>
<tr>
<td>3</td>
<td>Height offset (mm)</td>
</tr>
<tr>
<td>4</td>
<td>Horizontal distance from joint #1 to joint #2 (mm)</td>
</tr>
<tr>
<td>5</td>
<td>Orientation joint angle offset in degrees.</td>
</tr>
</tbody>
</table>

Return Values
When the ArmSet instruction is initiated without parameters, the system displays all the auxiliary arm numbers and parameters. The specified arm numbers and parameters will be displayed when only the arm number is specified.

Description
Allows the user to specify auxiliary arm parameters to be used in addition to the standard arm configuration. This is most useful when an auxiliary arm or hand is installed to the robot. When using an auxiliary arm, the arm is selected by the Arm instruction.

The link1Dist and orientAngOffset parameters are optional. If they are omitted, the default values are the standard arm values.

The auxiliary arm configuration capability is provided to allow users to configure the proper robot parameters for their robots when the actual robot configuration is a little different than the standard robot. For example, if the user mounted a 2nd orientation joint to the 2nd robot link, the user will probably want to define the proper robot linkages for the new auxiliary arm which is formed. This will allow the auxiliary arm to function properly under the following conditions:

- Specifying that a single data point be moved through by 2 or more arms.
- Using Pallet
- Using Continuous Path motion
- Using relative position specifications
- Using Local coordinates

For SCARA robots (including RS series) with rotating joints used with a Cartesian coordinate system, joint angle calculations are based on the parameters defined by the ArmSet parameters. Therefore, this command is critical if any auxiliary arm or hand definition is required.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.
ArmSet Statement

Note

Arm 0

Arm 0 cannot be defined or changed by the user. It is reserved since it is used to define the standard robot configuration. When the user sets Arm to 0 this means to use the standard robot arm parameters.

See Also

Arm, ArmClr

ArmSet Statement Example

The following examples are potential auxiliary arm definitions using the ArmSet and Arm instructions. ArmSet defines the auxiliary arm and Arm defines which Arm to use as the current arm. (Arm 0 is the default robot arm and cannot be adjusted by the user.)

From the command window:

> \textbf{ArmSet} 1, 300, -12, -30, 300, 0
> \textbf{ArmSet}
> Arm 0: 125.000, 0.000, 0.000, 225.000, 0.000
> Arm 1: 300.000, -12.000, -30.000, 300.000, 0.000

> Arm 0
> \textbf{Jump P1} \quad \textit{Jump to P1 using the Standard Arm Config}
> \textbf{Arm 1}
> \textbf{Jump P1} \quad \textit{Jump to P1 using auxiliary arm 1}
ArmSet Function

Returns one ArmSet parameter.

Syntax

\[ \text{ArmSet}(\text{armNumber}, \text{paramNumber}) \]

Parameters

- \( \text{armNumber} \)  
  Integer expression representing the arm number to retrieve values for.
- \( \text{paramNumber} \)  
  Integer expression representing the parameter to retrieve (0 to 5), as described below.

<table>
<thead>
<tr>
<th>SCARA Robots (including RS series)</th>
<th>Value Returned</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{paramNumber} )</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Horizontal distance from joint #2 to orientation center (mm)</td>
</tr>
<tr>
<td>2</td>
<td>Joint #2 angle offset (degree)</td>
</tr>
<tr>
<td>3</td>
<td>Height offset (mm)</td>
</tr>
<tr>
<td>4</td>
<td>Horizontal distance from joint #1 to joint #2 (mm)</td>
</tr>
<tr>
<td>5</td>
<td>Orientation joint angle offset in degrees.</td>
</tr>
</tbody>
</table>

Return Values

Real number containing the value of the specified parameter, as described above.

See Also

ArmClr, ArmSet Statement

ArmSet Function Example

\[
\text{Real } x
\]

\[
x = \text{ArmSet}(1, 1)
\]
Asc Function

Returns the ASCII code of the first character in a character string.
(Returns the character code in a decimal number.)

Syntax

Asc(string)

Parameters

string Any valid string expression of at least one character in length.

Return Values

Returns an integer representing the ASCII code of the first character in the string sent to the Asc function.

Description

The Asc function is used to convert a character to its ASCII numeric representation. The character string send to the ASC function may be a constant or a variable.

Note

Only the First Character ASCII Value is Returned

Although the Asc instruction allows character strings larger than 1 character in length, only the 1st character is actually used by the Asc instruction. Asc returns the ASCII value of the 1st character only.

See Also

Chr$, InStr, Left$, Len, Mid$, Right$, Space$, Str$, Val

Asc Function Example

This example uses the Asc instruction in a program and from the command window as follows:

```asc
def asctest
    integer a, b, c
    a = Asc("a")
    b = Asc("b")
    c = Asc("c")
    print "The ASCII value of a is ", a
    print "The ASCII value of b is ", b
    print "The ASCII value of c is ", c
end
```

From the command window:

```
>print asc("a")
97
>print asc("b")
98
>```
Asin Function

Returns the arcsine of a numeric expression.

Syntax

\[ \text{Asin}(\text{number}) \]

Parameters

\[ \text{number} \quad \text{Numeric expression representing the sine of an angle.} \]

Return Values

Real value, in radians, representing the arc sine of the parameter \( \text{number} \).

Description

Asin returns the arcsine of the numeric expression. Values range is from -1 to 1. The value returned by Asin will range from \(-\pi / 2\) to \(\pi / 2\) radians. If \( \text{number} \) is < -1 or > 1, an error occurs.

To convert from radians to degrees, use the RadToDeg function.

See Also

Abs, Acos, Atan, Atan2, Cos, DegToRad, RadToDeg, Sgn, Sin, Tan, Val

Asin Function Example

```plaintext
Function asintest
    Double x
    x = Sin(DegToRad(45))
    Print "Asin of ", x, " is ", Asin(x)
End
```
AtHome Function

Returns if the current robot is in its Home position or not.

Syntax
AtHome

Return Values
True if the current robot is in its Home position, otherwise False.

Description
The AtHome function returns if the current robot is in its Home position or not. To register the Home position, use HomeSet command or Robot Manager. To move to the Home position, use the Home command.

See Also
Home, HomeClr, HomeDef, HomeSet, Hordr, MCAlComplete
Atan Function

Returns the arctangent of a numeric expression.

Syntax

\[
\text{Atan}(\text{number})
\]

Parameters

\text{number} \quad \text{Numeric expression representing the tangent of an angular value.}

Return Values

Real value, in radians, representing the arctangent of the parameter \text{number}.

Description

Atan returns the arctangent of the numeric expression. The numeric expression (\text{number}) may be any numeric value. The value returned by Atan will range from -PI to PI radians.

To convert from radians to degrees, use the RadToDeg function.

See Also

Abs, Acos, Asin, Atan2, Cos, DegToRad, RadToDeg, Sgn, Sin, Tan, Val

Atan Function Example

Function atantest
    Real x, y
    x = 0
    y = 1
    Print "Atan of ", x, " is ", Atan(x)
    Print "Atan of ", y, " is ", Atan(y)
    Fend
**Atan2 Function**

Returns the angle of the imaginary line connecting points (0,0) and (X, Y) in radians.

**Syntax**

```
Atan2(X, Y)
```

**Parameters**

- **X**: Numeric expression representing the X coordinate.
- **Y**: Numeric expression representing the Y coordinate.

**Return Values**

Numeric value in radians (-PI to +PI).

**Description**

Atan2(X, Y) returns the angle of the line which connects points (0, 0) and (X, Y). This trigonometric function returns an arctangent angle in all four quadrants.

**See Also**

Abs, Acos, Asin, Atan, Cos, DegToRad, RadToDeg, Sgn, Sin, Tan, Val

**Atan2 Function Example**

```
Function at2test
    Real x, y
    Print "Please enter a number for the X Coordinate:" Input x
    Print "Please enter a number for the Y Coordinate:" Input y
    Print "Atan2 of ", x, ", ", y, ", is ", Atan2(x, y)
End
```
ATCLR Statement

Clears and initializes the average torque for one or more joints.

**Syntax**

```
ATCLR [j1 [, j2 [, j3 [, j4 [, j5 [, j6 [, j7 [, j8 [, j9]]]]]]]]
```

**Parameters**

- **j1 – j9**
  
  Integer expression representing the joint number. If no parameters are supplied, then the average torque values are cleared for all joints. The additional S axis is 8 and T axis is 9. If non-existent joint number is supplied, an error occurs.

**Description**

ATCLR clears the average torque values for the specified joints.

You must execute ATCLR before executing ATRQ.

**See Also**

ATRQ, PTRQ

**ATCLR Statement Example**

<Example 1>

The following is the example to display the torque values of specified joints after clearing the effective torque values of all joints.

```
> atclr
> go p1
> atrq 1
   0.028
> atrq
   0.028   0.008
   0.029   0.009
   0.000   0.000
>
```

<Example 2>

The following is the example to display the torque values of specified joints after clearing the effective torque values of J1, J4, and J5 for the vertical multi-axis robots.

```
> atclr 4, 1, 5
> go p1
> ptrq 1
   0.227
> ptrq 4
   0.083
```
ATRQ Statement

Displays the average torque for the specified joint.

Syntax

:\texttt{ATRQ [jointNumber]}\n
Parameters

\textit{jointNumber}  Optional. Integer expression representing the joint number.

The additional S axis is 8 and T axis is 9.

Return Values

Displays current average torque values for all joints.

Description

ATRQ displays the average RMS (root-mean-square) torque of the specified joint. The loading state of the motor can be obtained by this instruction. The result is a real value from 0 to 1 with 1 being maximum average torque.

You must execute ATCLR before this command is executed.

This instruction is time restricted. You must execute ATRQ within 60 seconds after ATCLR is executed. When this time is exceeded, error 4030 occurs.

See Also

ATCLR, ATRQ Function, PTRQ

ATRQ Statement Example

:\begin{verbatim}
> atclr
> go p1
> atrq 1
> 0.028
> atrq
> 0.028 0.008
> 0.029 0.009
> 0.000 0.000
> 
\end{verbatim}
ATRQ Function

Returns the average torque for the specified joint.

Syntax

\[ \text{ATRQ} \left( \text{jointNumber} \right) \]

Parameters

\( \text{jointNumber} \) Integer expression representing the joint number.
The additional S axis is 8 and T axis is 9.

Return Values

Real value from 0 to 1.

Description

The ATRQ function returns the average RMS (root-mean-square) torque of the specified joint. The loading state of the motor can be obtained by this instruction. The result is a real value from 0 to 1 with 1 being maximum average torque.

You must execute ATCLR before this function is executed.

This instruction is time restricted. You must execute ATRQ within 60 seconds after ATCLR is executed. When this time is exceeded, error 4030 occurs.

See Also

ATRQ Statement, PTCLR, PTRQ

ATRQ Function Example

This example uses the ATRQ function in a program:

```plaintext
Function CheckAvgTorque
    Integer i
    Go P1
    ATCLR
    Go P2
    Print "Average torques:"
    For i = 1 To 4
        Print "Joint ", i, ", " = ", ATRQ(i)
    Next i
End
```

ATRQ Function Example

This example uses the ATRQ function in a program:
AutoLJM Statement

Sets the Auto LJM function.

Syntax

AutoLJM { On | Off }

Parameters

On | Off

On: Enables the Auto LJM.
Off: Disables the Auto LJM.

Description

AutoLJM is available for the following commands:
Arc, Arc3, Go, Jump3, Jump3CP, Move

When AutoLJM is On, the manipulator operates with a least joint motion, just like using the LJM function, whether the LJM function is applied to the position data to be passed to each command or not. For example, to get the same effect as Go LJM(P1), you can write a program as follows.

AutoLJM On
Go P1
AutoLJM Off

Since AutoLJM can enable LJM within a particular section of a program, it is not necessary to edit each motion command.

When AutoLJM is Off, the LJM function is only enabled when it is applied to the position data to be passed to each motion command.

In any of the following cases, AutoLJM has the setting specified in the controller settings (factory default: Off).

Controller startup
Reset
All task stop
Motor On
Switching the Auto / Programming operation mode

Notes

Double application of AutoLJM and LJM function

If LJM function is applied to the point data to be passed to the motion command while AutoLJM is On, LJM will be doubly applied at the command execution.

For Move LJM(P1, Here) and Move LJM(P1), enabling AutoLJM will not affect the motion. However, if AutoLJM is enabled for Move LJM(P1, P0), motion completion positions of Move LJM(LJM(P1, P0), Here), which enabled AutoLJM, and the one of Move LJM(P1, P0), which did not enable AutoLJM, may be different.

It is recommended to write a program not to duplicate AutoLJM and LJM functions.

AutoLJM Usage Precaution

You can set the AutoLJM function to be enabled at the controller startup by setting the controller preferences. However, if Auto LJM is enabled at all times by controller preferences or commands, 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 create a program to apply the LJM function only when necessary by using LJM function or AutoLJM command.
AutoLJM Statement

See Also
AutoLJM Function, LJM Function

AutoLJM Statement Example

```
AutoLJM  On
Go  P1
Go  P2
AutoLJM  Off
```
AutoLJM Function

Returns the state of the AutoLJM.

Syntax
AutoLJM

Return Values
0  = Auto LJM OFF
1  = Auto LJM ON

See Also
AutoLJM

AutoLJM Function Example

If AutoLJM = Off Then
    Print "AutoLJM is off"
EndIf
AutoOrientationFlag

Changes orientation flag of N6-A1000**.

Syntax

AutoOrientationFlag { On | Off }

Parameters

On | Off

On: Enables the AutoOrientationFlag.

Off: Disables the AutoOrientationFlag. (Default)

Description

AutoOrientationFlag is available for following commands:

Go, BGo, TGo, Jump3, JumpTLZ

Change the following orientation flag:

<table>
<thead>
<tr>
<th>Model</th>
<th>Parameter OFF/ON</th>
<th>Orientation flag</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Hand</td>
<td>Elbow</td>
</tr>
<tr>
<td>N6-A1000**</td>
<td>OFF</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>-</td>
<td>✓</td>
</tr>
</tbody>
</table>

✓: When setting the AutoOrientationFlag to “ON”, the orientation flag is changed

*1: Wrist orientation flag is changed only when you change the elbow orientation flag. When you change the wrist orientation flag, it will be the orientation flag which minimizes the movement of Joint #4.

Use AutoOrientationFlag with LJM Function

When you use the command with LJM Function, Wrist Flag, J4Flag, and J6Flag will be the orientation selected by LJM Function.

For example, when you set orientationFlag of LJM Function to “3”, “Wrist Flag”, “J4Flag”, and “J6Flag” are selected so that Joint #5 will be the shortest movement. When you do not use LJM Function, “Wrist Flag”, “J4Flag”, and “J6Flag” are selected so that Joint #4 will be the shortest movement.

AutoOrientationFlag Example

Motor On

Power High

AutoOrientationFlag On

Go P1

Go P2

NOTE

When setting the AutoOrientationFlag to “ON”:

Flag is changed as follows due to the position of point P and the red line.

Point P is above the red line: Above
Point P is below the red line: Below
AutoOrientationFlag Function

Returns the state of the AutoOrientationFlag

Syntax
AutoOrientationFlag

Return Values
0 = AutoOrientationFlag OFF
1 = AutoOrientationFlag ON

See Also
AutoOrientationFlag

AutoOrientationFlag Function Example
If AutoOrientationFlag = Off Then
    Print " AutoOrientationFlag is off"
EndIf
AvgSpeedClear Statement

Clears and initializes the average of the absolute speed values for one or more joints.

Syntax
AvgSpeedClear \([j1, j2, j3, j4, j5, j6, j7, j8, j9]\\)\\]

Parameters
\(j1 \text{ to } j9\) Integer expression representing the joint number. If no parameters are supplied, then the average values for all joints are cleared. The additional S axis is 8 and T axis is 9. If non-existent joint number is supplied, an error occurs.

Description
AvgSpeedClear clears the average of the absolute speed values for the specified joints. You must execute AvgSpeedClear before executing AvgSpeed. This command does not support the PG additional axes.

See Also
AvgSpeed, PeakSpeed

AvgSpeedClear Statement Example

<Example 1>
The following is the example to display the average speed values of specified joints after clearing the average speed values of all joints.

```
> AvgSpeedClear
> Go P1
> AvgSpeed 1
  0.073
> AvgSpeed
  0.073  0.044
  0.021  0.069
  0.001  0.108
  0.000  0.000
  0.000
>
```

<Example 2>
The following is the example to display the average speed values of specified joints after clearing the average speed values of J1, J4, and J5 for the vertical multi-axis robots.

```
> AvgSpeedClear 4, 1, 5
> Go P1
> AvgSpeed 1
  0.226
> AvgSpeed 4
  0.207
```
AvgSpeed Statement

Displays the average of the absolute speed values for the specified joints.

Syntax

```
AvgSpeed [jointNumber]
```

Parameters

`jointNumber` Optional. Integer expression representing the joint number.
The additional S axis is 8 and T axis is 9.

Return Values

Displays the average of the absolute values of current speed for the specified joints. If no joint is specified, the average of the absolute speed values for all joints will be displayed.

Description

AvgSpeed displays the average value of the absolute speed values for the specified joints. The loading state of the motor can be obtained by this instruction. The result is a real value from 0 to 1 with 1 being the maximum average speed value.

If the average value is below 0.001, the result will be displayed as 0.

You must execute AvgSpeedClear before this command is executed.

This instruction is time restricted. You must execute AvgSpeed within 60 seconds after AvgSpeedClear is executed. When this time is exceeded, error 4088 occurs.

When using the virtual controller or conducting dry-run, the average of the absolute speed values is calculated from the commanded speed instead of the actual speed.

This command does not support the PG additional axes.

See Also

AvgSpeedClear, AvgSpeed Function, PeakSpeed

AvgSpeed Statement Example

```
> AvgSpeedClear
> Go P1
> AvgSpeed 1
  0.226
> AvgSpeed
  0.226  0.133
  0.064  0.207
  0.003  0.314
  0.000  0.000
  0.000
> 
```
AvgSpeed Function

Returns the average value of the absolute speed values for the specified joints.

Syntax

AvgSpeed (jointNumber)

Parameters

jointNumber  Integer expression representing the joint number.
The additional S axis is 8 and T axis is 9.

Return Values

Real value from 0 to 1.

Description

AvgSpeed function returns the average value of the absolute speed values for the specified joints. The loading state of the motor can be obtained by this function. The result is a real value from 0 to 1 with 1 being the maximum average speed value.

You must execute AvgSpeedClear before this command is executed.

This instruction is time restricted. You must execute AvgSpeed function within 60 seconds after AvgSpeed statement is executed. When this time is exceeded, error 4088 occurs.

When using the virtual controller or conducting dry-run, the average of the absolute speed values is calculated from the commanded speed instead of the actual speed.
This command does not support the PG additional axes.

See Also

AvgSpeed, AvgSpeedClear, PeakSpeed

AvgSpeed Function Example

This example uses the AvgSpeed function in a program:

```plaintext
Function CheckAvgSpeed
    Integer i

    Go P1
    AvgSpeedClear
    Go P2
    Print "Average speeds:"
    For i = 1 To 6
        Print "Joint ", i, " = ", AvgSpeed (i)
    Next i

Fend
```
AvoidSingularity Statement

Sets the singularity avoiding function.

Syntax

AvoidSingularity { mode }

Parameters

<table>
<thead>
<tr>
<th>mode</th>
<th>Integer expression representing a singularity avoiding mode to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>SING_NONE</td>
<td>0</td>
</tr>
<tr>
<td>SING_THRU</td>
<td>1</td>
</tr>
<tr>
<td>SING_THRUROT</td>
<td>2</td>
</tr>
<tr>
<td>SING_VSD</td>
<td>3</td>
</tr>
<tr>
<td>SING_AUTO</td>
<td>4</td>
</tr>
<tr>
<td>SING_AVOID</td>
<td>5</td>
</tr>
</tbody>
</table>

Description

AvoidSingularity is available for following commands.
Move, Arc, Arc3, Jump3, Jump3CP, JumpTLZ

A singularity avoiding function is to prevent acceleration errors when the vertical 6-axis (including N series) or RS series robot approaches to the singularity in CP motion by passing a different trajectory and returning to the original trajectory after passing the singularity. Since the singularity avoiding function is usually set to “1: Enabled” at the controller startup, it is not necessary to change the setting. If you do not want a singularity avoidance to ensure compatibility with software which does not support the singularity avoiding function, or to avoid a trajectory gap, disable the function.

A variable speed CP motion function automatically controls speed while keeping the trajectory when the vertical 6-axis (including N series) or RS series robot approaches to the singularity in order to avoid the acceleration error and overspeed error, and returns to the normal speed command after leaving the singularity. To pass the singularity while keeping the trajectory, Joint #1, #2, #4, and #6 may move largely.

If the AvoidSingularity parameter is changed, this function remains enabled until the next controller startup. At the controller startup, AvoidSingularity has the setting specified in the controller setting (factory default: 1). Also, parameters for SingularityAngle, SingularitySpeed, and SingularityDist are reset to the default values when AvoidSingularity setting is changed.

SING_AUTO mode is the combination of SING_THRU and SING_VSD modes. SING_THRU or SING_VSD is selected depending on the motion or speed.
AvoidSingularity Statement

Notes

Condition setting of singularity neighborhood for vertical 6-axis robot and N series robot
To determine whether the manipulator approaches to the wrist singularity neighborhood, angle of Joint #5 and angular velocity of Joint #4 are used. By default, Joint #5 angle is set to ±10 degree, and Joint #4 angle is set to ±10 % with respect to the maximum joint velocity. To change these settings, use SingularityAngle and SingularitySpeed commands. Also, to determine whether the manipulator approaches to the hand singularity neighborhood, the coordinates of the point P is used. By default, distance between the point P and Joint #1 rotation axis is set to 30 mm. To change this setting, use SingularityDist command.

Condition setting of singularity neighborhood for RS series robot
To determine whether the manipulator approaches to the hand singularity neighborhood, the coordinates of the origin point in the default tool 0 coordinate system is used. By default, distance between the origin point and Joint #1 rotation axis is set to 30 mm. To change this setting, use SingularityDist command.

Cautions for N series robot
Unlike other models, the default setting of singularity avoidance function for N series is “3: Enables variable speed CP motion function.” N series robots have the elbow singularity other than the wrist and hand singularities. The elbow singularity area is where the Joint #3 is at 0 degree (the Joint #3 and Joint #2 overlap each other). For details of avoiding motion near the elbow singularity area, refer to the EPSON RC+ User’s Guide.

Difference between SING_THRU and SING_AVOID
SING_THRU avoids the wrist and shoulder singularities, but not the elbow singularity. To avoid the elbow singularity, use SING_AVOID. Note, however, that the elbow singularity avoiding motion changes the trajectory largely than the other singularity avoiding motions. When SING_AVOID is selected for the manipulator models other than N series, an error 4002 occurs.

See Also
AvoidSingularity Function, SingularityAngle, SingularitySpeed, SingularityDist

AvoidSingularity Statement Example

```
AvoidSingularity 0  'Disables the singularity avoidance and operate the manipulator
Move P1
Move P2
AvoidSingularity 1
```
AvoidSingularity Function

Returns the state of AvoidSingularity.

Syntax

AvoidSingularity

Return Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Singularity avoiding function disabled</td>
</tr>
<tr>
<td>1</td>
<td>Singularity avoiding function enabled</td>
</tr>
<tr>
<td>2</td>
<td>Singularity avoiding function enabled for CP motion commands with an ROT modifier</td>
</tr>
<tr>
<td>3</td>
<td>Variable speed CP motion function enabled</td>
</tr>
<tr>
<td>4</td>
<td>Automatic selection of the singularity avoiding function or the variable speed CP motion function</td>
</tr>
<tr>
<td>5</td>
<td>Elbow singularity avoiding function enabled</td>
</tr>
</tbody>
</table>

See Also

AvoidSingularity

AvoidSingularity Function Example

```
If AvoidSingularity = Off Then
    Print "AvoidSingularity is off"
EndIf
```
Base Statement

Defines and displays the base coordinate system.

Syntax
(1) Base pCoordinateData
(2) Base pOrigin, pXaxis, pYaxis [, { X | Y }]

Parameters
pCoordinateData  Point data representing the coordinate data of the origin and direction.
pOrigin  Integer expression representing the origin point using robot coordinate system.
pXaxis  Integer expression representing a point along the X axis using robot coordinate system if X alignment is specified.
pYaxis  Integer expression representing a point along the Y axis using robot coordinate system if Y alignment is specified.
X | Y  Optional. If X alignment is specified, then pXaxis is on the X axis of the new coordinate system and only the Z coordinate of pYaxis is used. If Y alignment is specified, then pYaxis is on the Y axis of the new coordinate system and only the Z coordinate of pXaxis is used. If omitted, X alignment is assumed.

Description
Defines the robot base coordinate system by specifying base coordinate system origin and rotation angle in relation to the robot absolute coordinate system.

To reset the Base coordinate system to default, execute the following statement. This will make the base coordinate system the same as the robot absolute coordinate system.

Base XY(0, 0, 0, 0)

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Note
Changing the base coordinate system affects all local definitions
When base coordinates are changed, all local coordinate systems must be re-defined.

See Also
Local

Base Statement Example
Define base coordinate system origin at 100 mm on X axis and 100 mm on Y axis

> Base XY(100, 100, 0, 0)
BClr Function

Clears one bit in a number and returns the new value

Syntax

\[
\text{BClr} \ (\text{number, bitNum})
\]

Parameters

- **number**: Specifies the numeric value to clear the bit by an expression or numeric value.
- **bitNum**: Specifies the bit (integer from 0 to 31) to be cleared by an expression or numeric value.

Return Values

Returns the new value of the specified numeric value (integer).

See Also

- BClr64, BSet, BSet64, BTst, BTst64

BClr Function Example

\[
\text{flags} = \text{BClr}(\text{flags, 1})
\]
BClr64 Function

Clears one bit in a number and returns the new value.

Syntax

\[
\text{BClr64}(\text{number, bitNum})
\]

Parameters

- **number**: Specifies the numeric value to clear the bit by an expression or numeric value.
- **bitNum**: Specifies the bit (integer from 0 to 63) to be cleared by an expression or numeric value.

Return Values

Returns the new value of the specified numeric value (integer).

See Also

BClr, BSet, BSet64, BTst, BTst64

BClr64 Function Example

\[
\text{flags} = \text{BClr64}(\text{flags}, 1)
\]
BGo Statement

Executes Point to Point relative motion, in the selected local coordinate system.

Syntax

```
BGo destination [CP] [PerformMode modeNumber] [searchExpr] [!]...! [SYNC]
```

Parameters

- **destination**
  - The target destination of the motion using a point expression.
- **CP**
  - Optional. Specifies continuous path motion.
- **PerformMode**
  - Optional. Specify the robot performance mode.
- **modeNumber**
  - Specify the operation mode assigned to `PerformMode` with an integer value (1 to 3) or with the following constant. If `PerformMode` is specified, this parameter cannot be omitted.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_STANDARD</td>
<td>1</td>
<td>Sets the Standard mode.</td>
</tr>
<tr>
<td>MODE_HIGH_SPEED</td>
<td>2</td>
<td>Sets the High-speed mode.</td>
</tr>
<tr>
<td>MODE_LOW_OSCILLATION</td>
<td>3</td>
<td>Sets the Low-oscillation mode.</td>
</tr>
</tbody>
</table>

- **searchExpr**
  - Optional. A Till or Find expression.
  - **Till**
    - `Till Sw(expr) = {On | Off}`
  - **Find**
    - `Find Sw(expr) = {On | Off}`
- **!]...!**
  - Optional. Parallel Processing statements can be added to execute I/O and other commands during motion.
- **SYNC**
  - Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Executes point to point relative motion, in the selected local coordinate system that is specified in the `destination` point expression.

If a local coordinate system is not specified, relative motion will occur in local 0 (base coordinate system).

Arm orientation attributes specified in the `destination` point expression are ignored. The manipulator keeps the current arm orientation attributes. However, for a 6-Axis manipulator (including N series), the arm orientation attributes are automatically changed in such a way that joint travel distance is as small as possible. This is equivalent to specifying the LJM modifier parameter for Move statement. Therefore, if you want to change the arm orientation larger than 180 degrees, execute it in several times.

The Till modifier is used to complete BGo by decelerating and stopping the robot at an intermediate travel position if the current Till condition is satisfied.

The Find modifier is used to store a point in FindPos when the Find condition becomes true during motion.

When parallel processing is used, other processing can be executed in parallel with the motion command.

The CP parameter causes acceleration of the next motion command to start when the deceleration starts for the current motion command. In this case the robot will not stop at the destination coordinate and will continue to move to the next point.

Deceleration motion and acceleration motion of different modes can be combined when `PerformMode` is set while the path motion is enabled. Some combinations are not available depending on operation modes. For details, refer to `PerformMode Statement`.

106 EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
See Also
Accel, BMove, Find, !....! Parallel Processing, Point Assignment, PerformMode, Speed, Till, TGo, TMove, Tool

BGo Statement Example

> BGo XY(100, 0, 0, 0) "Move 100 mm in X direction (in the local coordinate system)

Function BGoTest

    Speed 50
    Accel 50, 50
    Power High

    P1 = XY(300, 300, -20, 0)
P2 = XY(300, 300, -20, 0) /L
Local 1, XY(0, 0, 0, 45)

    GoP1
    Print Here
BGo XY(0, 50, 0, 0)
    Print Here

    Go P2
    Print Here
BGo XY(0, 50, 0, 0)
    Print Here

    BGo XY(0, 50, 0, 0) /1
    Print Here

Fend

[Output]
X:  300.000 Y:  300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y:  350.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y:  350.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
X:  300.000 Y:  350.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
X:  264.645 Y:  385.355 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
BMove Statement

Executes linear interpolation relative motion, in the selected local coordinate system.

Syntax

BMove destination [ROT] [CP] [searchExpr] [!!] [SYNC]

Parameters

destination
The target destination of the motion using a point expression.

ROT
Optional. Decides the speed/acceleration/deceleration in favor of tool rotation.

CP
Optional. Specifies continuous path motion.

searchExpr
Optional. A Till or Find expression.

Till | Find
Till Sw(expr) = {On | Off}
Find Sw(expr) = {On | Off}

!!
Optional. Parallel Processing statements can be added to execute I/O and other commands during motion.

SYNC
Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Executes linear interpolated relative motion, in the selected local coordinate system that is specified in the destination point expression.

If a local coordinate system is not specified, relative motion will occur in local 0 (base coordinate system).

Arm orientation attributes specified in the destination point expression are ignored. The manipulator keeps the current arm orientation attributes. However, for a 6-Axis manipulator (including N series), the arm orientation attributes are automatically changed in such a way that joint travel distance is as small as possible. This is equivalent to specifying the LJM modifier parameter for Move statement. Therefore, if you want to change the arm orientation larger than 180 degrees, execute it in several times.

BMove uses the SpeedS speed value and AccelS acceleration and deceleration values. Refer to Using BMove with CP below on the relation between the speed/acceleration and the acceleration/deceleration. If, however, the ROT modifier parameter is used, BMove uses the SpeedR speed value and AccelR acceleration and deceleration values. In this case SpeedS speed value and AccelS acceleration and deceleration value have no effect.

Usually, when the move distance is “0” and only the tool orientation is changed, an error will occur. However, by using the ROT parameter and giving priority to the acceleration and the deceleration of the tool rotation, it is possible to move without an error. When there is not an orientational change with the ROT modifier parameter and movement distance is not “0”, an error will occur.

Also, when the tool rotation is large as compared to move distance, and when the rotation speed exceeds the specified speed of the manipulator, an error will occur. In this case, please reduce the speed or append the ROT modifier parameter to give priority to the rotational speed/acceleration/deceleration.

The Till modifier is used to complete BMove by decelerating and stopping the robot at an intermediate travel position if the current Till condition is satisfied.

The Find modifier is used to store a point in FindPos when the Find condition becomes true during motion.
When Till is used and the Till condition is satisfied, the manipulator halts immediately and the motion command is finished. If the Till condition is not satisfied, the manipulator moves to the destination point.

When Find is used and the Find condition is satisfied, the current position is stored. Please refer to Find for details.

When parallel processing is used, other processing can be executed in parallel with the motion command.

**Note**

**Using BMove with CP**

The CP parameter causes the arm to move to destination without decelerating or stopping at the point defined by destination. This is done to allow the user to string a series of motion instructions together to cause the arm to move along a continuous path while maintaining a specified speed throughout all the motion. The BMove instruction without CP always causes the arm to decelerate to a stop prior to reaching the point destination.

**See Also**

AccelS, BGo, Find, !....! Parallel Processing, Point Assignment, SpeedS, TGo, Till, TMove, Tool

**BMove Statement Example**

```plaintext
> BMove XY(100, 0, 0, 0) *Move 100 mm in the X direction (in the local coordinate system)

Function BMoveTest

Speed 50
Accel 50, 50
SpeedS 100
AccelS 1000, 1000
Power High

P1 = XY(300, 300, -20, 0)
P2 = XY(300, 300, -20, 0) /L
Local 1, XY(0, 0, 0, 45)

Go P1
Print Here
BMove XY(0, 50, 0, 0)
Print Here

Go P2
Print Here
BMove XY(0, 50, 0, 0)
Print Here

BMove XY(0, 50, 0, 0) /L
Print Here

Fend

[Output]
X:  300.000 Y:  300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y:  350.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y:  300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
X:  300.000 Y:  350.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
X:  264.645 Y:  385.355 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
```
### Boolean Statement

Declares variables of type Boolean. (2 byte whole number).

**Syntax**

```
Boolean varName ((subscripts)) [ , varName ((subscripts))]...
```

**Parameters**

- **varName**
  - Variable name which the user wants to declare as type Boolean.

- **subscripts**
  - Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows
  
  `(ubound1, [ubound2], [ubound3])`

  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

  - Local variable: 2,000
  - Global Preserve variable: 4,000
  - Global variable and module variable: 100,000

**Description**

Boolean is used to declare variables as type Boolean. Variables of type Boolean can contain one of two values, False and True. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

**See Also**

Byte, Double, Global, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UINT64, UShort

**Boolean Statement Example**

```plaintext
Boolean partOK
Boolean A(10)       ' Single dimension array of boolean
Boolean B(10, 10)   ' Two dimension array of boolean
Boolean C(5, 5, 5)  ' Three dimension array of boolean

partOK = CheckPart()
If Not partOK Then
    Print "Part check failed"
EndIf
```
BOpen Statement

Opens file in binary mode.

**Syntax**

```
BOpen fileName As #fileNumber
Close #fileNumber
```

**Parameters**

- **fileName**: String expression that specifies valid path and file name.
  If specifying only a file name, the file must be in the current directory.
  See ChDisk for the details.
- **fileNumber**: Integer expression representing values from 30 to 63.

**Description**

Opens the specified file and identifies it by the specified file number. This statement is used for accessing the specified file in binary mode. If the specified file is not found, it will create a new file. If the file exists, it will read and write the data from the beginning.

Use the ReadBin and WriteBin commands to read and write data in binary mode.

**Note**

A network path is available.

The specified `fileNumber` identifies the file while it is open and cannot be used to refer to a different file until the current file is closed. `fileNumber` is used by other file operations such as ReadBin, WriteBin, Seek, Eof, Flush, and Close.

The read/write position (pointer) of the file can be changed using the Seek command. When switching between read and write access, use Seek to reposition the file pointer.

Use the Close statement to close the file and release the file number.

It is recommended that you use the FreeFile function to obtain the file number so that more than one task are not using the same number.

**See Also**

Close, AOpen, FreeFile, ReadBin, ROpen, UOpen, WOpen, WriteBin

**BOpen Statement Example**

```
Integer fileNum, i

fileNum = FreeFile
BOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    WriteBin #fileNum, i
Next i
Flush #fileNum
Seek #fileNum, 10
ReadBin #fileNum, i
Print "data = ", i
Close #fileNum
```
Box Statement

Specifies and displays the approach check area.

Syntax

1. Box AreaNum [robotNumber], minX, maxX, minY, maxY, minZ, maxZ [localNumber]
2. Box AreaNum, robotNumber, minX, maxX, minY, maxY, minZ, maxZ, remote OutLogic [localNumber]
3. Box AreaNum, robotNumber
4. Box

Parameters

AreaNum Integer expression representing the area number from 1 to 15.

robotNumber Integer expression that specifies which robot you want to configure.

If robotNumber is omitted in syntax (1), the current robot number is used. You cannot omit robotNumber in syntax (2) and (3).

minX The minimum X coordinate position which can be set to the approach check area.

maxX The maximum X coordinate position which can be set to the approach check area.

minY The minimum Y coordinate position which can be set to the approach check area.

maxY The maximum Y coordinate position which can be set to the approach check area.

minZ The minimum Z coordinate position which can be set to the approach check area.

maxZ The maximum Z coordinate position which can be set to the approach check area.

Remote OutLogic On | Off

Set the Remote output logic. To set I/O output to On when the Box approaches, use On. To set I/O output to Off when the Box approaches, use Off. When the parameter is omitted, On will be used.

localNumber Specify the local coordinate system number from 0 to 15.

Be sure to add “/LOCAL” before the number. When the parameter is omitted, the local coordinate system number “0” will be used.

Return Values

When Syntax (3) is used, the area setting of the specified area is displayed.

When Syntax (4) is used, the area settings for all area numbers of the current robot are displayed.

Description

Box is used to set the approach check area. The approach check area is for checking approaches of the robot end effector in the approach check area. The position of the end effector is calculated by the current tool. The approach check area is set on the base coordinate system of the robot or the local coordinate system specified by localNumber, and is between the specified maximum and minimum X, Y, and Z of the specified coordinate system.

When the approach check area is used, the system detects approaches in any motor power status during the controller is ON.

You can also use GetRobotInsideBox function or InsideBox function to get the result of the approach check. GetRobotInsideBox function can be used for wait condition of Wait command. You can provide the check result to the I/O by setting the remote output setting.
When several robots use one area, you should define the area from each robot coordinate system.

Configure the Box 1 from Robot 1 position
Box 1, 1, 100, 200, 0, 100, 0, 100
Lower limit of axes X, Y, Z is (100, 0, 0) and upper limit is (200, 100, 100)

Configure the Box 1 from Robot 2
Box 1, 2, −200, −100, 0, 100, 0, 100
Lower limit of axes X, Y, Z is (−200, 0, 0) and upper limit is (−100, 100, 100)

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Notes

Turning Off Approach Check Area by coordinate axis
You can turn off the approach check area of each coordinate axis. To turn off only the Z axis, define \( \text{minZ} \) and \( \text{maxZ} \) to be 0. For example Box 1, 200, 300, 0, 500, 0, 0.
In this case, it checks if the robot end effector is in the XY dimensional area.

Default values of Approach Check Area
The default values for the Box statement are “0, 0, 0, 0, 0, 0”. (Approach Check Area Checking is turned off.)

Tool Selection
The approach check is executed for the current tool. When you change the tool, the approach check may display the tool approach from inside to outside of the area or the other way although the robot is not operating.

Additional axis
For the robot which has the additional ST axis (including the running axis), the approach check plane to set doesn’t depend on the position of additional axis, but is based on the robot base coordinate system.

Tip
Set Box statement from Robot Manager
EPSON RC+ has a point and click dialog box for defining the approach check area. The simplest method to set the Box values is by using the Box page on the Robot Manager.

See Also
BoxClr, BoxDef, GetRobotInsideBox, InsideBox, Plane
Box Statement Example

<Example 1>
These are examples to set the approach check area using Box statement.

> Box 1, -200, 300, 0, 500, -100, 0

> Box
Box 1: 1, -200.000, 300.000, 0.000, 500.000, -100.000, 0.000, ON /LOCAL0

<Example 2>
The following is a simple program to set the Box values by specifying the local coordinate system numbers 1 and 2.

Function SetBox

 Integer i
  Box 1, -200, 300, 0, 500, -100, 0 /LOCAL1
  i = 2
  Box 2, 100, 200, 0, 100, -200, 100 /LOCAL(i)

Fend
Box Function

Returns the specified approach check area.

Syntax

Box(AreaNum[, robotNumber], limit)

Parameters

AreaNum   Integer expression representing the area number.
robotNumber  Optional. Integer expression that specifies which robot you want to configure.
              If omitted, the current robot number is used.
limit     Integer expression that specifies which limit to return.
           1: Lower limit
           2: Upper limit

Return Values

When you select 1 for limit, the point contains the lower limit of the X, Y, Z coordinates.
When you select 2 for limit, the point contains the upper limit of the X, Y, Z coordinates.

See Also

Box, BoxClr, BoxDef, GetRobotInsideBox, InsideBox

Box Function Example

P1 = Box(1,1)
P2 = Box(1,2)
BoxClr Statement

Clears the definition of approach check area.

**Syntax**

BoxClr AreaNum[, robotNumber]

**Parameters**

- **AreaNum**
  Integer expression representing the area number from 1 to 15.

- **robotNumber**
  Optional. Integer expression that specifies which robot you want to configure. If omitted, the current robot number is used.

**Description**

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

**See Also**

Box, BoxDef, GetRobotInsideBox, InsideBox

**BoxClr Statement Example**

This example uses BoxClr function in a program.

```plaintext
Function ClearBox
    If BoxDef(1) = True Then
        BoxClr 1
    EndIf
EndFunction
```
BoxDef Function

Returns whether Box has been defined or not.

Syntax
BoxDef(AreaNum) [, robotNumber]

Parameters
AreaNum Integer expression representing an area number from 1 to 15.
robotNumber Integer expression representing a robot number you want to configure.
If omitted, the current robot will be specified.

Return Values
True if approach check area is defined for the specified area number, otherwise False.

See Also
Box, BoxClr, GetRobotInsideBox, InsideBox

BoxDef Function Example
This example uses BoxDef function in a program.

    Function ClearBox
        If BoxDef(1) = True Then
            BoxClr 1
        EndIf
    EndFunction

    Function ClearBox
        If BoxDef(1) = True Then
            BoxClr 1
        EndIf
    EndFunction
Brake Statement

Turns brake on or off for specified joint of the current robot.

Syntax

Brake status, jointNumber

Parameters

status The keyword On is used to turn the brake on. The keyword Off is used to turn the brake off.

jointNumber The joint number from 1 to 6.

Description

The Brake command is used to turn brakes on or off for one joint of the 6-axis robot (including N series). It can only be used by input from the command window. This command is intended for use by maintenance personnel only.

When the Brake statement is executed, the robot control parameter is initialized.

See Motor On for the details.

WARNING

Use extreme caution when turning off a brake. Ensure that the joint is properly supported, otherwise the joint can fall and cause damage to the robot and personnel.

Note

Before releasing the brake, be ready to use the emergency stop switch

When the controller is in emergency stop status, the motor brakes are locked. Be aware that the robot arm may fall by its own weight when the brake is turned off with Brake command.

See Also

Motor, Power, Reset, SFree, SLock

Brake Statement Example

> brake on, 1
> brake off, 1
Brake Function

Returns brake status for specified joint.

Syntax

\[ \text{Brake} \ (\text{jointNumber}) \]

Parameters

\[ \text{jointNumber} \] Integer expression representing the joint number. Value are from 1 to the number of joints on the robot.

Return Values

0 = Brake off, 1 = Brake on.

See Also

Brake

Brake Function Example

\[
\text{If } \text{brake}(1) = \text{Off Then}
\text{Print } "\text{Joint 1 brake is off}"
\text{EndIf}
\]
BSet Function

Sets a bit in a number and returns the new value.

**Syntax**

```
BSet (number, bitNum)
```

**Parameters**

- `number` Specifies the value to set the bit with an expression or numeric value.
- `bitNum` Specifies the bit (integer from 0 to 31) to be set by an expression or numeric value.

**Return Values**

Returns the bit set value of the specified numeric value (integer).

**See Also**

- `BClr`, `BClr64`, `BSet64`, `BTst`, `BTst64`

**BSet Function Example**

```plaintext
flags = BSet(flags, 1)
```
BSet64 Function

Sets a bit in a number and returns the new value.

Syntax

**BSet64** (*number, bitNum*)

Parameters

- **number**: Specifies the value to set the bit with an expression or numeric value.
- **bitNum**: Specifies the bit (integer from 0 to 63) to be set by an expression or numeric value.

Return Values

Returns the bit set value of the specified numeric value (integer).

See Also

BClr, BClr64, BSet, BTst, BTst64

**BSet64 Function Example**

```plaintext
flags = BSet64(flags, 1)
```
BTst Function

Returns the status of 1 bit in a number.

Syntax

\[
\text{BTst} \ (\text{number}, \ \text{bitNum})
\]

Parameters

- **number** Specifies the number for the bit test with an expression or numeric value.
- **bitNum** Specifies the bit (integer from 0 to 31) to be tested.

Return Values

Returns the bit test results (integer 1 or 0) of the specified numeric value.

See Also

BClr, BClr64, BSet, BSet64, BTst64

BTst Function Example

\[
\text{If } \text{BTst}(\text{flags, 1}) \ \text{Then} \\
\quad \text{Print } "\text{Bit 1 is set}" \\
\text{EndIf}
\]
BTst64 Function

Returns the status of 1 bit in a number.

Syntax

\[ \text{BTst64 (number, bitNum)} \]

Parameters

- **number**: Specifies the number for the bit test with an expression or numeric value.
- **bitNum**: Specifies the bit (integer from 0 to 63) to be tested.

Return Values

Returns the bit test results (integer 1 or 0) of the specified numeric value.

See Also

- BClr, BClr64, BSet, BSet64, BTst

BTst64 Function Example

```
If BTst64(flags, 1) Then
    Print "Bit 1 is set"
EndIf
```
Byte Statement

Declares variables of type Byte. (2 byte whole number).

Syntax

```
Byte varName [[subscripts]] [ , varName [[subscripts]]... ]
```

Parameters

- **varName**: Variable name which the user wants to declare as type Byte.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows:
  
  \[(ubound1, [ubound2], [ubound3])\]
  
  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

  - Local variable 2,000
  - Global Preserve variable 4,000
  - Global variable and module variable 100,000

Description

Byte is used to declare variables as type Byte. Variables of type Byte can contain whole numbers ranging in value from -128 to +127. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

Boolean, Double, Global, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UInt64, UShort

Byte Statement Example

The following example declares a variable of type Byte and then assigns a value to it. A bitwise And is then done to see if the high bit of the value in the variable test_ok is On (1) or Off (0). The result is printed to the display screen. (Of course in this example the high bit of the variable test_ok will always be set since we assigned the variable the value of 15.)

```
Function Test
    Byte A(10)   ’ Single dimension array of byte
    Byte B(10, 10)   ’ Two dimension array of byte
    Byte C(5, 5, 5)   ’ Three dimension array of byte
    Byte test_ok
    test_ok = 15
    Print "Initial Value of test_ok = ", test_ok
    test_ok = (test_ok And 8)
    If test_ok <> 8 Then
        Print "test_ok high bit is ON"
    Else
        Print "test_ok high bit is OFF"
    EndIf
EndFunction
```
Calib Statement

Replaces the current arm posture pulse values with the current CalPls values.

Syntax

```
Calib joint1, joint2, joint3, joint4, joint5, joint6, joint7, joint8, joint9
```

Parameters

```
joint
```

Integer number from 1 to 9 that specifies the joint number to calibrate. While normally only one joint may need calibration at a time, up to all nine joints may be calibrated with the Calib command at the same time. Additional S axis is 8 and T axis is 9.

Description

Automatically calculates and specifies the offset (Hofs) value. This offset is necessary for matching the origin for each robot joint motor to the corresponding robot mechanical origin.

The Calib command should be used when the motor pulse value has changed. The most common occurrence for use is after changing a motor. Normally, the calibration position pulse values would match the CalPls pulse values. However, after maintenance operations such as changing the motors, these two sets of values will no longer match, and therefore calibration becomes necessary.

Calibration may be accomplished by moving the arm to a desired calibration position, and then executing the Calib command. By executing Calib, the calibration position pulse value is changed to the CalPls value, (the correct pulse value for the calibration position)

In order to perform a proper calibration, Hofs values must be determined. To have Hofs values automatically calculated, move the arm to the desired calibration position, and execute Calib. The controller automatically calculates Hofs values based on the calibration pulse values and on the CalPls pulse values.

Note

Use caution when using the Calib command

Calib is intended to be used for maintenance purposes only. Execute Calib only when necessary. Executing Calib causes the Hofs value to be replaced. Because unintended Hofs value changes can cause unpredictable robot motion, use caution in executing Calib only when necessary.

Potential Error

No Joint Number Specified Error

If the joint number is not specified with the Calib command, an error will occur.

See Also

CalPls, Hofs
Calib Statement Example

Example from the command window.

> CalPls
  Display current CalPls values
  65523, 43320, -1550, 21351

> Pulse
  Display current Pulse values
  PULSE: 1: 65526 pls 2: 49358 pls 3: 1542 pls 4: 21299 pls

> Calib 2
  Execute calibration for joint 2 only

> Pulse
  Display (changed) Pulse values
  PULSE: 1: 65526 pls 2: 43320 pls 3: 1542 pls 4: 21299 pls

>
Call Statement

Calls a user function.

Syntax
Call funcName [(argList)]

Parameters
- **funcName**: The name of a Function which is being called.
- **argList**: Optional. List of arguments that were specified in the Function declaration. For the argument, use the following syntax:
  - `ByRef varName`, or numerical expression
- **ByRef**: Optional. Specify ByRef when you refer to the variable to be seen by the calling function. In this case, the argument change in a function can be reflected to the variable of the calling side. You can change the values received as a reference.

Description
The Call instruction causes the transfer of program control to a function (defined in Function...Fend). This means that the Call instruction causes program execution to leave the current function and transfer to the function specified by Call. Program execution then continues in that function until an Exit Function or Fend instruction is reached. Control is then passed back to the original calling function at the next statement after the Call instruction.

You may omit the Call keyword and argument parentheses. For example, here is a call statement used with or without the Call keyword:

```
Call MyFunc(1, 2)
MyFunc 1, 2
```

You can call an external function in a dynamic link library (DLL). For details, refer to Declare Statement. To execute a subroutine within a function, use GoSub...Return.

You can specify a variable as an argument. Specifying the ByRef parameter, you can reflect the change of argument in the function to the variable of the calling side.

When specifying the ByRef parameter, you need to specify ByRef as well for the argument list of the function definition (Function statement) and DLL function definition (Declare statement).

ByRef is necessary when giving an array variable as an argument.

See Also
- Function, GoSub

Call Statement Example

```
<File1: MAIN.PRG>
  Function main
  Call InitRobot
  Fend

<File2: INIT.PRG>
  Function InitRobot
  If Motor = Off Then
    Motor On
  EndIf
  Power High
  Speed 50
  Accel 75, 75
  Fend
```
CalPls Statement

Specifies and displays the position and orientation pulse values for calibration.

Syntax


(2) CalPls

Parameters

j1Pulses  First joint pulse value.  This is a long integer expression.

j2Pulses  Second joint pulse value.  This is a long integer expression.

j3Pulses  Third joint pulse value.  This is a long integer expression.

j4Pulses  Fourth joint pulse value.  This is a long integer expression.

j5Pulses  Optional.  Fifth joint pulse value.  This is a long integer expression.

j6Pulses  Optional.  Sixth joint pulse value.  This is a long integer expression.

j7Pulses  Optional.  Seventh joint pulse value.  This is a long integer expression.

j8Pulses  Optional.  Eighth joint pulse value.  This is a long integer expression.

j9Pulses  Optional.  Nineth joint pulse value.  This is a long integer expression.

Return Values

When parameters are omitted, displays the current CalPls values.

Description

Specifies and maintains the correct position pulse value(s) for calibration.

CalPls is intended to be used for maintenance, such as after changing motors or when motor zero position needs to be matched to the corresponding arm mechanical zero position.  This matching of motor zero position to corresponding arm mechanical zero position is called calibration.

Normally, the calibration position Pulse values match the CalPls pulse values.  However, after performing maintenance operations such as changing motors, these two sets of values no longer match, and therefore calibration becomes necessary.

Calibration may be accomplished by moving the arm to a certain calibration position and then executing Calib.  By executing Calib, the calibration position pulse value is changed to the CalPls value (the correct pulse value for the calibration position.)

Hofs values must be determined to execute calibration.  To have Hofs values automatically calculated, move the arm to the desired calibration position, and execute Calib.  The controller automatically calculates Hofs values based on calibration position pulse values and on the CalPls values.

Note

CalPls Values Cannot be Changed by cycling power

CalPls values are not initialized by turning main power to the controller off and then on again. The only method to modify the CalPls values is to execute the Calib command.

See Also

Calib, Hofs
CalPls Statement Example
Monitor window operation

> CalPls  "Display current CalPls values" 65523, 43320, -1550, 21351
> Pulse
PULSE: 1: 65526 pls  2: 49358 pls  3: -1542 pls  4: 21299 pls
> Calib 4
> Pulse
PULSE: 1: 65526 pls  2: 49358 pls  3: -1542 pls  4: 21351 pls
>
CalPls Function

Returns calibration pulse value specified by the CalPls Statement.

Syntax

\[ \text{CalPls}(\text{joint}) \]

Parameters

\( \text{joint} \) Integer expression representing a robot joint number or 0 to return CalPls status.
The additional S axis is 8 and T axis is 9.

Return Values

Integer value containing number of calibration pulses. When \( \text{joint} \) is 0, returns 1 or 0 depending on if
CalPls has been executed.

See Also

CalPls

CalPls Function Example

This example uses the CalPls function in a program:

```plaintext
Function DisplayCalPlsValues
    Integer i
    Print "CalPls Values:"
    For i = 1 To 4
        Print "Joint ", i, " CalPls = ", CalPls(i)
    Next i
End
```
ChDir Statement

Changes and displays the current directory.

Syntax
(1) ChDir pathName
(2) ChDir

Parameters
pathName String expression representing the name of the new default path.
See ChDisk for the details.

Description
(1) Changes to the specified directory by specifying the parameter.
(2) When the parameter is omitted, the current directory is displayed. This is used to display the current
directory when it is not known.

ChDir is available only with the PC disk.

When the power is ON, the root directory will be the current directory if no project is open, and if a project
is open, the project directory will be the current directory.

If you change the drive with ChDrive, the root directory will be the current directory.

The parameter is stored in compact flash in controller. Therefore, writing to command flash occurs when
executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We
recommend to use this command minimally.

See Also
ChDrive, ChDisk, CurDir$

ChDir Statement Example
The following examples are done from the command window.

> ChDir \   'Change current directory to the root directory
> ChDir..   'Change current directory to parent dir
> Cd \TEST\H55  'Change current directory to \H55 in \TEST
> Cd A:\TEST\H55\  'Display current directory
ChDisk Statement

Sets the object disk for file operations.

Syntax

ChDisk PC|USB|RAM

Parameters

<table>
<thead>
<tr>
<th></th>
<th>Folders (such as Hard disk) on the Windows Part</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>Folders (such as Hard disk) on the Windows Part</td>
</tr>
<tr>
<td></td>
<td>The initial setting is PC and normally you don’t have to change</td>
</tr>
<tr>
<td></td>
<td>the setting from PC.</td>
</tr>
<tr>
<td></td>
<td>Accesses to the files on the project folders.</td>
</tr>
<tr>
<td>USB</td>
<td>USB memory connected to the controller memory port</td>
</tr>
<tr>
<td></td>
<td>This is useful to exchange files when you don’t use the Windows</td>
</tr>
<tr>
<td></td>
<td>Part (RC+).</td>
</tr>
<tr>
<td>RAM</td>
<td>Temporary files on the memory</td>
</tr>
<tr>
<td></td>
<td>These files are not saves when you turn off the controller.</td>
</tr>
<tr>
<td></td>
<td>This is useful to save the data temporary.</td>
</tr>
</tbody>
</table>

Some of the SPEL+ commands change the object of the file operations according to the ChDisk setting. Also, the ChDisk setting is available only with the PC disk for some commands.

<table>
<thead>
<tr>
<th>ChDisk</th>
<th>Curve</th>
<th>Object is always the project folders.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CVMove</td>
<td>File name can be specified.</td>
</tr>
<tr>
<td></td>
<td>LoadPoints</td>
<td>If path is specified, an error occurs.</td>
</tr>
<tr>
<td></td>
<td>SavePoints</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ImportPoints file name</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ChDisk</th>
<th>Access, Excel file name of OpenDB</th>
<th>Object is always the Windows folders.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ImportPoints source path</td>
<td>If only file name is specified, it can</td>
</tr>
<tr>
<td></td>
<td>VLoadModel</td>
<td>be affected by the current drive and</td>
</tr>
<tr>
<td></td>
<td>VSaveImage</td>
<td>folder.</td>
</tr>
<tr>
<td></td>
<td>VSaveModel</td>
<td>You can also specify a full path.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Executable when ChDisk is PC</th>
<th>ChDir</th>
<th>Object is always the Windows folders.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FolderExists</td>
<td>If only file name is specified, it can be affected by the current drive and folder.</td>
</tr>
<tr>
<td></td>
<td>MkDir</td>
<td>You can also specify a full path.</td>
</tr>
<tr>
<td></td>
<td>RenDir</td>
<td>USB and RAM have no idea of directory.</td>
</tr>
<tr>
<td></td>
<td>RmDir</td>
<td></td>
</tr>
</tbody>
</table>
### ChDisk Statement

<table>
<thead>
<tr>
<th>Executable when ChDisk is USB or RAM</th>
<th>Copy</th>
<th>Del</th>
<th>FileDataTime</th>
<th>FileExist</th>
<th>FileLen</th>
<th>AOpen, BOpen, ROpen, UOpen, WOpen</th>
<th>Rename</th>
<th>When ChDisk is PC:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>If only file name and directory name are specified, it can be affected by the current drive and folder.</td>
<td></td>
<td>You can also specify a full path.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When ChDisk is USB or RAM:</td>
<td></td>
<td>Only file name can be specified and if a path is specified, an error occurs.</td>
</tr>
<tr>
<td>Special</td>
<td>Declare</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>See Declare for the details.</td>
<td></td>
<td>Any specified file name can be accepted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>It cannot be affected by the current drive and folder</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How to decide a full path when ChDisk is PC is as follows:

<table>
<thead>
<tr>
<th>Only file name</th>
<th>“abc.txt”</th>
<th>Current drive + Current directory + Specified file name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full path without a drive</td>
<td>“abc.txt”</td>
<td>Current drive + Specified full path “C:\abc.txt”</td>
</tr>
<tr>
<td>Full path with a drive</td>
<td>“d:\abc.txt”</td>
<td>Specified full path “d:\abc.txt”</td>
</tr>
<tr>
<td>Drive is a network folder</td>
<td>“k:\abc.txt”</td>
<td>Specified full path “k:\abc.txt”</td>
</tr>
<tr>
<td>Network path</td>
<td>“\Epson\data\abc.txt”</td>
<td>Specified full path “\Epson\data\abc.txt”</td>
</tr>
</tbody>
</table>

You can have one ChDisk setting per controller.
If you want to set more than one disk as a system, take an exceptional control to switch the ChDisk setting.

**See Also**
ChDir, ChDrive, CurDisk$%

**ChDisk Statement Example**
Examples from the Command window.

> ChDisk  PC

---

**Notes:**
- EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
- Page: 133
ChDrive Statement

Changes the current disk drive for file operations.

Syntax

ChDrive drive

Parameters

drive String expression or literal containing a valid drive letter.

Description

ChDrive is available only with the PC disk.

When the power is turned on, the “C” drive will be the current drive if a project is closed. If a project is open, the drive of the opened project will be the current drive.

See ChDisk for the details.

The parameter is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also

ChDir, ChDisk, CurDrive$

ChDrive Statement Example

The following examples are done from the command window.

> ChDrive d
ChkCom Function

Returns number of characters in the reception buffer of a communication port.

Syntax

ChkCom ( portNumber As Integer )

Parameters

portNumber

Integer value that specifies the RS-232C port number
Real Part 1 to 8
Windows Part 1001 to 1008

Return Values

Number of characters received (integer).

If the port cannot receive characters, the following negative values are returned to report the current port status:

-2 Port is used by another task
-3 Port is not open

See Also

CloseCom, OpenCom, Read, Write

ChkCom Function Example

Integer numChars

numChars = ChkCom(1)
ChkNet Function

Returns number of characters in the reception buffer of a network port.

Syntax

`ChkNet ( portNumber As Integer )`

Parameters

- `portNumber`: TCP/IP port number (201 to 216)

Return Values

Number of characters received (integer).

If the port cannot receive characters, the following negative values are returned to report the current port status:

- `-1`: Port is open but communication has not been established
- `-2`: Port is used by another task
- `-3`: Port is not open

See Also

CloseNet, OpenNet, Read, Write

ChkNet Function Example

```basic
Integer numChars
numChars = ChkNet(201)
```
Chr$ Function

Returns the character specified by a numeric ASCII value.

Syntax

\[ \text{Chr$}(number) \]

Parameters

- **number**: An integer expression between 1 and 255.

Return Values

Returns a character that corresponds with the specified ASCII code specified by the value of \( \text{number} \).

Description

Chr$ returns a character string (1 character) having the ASCII value of the parameter \( \text{number} \). When the \( \text{number} \) specified is outside of the range from 1 to 255, an error will occur.

See Also

Asc, Instr, Left$, Len, Mid$, Right$, Space$, Str$, Val

Chr$ Function Example

The following example declares a variable of type String and then assigns the string "ABC" to it. The Chr$ instruction is used to convert the numeric ASCII values into the characters "A", "B" and "C". The &H means the number following is represented in hexadecimal form. (&H41 means Hex 41)

```
Function Test
    String temp$
    temp$ = Chr$(&H41) + Chr$(&H42) + Chr$(&H43)
    Print "The value of temp = ", temp$
Fend
```
Erases the robot position data memory.

Syntax

```
ClearPoints
```

Description

ClearPoints initializes the robot position data area. Use this instruction to erase point definitions which reside in memory before teaching new points.

See Also

Plist, LoadPoints, SavePoints

ClearPoints Statement Example

The example below shows simple examples of using the ClearPoints command (from the command window). Notice that no teach points are shown when initiating the Plist command once the ClearPoints command is given.

```
>P1=100,200,-20,0/R
>P2=0,300,0,20/L
>p1=100,200,-20,0/R
>p2=0,300,0,20/L
>clearpoints
>p1=100,200,-20,0/R
>p2=0,300,0,20/L
>plist
>
Close Statement

Closes a file that has been opened with AOpen, BOpen, ROpen, UOpen, or WOpen.

Syntax

```
Close #fileNumber
```

Parameters

- **fileNumber**: Integer expression whose value is from 30 to 63.

Description

Closes the file referenced by file handle `fileNumber` and releases it.

See Also

AOpen, BOpen, Flush, FreeFile, Input #, Print #, ROpen, UOpen, WOpen

Close Statement Example

This example opens a file, writes some data to it, then later opens the same file and reads the data into an array variable.

```plaintext
Integer fileNumber, i, j

fileNumber = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

FileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
    Input #fileNum, j
    Print j
Next i
Close #fileNum
```
CloseCom Statement

Closes the RS-232C port that has been opened with OpenCom.

Syntax

```
CloseCom  #portNumber| All
```

Parameters

- `portNumber` RS-232C port number to close.
  - Real Part: 1 to 8
  - Windows Part: 1001 to 1008
  - If All is specified, the task will close all the open RS-232C ports.

See Also

ChkCom, OpenCom

CloseCom Statement Example

```
CloseCom  #1
```
CloseDB Statement

Closes the database that has been opened with the OpenDB command and releases the file number.

Syntax

```
CloseDB #fileNumber
```

Parameters

- **fileNumber**: Database number specified with OpenDB from 501 to 508

Description

CloseDB closes the database and Excel book, and releases the database number.

Note

- Connection of PC with installed RC+ is required.

See Also

- OpenDB, SelectDB, UpdateDB, DeleteDB, Input #, Print #

CloseDB Statement Example

Refer to OpenDB use example.
CloseNet Statement

Closes the TCP/IP port previously opened with OpenNet.

**Syntax**

```
CloseNet  #portNumber | All
```

**Parameters**

- `portNumber`  
  TCP/IP port number to close (201 to 216)  
  If All is specified, the task will close all the open TCP/IP ports.

**See Also**

- ChkNet, OpenNet

**CloseNet Statement Example**

```
CloseNet  #201
```
Cls Statement

Clears the EPSON RC+ Run, Operator, or Command window text area. Clears also the TP print panel.

Syntax
(1) Cls #deviceID
(2) Cls

Parameters
deviceID 21 RC+
24 TP (TP1 only)
20 TP3
When deviceID is omitted, the display device is cleared.

Description
Cls clears the current EPSON RC+ Run or Operator window text area, depending on where the program was started from.

If Cls is executed from a program that was started from the Command window, the command window text area is cleared.

When deviceID is omitted, the display of the current display device is cleared.

Cls Statement Example
If this example is run from the Run window or Operator window, the text area of the window will be cleared when Cls executes.

Function main
  Integer i
  Do
    For i = 1 To 10
      Print i
    Next i
  Cls
  Loop
Fend
Cnv_AbortTrack Statement

Aborts tracking motion to a conveyor queue point.

Syntax

Cnv_AbortTrack [ stopZheight ]

Parameters

stopZheight  Optional. Real expression that specifies the Z position the robot should move to after aborting the track.

Description

When a motion command to a conveyor queue point is in progress, Cnv_AbortTrack can be executed to abort it.

If stopZHeight is specified, the robot will move up to this value only if the Z axis position at the time of abort is below stopZHeight and will then be decelerated to a stop.

If stopZHeight is omitted, the robot is decelerated to a stop without the depart motion in the Z direction.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_RobotConveyor Statement

Cnv_AbortTrack Statement Example

' Task to monitor robot whose part being tracked has gone downstream

Function WatchDownstream

   Robot 1
   Do
      If g_TrackInCycle And Cnv_QueLen(1, CNV_QUELEN_DOWNSTREAM) > 0 Then
         ' Abort tracking for current robot and move robot Z axis to 0
         g_AbortTrackInCycle = TRUE
         Cnv_AbortTrack 0
         g_AbortTrackInCycle = FALSE
      EndIf
      Wait 0.01
   Loop
Fend
Cnv_Accel Statement

Sets acceleration and deceleration of the tracking motion in the Conveyor Tracking.

Syntax

\[ \text{Cnv\_Accel} \left( \text{conveyorNumber} \right), \text{accel/decel} \]

Parameters

- \text{conveyorNumber} \quad \text{Integer expression representing the conveyor number (1 to 16)}
- \text{accel/decel} \quad \text{Acceleration and deceleration of tracking motion}

Description

Sets acceleration and deceleration of the tracking motion in Conveyor Tracking. Acceleration and deceleration cannot be set separately. Change the parameters when acceleration setting error occurs, or when it is required to reduce work picking time. The default value is 2000[mm/sec^2].

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Accel Function

Cnv_Accel Statement Example

\[ \text{Cnv\_Accel} \ 1,2000 \]
Cnv_Accel Function

Returns acceleration and deceleration of tracking motion in Conveyor Tracking.

Syntax

\[ \text{Cnv\_Accel}\ (\text{conveyorNumber}) \]

Parameters

- \textit{conveyorNumber} Integer expression representing the conveyor number (1 to 16)

Return Values

- Real value in millimeters.

Note

- This command will only work if the Conveyor Tracking option is active.

See Also

- \text{Cnv\_Accel}

Cnv_Accel Function Example

\[ \text{Print}\ \text{Cnv\_Accel}\ (1) \]
Cnv_Downstream Statement

Sets the downstream limit of the specified conveyor.

Syntax

```plaintext
Cnv_Downstream (conveyorNumber) , lowerLimit
```

Parameters

- `conveyorNumber` Integer expression representing the conveyor number (1 to 16)
- `lowerLimit` A border on the downstream side of the tracking area

Return Values

By using Cnv_Downstream, you can change the downstream limit which was set in the calibration wizard. However, if skewed downstream limit is used, you cannot change the value by Cnv_Downstream.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Upstream

Cnv_Downstream Statement Example

```plaintext
Cnv_Downstream 1, 500
```
**Cnv_Downstream Function**

Returns the downstream limit for the specified conveyor.

**Syntax**

```
Cnv_Downstream (conveyorNumber )
```

**Parameters**

- `conveyorNumber` Integer expression representing the conveyor number (1 to 16)

**Return Values**

Real value in millimeters.

**Note**

This command will only work if the Conveyor Tracking option is active.

**See Also**

- Cnv_Upstream

**Cnv_Downstream Function Example**

```
Print "Downstream limit: ", Cnv_Downstream(1)
```
Cnv_Fine Statement

Sets the value of Cnv_Fine for one conveyor.

Syntax

```plaintext
Cnv_Fine  conveyorNumber [, fineValue]
```

Parameters

- `conveyorNumber` Integer expression representing the conveyor number (1 to 16)
- `fineValue` Optional. Real expression that specifies the distance at which tracking is completed in millimeters. A value of 0 means that Cnv_Fine is not used.
  If omitted, the current Cnv_Fine setting is displayed.

Description

After confirming the tracking operation is complete, specify the distance from the part that is acceptable for the next command. When specifying “0”, the Cnv_Fine setting will not be used and the next command will be accepted when the motion command is complete.

The default value of “0” mm is automatically set when the following conditions occur:
- Conveyor is created.
- Controller is started.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_Fine Function

Cnv_Fine Statement Example

```plaintext
Cnv_Fine  1, 5
```
Cnv_Fine Function

Returns the current Cnv_Fine setting.

Syntax

\[
\text{Cnv\_Fine}\ (\text{conveyorNumber})
\]

Parameters

\text{conveyorNumber} \quad \text{Integer expression representing the conveyor number (1 to 16).}

Return Values

Real value of Cnv_Fine in millimeters.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Fine Statement

Cnv_Fine Function Example

\[
\text{Real } f
\]

\[
f = \text{Cnv\_Fine}(1)
\]
Cnv_Flag Function

Returns the tracking state of the robot.

Syntax

Cnv_Flag (conveyorNumber)

Parameters

conveyorNumber Integer expression representing the conveyor number (1 to 16).

Return Values

0: Tracking is not canceled or aborted.
1: Tracking has been canceled.
   The downstream limit position is improper. Set the downstream limit closer to the upstream than the current position.
2: Tracking has been aborted.
   The downstream limit position or the robot waiting position is improper. Set the downstream limit closer to the upstream than the current position or move the robot waiting position closer to the downstream limit.
3: Tracking has been aborted.
   The downstream limit position or picking time is improper. Set the downstream limit closer to the upstream than the current position, or shorten the work picking time.
4: Tracking has been canceled.
   The number of work pieces is exceeding the processing capacity of the robot.

The return values other than “0” are returned only when the tracking abort line is defined. When the value other than “0” is displayed, it is recommended to take the above-described countermeasures for each return value. For details on the tracking abort line, refer to the User’s Guide.

Note

This command will only work if the Conveyor Tracking option is active.

Cnv_Flag Function Example

Print Cnv_Flag (1)
Cnv_LPulse Function

Returns the pulse value latched by the conveyor trigger.

Syntax

    Cnv_LPulse (conveyorNumber)

Parameters

    ConveyorNumber  Integer expression that specifies the conveyor number (1 to 16)

Description

Returns the latest conveyor pulses latched by the hardware trigger wires or Cnv_Trigger.

Return Values

Long value that contains the latched pulses of the specified conveyor.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

    Cnv_Trigger, Cnv_Pulse

Cnv_LPulse Function Example

    Print "Latched conveyor position: ", Cnv_LPulse(1)
Cnv_Mode Statement

Sets a tracking mode of Conveyor Tracking.

Syntax

Cnv_Mode (conveyorNumber, modeNumber)

Parameters

- **conveyorNumber**: Integer expression that specifies the conveyor number (1 to 16)
- **modeNumber**: 0: Picking quantity-priority mode  
  1: Picking accuracy-priority mode  
  2: Variable speed conveyor mode

Description

Sets a tracking mode of Conveyor Tracking.

Cnv_Mode is only available for linear conveyors.

Sets the tracking mode before starting the tracking motion. If the parameters are not set or the conveyor speed is 350 mm/sec or faster, the picking quantity priority mode will be set.

Picking quantity-priority mode: Although this mode is inferior in picking accuracy to the picking accuracy-priority mode, it takes less time to catch up with the moving work pieces. Therefore, this mode is suitable for the conveyor systems in which space between the work pieces is narrow or the fast-speed conveyor systems.

Picking accuracy-priority mode: Although this mode takes longer time to catch up with the work pieces compared to the picking quantity-priority mode, this improves the picking accuracy. Therefore, this mode is suitable for the conveyor systems for small work pieces.

Variable speed conveyor mode: This mode can be used for conveyors which repeats stops and moves randomly. It also can be used for conveyors move at constant speed. However, this mode is inferior in picking quantity to the Picking-quantity mode and inferior in accuracy to the Picking accuracy-priority mode.

The modes “0” and “1” are only supported by the circular conveyors. When “2” is specified, the manipulator moves as same as the mode “0”.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Mode Function

Cnv_Mode Statement Example

    Cnv_Mode 1, 1
Cnv_Mode Function

Returns a tracking mode of Conveyor Tracking.

Syntax

\[
\text{Cnv\_Mode}(\text{conveyorNumber})
\]

Parameters

- \text{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)

Return Values

- Returns a real value from 0 to 2.
- 0: Picking quantity-priority mode
- 1: Picking accuracy-priority mode
- 2: Variable speed conveyor mode

Note

This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_Mode Statement

Cnv_Mode Function Example

\[
\text{Print Cnv\_Mode (1)}
\]
Cnv_Name$ Function

Returns the name of the specified conveyor.

Syntax
   Cnv_Name$ (conveyorNumber)

Parameters
   conveyorNumber   Integer expression that specifies the conveyor number (1 to 16)

Return Values
   A string containing the conveyor name.

Note
   This command will only work if the Conveyor Tracking option is active.

See Also
   Cnv_Number

Cnv_Name$ Function Example

   Print "Conveyor 1 Name: ", Cnv_Name$(1)
Cnv_Number Function

Returns the number of a conveyor specified by name.

Syntax

\[
\text{Cnv\_Number (conveyorName)}
\]

Parameters

\[\text{conveyorName} \quad \text{String expression representing the conveyor name.}\]

Return Values

Integer conveyor number.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv\_Name$

Cnv\_Number Function Example

\[
\text{Integer cnvNum}
\]

\[
\text{cnvNum} = \text{Cnv\_Number("Main Conveyor")}
\]
Cnv_OffsetAngle Statement

Sets the offset value for the conveyor queue data.

Syntax

\[ \text{Cnv\_OffsetAngle \ conveyorNumber [, offsetAngle]} \]

Parameters

- \text{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)
- \text{offsetAngle} Real value representing the offset value for the conveyor queue data (unit: degree).
  Optional. If omitted, the current offset is displayed.

Description

Sets the offset value for the conveyor queue data.
Cnv_OffsetAngle is available for the circular conveyor.
Conveyor Tracking may have tracking delay according to the conveyor speed. If the tracking delay is occurred, the robot handles the parts in the wrong position moved by the tracking delay.
Cnv_OffsetAngle gives the offset value to the queue in order to move the robot back to the correct position.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_OffsetAngle Function

Cnv_OffsetAngle Statement Example

\[ \text{Cnv\_OffsetAngle 1, 5} \]
Cnv_OffsetAngle Function

Returns the offset value of the conveyor queue data.

Syntax

```
Cnv_OffsetAngle (conveyorNumber)
```

Parameters

- `conveyorNumber`  Integer expression that specifies the conveyor number (1 to 16)

Return Values

- Integer value (unit: degree).

Note

- This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_OffsetAngle Statement

Cnv_OffsetAngle Function Example

```
Real offsetAngle

offsetAngle = Cnv_OffsetAngle (1)
```
Cnv_Point Function

Returns a robot point in the specified conveyor's coordinate system derived from sensor coordinates.

Syntax

Cnv_Point(conveyorNumber, sensorX, sensorY [, sensorU])

Parameters

- **conveyorNumber**: Integer expression that specifies the conveyor number (1 to 16)
- **sensorX**: Real expression for the sensor X coordinate.
- **sensorY**: Real expression for the sensor Y coordinate.
- **sensorU**: Optional. Real expression for the sensor U coordinate.

Return Values

Robot point in conveyor coordinate system.

Description

The Cnv_Point function must be used to create points that can be added to a conveyor queue. For vision conveyors, **sensorX** and **sensorY** are the vision coordinates from the camera. For sensor conveyors, **sensorX** and **sensorY** can be 0, since this is the origin of the conveyor's coordinate system.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_Speed

Cnv_Point Function Example

```plaintext
Boolean found
Integer i, numFound
Real x, y, u

cnv_trigger 1
vrun findparts
vget findparts.part.numberfound, numfound
for i = 1 to numfound
    vget findparts.part.cameraXYU(i), found, x, y, u
    cnv_queadd 1, Cnv_Point(1, x, y)
ext i
```
Cnv_PosErr Function

Returns deviation in current tracking position compared to tracking target.

Syntax

\[
\text{Cnv\_PosErr (conveyorNumber)}
\]

Parameters

- \text{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)

Return Values

Real value in millimeters.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_MakePoint

Cnv_PosErr Function Example

Print "Conveyor 1 position error: ", Cnv_PosErr(1)
Cnv_Pulse Function

Returns the current position of a conveyor in pulses.

Syntax

\function{Cnv_Pulse}{conveyorNumber}

Parameters

\var{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)

Return Values

Long value of current pulses for specified conveyor.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Trigger, Cnv_LPulse

Cnv_Pulse Function Example

```
Print "Current conveyor position: ", Cnv_Pulse(1)
```
Cnv_QueAdd Statement

Adds a robot point to a conveyor queue.

Syntax
Cnv_QueAdd conveyorNumber, pointData [, userData ]

Parameters
- **conveyorNumber**: Integer expression that specifies the conveyor number (1 to 16)
- **pointData**: The robot point to add to the conveyor queue.
- **userData**: Optional. Real expression used to store user data along with the point.

Description
- **pointData** is added to the end of the specified conveyor’s queue. It is registered together with the currently latched conveyor pulse position.

- If the distance between **pointData** and the previous point in the queue is at or below that specified by Cnv_QueReject, the point data will not be added to the queue, and no error will occur.

- The maximum queue data value is 1000.

Note
- This command will only work if the Conveyor Tracking option is active.

See Also
- Cnv_RobotConveyor

Cnv_QueAdd Statement Example

```plaintext
Boolean found
Integer i, numFound
Real x, y, u

Cnv_Trigger 1
VRun FindParts
VGet FindParts.Part.NumberFound, numFound
For i = 1 To numFound
    VGet FindParts.Part.CameraXYU(i), found, x, y, u
    Cnv_QueAdd 1, Cnv_Point(1, x, y)
Next i
```
**Cnv_QueGet Function**

Returns a point from the specified conveyor's queue.

**Syntax**

Cnv_QueGet (conveyorNumber [, index ])

**Parameters**

- **conveyorNumber**: Integer expression that specifies the conveyor number (1 to 16)
- **index**: Optional. Integer expression representing the index of the queue data to retrieve.

**Return Values**

A robot point in the specified conveyor's coordinate system.

**Description**

Use Cnv_QueGet to retrieve points from the conveyor queue. When `queNumber` is omitted, the first point in the queue is returned. Otherwise, the point from the specified `queNumber` is returned.

Cnv_QueGet does not delete the point from the queue. Instead, you must use Cnv_QueRemove to delete it.

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.

**Note**

This command will only work if the Conveyor Tracking option is active.

**See Also**

Cnv_QueLen, Cnv_QueRemove

**Cnv_QueGet Function Example**

```
' Jump to the first part in the queue and track it
Jump  Cnv_QueGet(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1
Cnv_QueRemove 1
```
Cnv_QueLen Function

Returns the number of items in the specified conveyor's queue.

Syntax

Cnv_QueLen (conveyorNumber [, paramNumber ])

Parameters

conveyorNumber Integer expression that specifies the conveyor number (1 to 16)

paramNumber Optional. Integer expression that specifies which data to return the length for.

Symbolic constant | Value | Meaning
------------------|-------|---------------------
CNV_QUELEN_ALL    | 0     | Returns total number of items in queue.
CNV_QUELEN_UPSTREAM | 1     | Returns number of items upstream.
CNV_QUELEN_PICKUPAREA | 2     | Returns number of items in pickup area.
CNV_QUELEN_DOWNSTREAM | 3     | Return number of items downstream.

Return Values

Integer number of items.

Description

Cnv_QueLen is used to find out how many items are available in the queue. Typically, who will want to know how many items are in the pickup area.

You can also use Cnv_QueLen as an argument to the Wait statement.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_QueGet

Cnv_QueLen Function Example

Do
    Do While Cnv_QueLen (1, CNV_QUELEN_DOWNSTREAM) > 0
        Cnv_QueRemove 1, 0
    Loop
    If Cnv_QueLen (1, CNV_QUELEN_PICKUPAREA) > 0 Then
        Jump Cnv_QueGet (1, 0) C0
        On gripper
        Wait .1
        Cnv_QueRemove 1, 0
        Jump place
        Off gripper
        Jump idlePos
    EndIf
Loop
Cnv_QueList Statement

Displays a list of items in the specified conveyor's queue.

Syntax

\[ \text{Cnv\_QueList \ conveyorNumber}, \ \text{numOfItems} ] \]

Parameters

- \textit{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)
- \textit{numOfItems} Optional. Integer expression to specify how many items to display. If omitted, all items are displayed.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_QueGet

Cnv_QueList Statement Example

\[ \text{Cnv\_QueList} \ 1 \]
Cnv_QueMove Statement

Moves data from upstream conveyor queue to downstream conveyor queue.

**Syntax**

```
Cnv_QueMove conveyorNumber [, index ] [, userData ]
```

**Parameters**

- `conveyorNumber`  
  Integer expression that specifies the conveyor number (1 to 16)
- `index`  
  Optional. Integer expression that specifies the index of the queue to move.  
  (The first item in the queue is index #0.)
- `userData`  
  Optional. Real expression used to store user data along with the item.

**Description**

Cnv_QueMove is used to move one or more items from a conveyor queue to its associated downstream conveyor queue. If `index` is specified, the first item (`index #0`) of the queue is moved.

**Note**

This command will only work if the Conveyor Tracking option is active.

**See Also**

Cnv_QueGet

**Cnv_QueMove Statement Example**

```
Cnv_QueMove 1
```
Cnv_QueReject Statement

Sets and displays the queue reject distance for a conveyor.

Syntax

\texttt{Cnv\_QueReject conveyorNumber [, rejectDistance ]}

Parameters

- \textit{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)
- \textit{rejectDistance} Optional. Real expression specifying the minimum distance between parts allowed in the queue in millimeters. If a negative value is specified, 0 mm will be set. If omitted, the current \textit{rejectDistance} is displayed.

Description

Use \texttt{Cnv\_QueReject} to specify the minimum distance between parts to prevent double registration in the queue. As parts are scanned by the vision system, they will be found more than once, but they should only be registered once. \texttt{Cnv\_QueReject} helps the system filter out double registration.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

\texttt{Cnv\_QueReject Function}

Cnv_QueReject Statement Example

\begin{verbatim}
   Cnv_QueReject 1, 20
\end{verbatim}
Cnv_QueReject Function

Returns the current part reject distance for a conveyor.

Syntax

\[ \text{Cnv\_QueReject (conveyorNumber)} \]

Parameters

- \textit{conveyorNumber} Integer expression that specifies the conveyor number (1 to 16)

Return Values

Real value in millimeters.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_QueReject Statement

Cnv_QueReject Function Example

\begin{verbatim}
Real rejectDist
RejectDist = Cnv_QueReject(1)
\end{verbatim}
Cnv_QueRemove Statement

Removes items from a conveyor queue.

Syntax

Cnv_QueRemove conveyorNumber [, index | All ]

Parameters

conveyorNumber  Integer expression that specifies the conveyor number (1 to 16)
index  Optional. Integer expression specifying the index of the first item to remove or specify All to remove all.

Description

Use Cnv_QueRemove to remove one or more items from a conveyor queue. Typically, you remove items from the queue after you are finished with the data.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_QueAdd

Cnv_QueRemove Statement Example

Jump Cnv_QueGet(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1

' Remove the data from the conveyor
Cnv_QueRemove 1
Cnv_QueUserData Statement

Sets and displays user data associated with a queue entry.

Syntax

Cnv_QueUserData conveyorNumber [ , index ] [ , userData ]

Parameters

- **conveyorNumber**: Integer expression that specifies the conveyor number (1 to 16)
- **index**: Optional. Integer expression specifying the index of the item number in the queue.
- **userData**: Optional. Real expression specifying user data.

Description

Cnv_QueUserData is used to store your own data with each item in a conveyor queue. User data is optional. It is not necessary for normal operation.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_QueUserData Function

Cnv_QueUserData Statement Example

Cnv_QueUserData 1, 1, angle
Cnv_QueUserData Function

Returns the user data value associated with an item in a conveyor queue.

Syntax

\[ \text{Cnv\_QueUserData} \left( \text{conveyorNumber} \ [, \text{index}] \right) \]

Parameters

- \text{conveyorNumber} \quad \text{Integer expression that specifies the conveyor number (1 to 16)}
- \text{index} \quad \text{Optional. Integer expression specifying the index of the item number in the queue.}

Return Values

Real value.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

- Cnv_QueUserData Statement

Cnv_QueUserData Function Example

' Add to queue
Cnv_QueAdd 1, Cnv_Point(1, x, y), angle

' Remove from queue
angle = \text{Cnv\_QueUserData}(1) \quad \text{default to queue index of 0}
Jump Cnv_QueGet(1) :U(angle)
Cnv_QueRemove 1
Cnv_RobotConveyor Function

Returns the conveyor being tracked by a robot.

Syntax

Cnv_RobotConveyor [ ( robotNumber ) ]

Parameters

robotNumber  Integer expression representing the robot number.

Return Values

Integer conveyor number.  0 = no conveyor being tracked.

Description

When using multiple robots, you can use Cnv_RobotConveyor to see which conveyor a robot is currently tracking.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_MakePoint Statement

Cnv_RobotConveyor Function Example

    Integer cnvNum
    cnvNum = Cnv_RobotConveyor(1)
Cnv_Speed Function

Returns the current speed of a conveyor.

Syntax

Cnv_Speed (conveyorNumber)

Parameters

conveyorNumber  Integer expression that specifies the conveyor number (1 to 16)

Return Values

For straight conveyors, a real value in millimeters per second. For circular conveyors, a real value in degrees per sec.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Pulse

Cnv_Speed Statement Example

Print "Conveyor speed: ", Cnv_Speed(1)
Cnv_Trigger Statement

Latches current conveyor position for the next Cnv_QueAdd statement.

Syntax

Cnv_Trigger conveyorNumber

Parameters

conveyorNumber  Integer expression that specifies the conveyor number (1 to 16)

Description

Cnv_Trigger is a software trigger command that must be used if there is no hardware trigger wired to the
PG board for the conveyor encoder.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_QueAdd

Cnv_Trigger Statement Example

Boolean found
Integer i, numFound
Real x, y, u

Cnv_Trigger 1
VRun FindParts
VGet FindParts.Part.NumberFound, numFound
For i = 1 To numFound
  VGet FindParts.Part.CameraXYU(i), found, x, y, u
  Cnv_QueAdd 1, Cnv_Point(1, x, y)
Next i
Cnv_Upstream Statement

Sets the upstream limit of the specified conveyor.

Syntax

Cnv_Upstream (conveyorNumber, upperLimit)

Parameters

- **conveyorNumber**: Integer expression representing the conveyor number (1 to 16)
- **upperLimit**: A border on the upstream side of the tracking area

Return Values

By using Cnv_Upstream, you can change the upstream limit which was set in the calibration wizard. However, if skewed upstream limit is used, you cannot change the value by Cnv_Upstream.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Downstream

Cnv_Upstream Statement Example

Cnv_Upstream 1,200
Cnv_Upstream Function

Returns the upstream limit for the specified conveyor.

Syntax

\[
\text{Cnv\_Upstream} (\text{conveyorNumber})
\]

Parameters

\text{conveyorNumber} \quad \text{Integer expression that specifies the conveyor number (1 to 16)}

Return Values

Real value in millimeters.

Note

This command will only work if the Conveyor Tracking option is active.

See Also

Cnv_Downstream

Cnv_Upstream Function Example

Print "Upstream limit: ", Cnv_Upstream(1)
CollisionDetect Statement

Enables or disables the collision detection (detetction of robot motion error) of the current robot.

Syntax
(1) CollisionDetect status
(2) CollisionDetect status, jointNumber
(3) CollisionDetect

Parameters
status On: Enables the collision detection (detetction of robot motion error).
      Off: Disables the collision detection (detetction of robot motion error).

jointNumber SCARA robots (including RS series): Specify the joint by a joint number from 1 to 4
           Vertical 6-axis robots (including N series): Specify the joint by a joint number from 1 to 6

Result
Returns the current CollisionDetect status when the parameters are omitted.

Description
Detect the robot motion error from differentiation between desired speed and the actual speed (speed deviation value). Errors can be detected by this function is classified into A and B.
   A: Collision or contact of robot arm or hand occurs
   B: Robot motion errors other than collision or contact

Also, error B is classified into below according to the power condition.

Error in high power
   Torque saturation due to little setting of Weight or Inertia.
   Torque saturation due to combined motion of multiple joints and throwing around the long object.
   Torque saturation due to supply voltage reduction.
   Error motion due to hardware error or software malfunction.

Error in low power
   Error motion due to hardware error or software malfunction.
   Torque saturation in low power due to holding a hand or long object that exceeds the weight described in the specifications.

The collision detection is available for the general-purpose robots supported by the EPSON RC+ 7.0 Ver.7.2 or later (vertical 6-axis and SCARA robots). If this command is used while unsupported robot (X5 series, etc.) is connected, an error occurs.
Execution of this command takes a little time. If cycle time is prioritized, minimize the use of this command in the program.
This function can be enabled or disabled for each joint or all joints. The default is “all joints on”.
(The default is off if the firmware version is before Ver 7.2.0.x.)
The setting returns to the default when the Controller is turned off. In other cases, the setting does not change unless otherwise configured by this command explicitly.
Output the following messages and stop the robot when the collisition is detected.
   Error 5057 “Collision was detected in High power mode” (detetction of robot motion error).
   Error 5058 “Collision was detected in Low power mode” (detetction of robot motion error).

For reducing damage in High power mode, using the command together with the upper limit torque restriction by LimitTorque is also effective. For reducing damage in Low power mode, using the command together with the upper limit torque restriction by LimitTorqueLP is also effective.
CollisionDetect Statement

Also refer to EPSON RC+ 7.0 User’s Guide “6.18.10 Collision Detection Function (detection of robot motion error)”.

See Also

LimitTorque, LimitTorque Function, LimitTorqueLP, LimitTorqueLP Function

CollisionDetectStatement Example

<table>
<thead>
<tr>
<th>CollisionDetect</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>' Turns On the collision detection for all joints</td>
</tr>
<tr>
<td>Off, 5</td>
<td>' Turns On the collision detection for only Joint #5</td>
</tr>
<tr>
<td></td>
<td>' The result will be displayed as “on, on, on, off, on”</td>
</tr>
</tbody>
</table>
CollisionDetect Function

Returns the setting value of CollisionDetect command.

Syntax

CollisionDetect(jointNumber)

Parameters

jointNumber Specify the joint by a joint number from 1 to 6.

Return Values

Returns the setting value of CollisionDetect command by an integer.
1 = ON
0 = OFF

See Also

CollisionDetect

CollisionDetect Function Example

Print CollisionDetect(1) ' Displays CollisionDetect value of the Joint #1.
Cont Statement

Resumes the controller after a Pause statement has been executed and continues the execution of all tasks. This command is for the experienced user and you need to understand the command specification before the use.

Syntax

Cont

Description

To execute the Cont statement from a program, you need to set the [Enable advanced task commands] checkbox in Setup | System Configuration | Controller | Preferences page of the EPSON RC+. However, even if this preference is enabled, you cannot execute the Cont statement from a task executed by Trap SGClose.

The Cont command resumes the controller tasks paused by the Pause statement or safeguard open and continues all tasks execution. It has the same function as the <Continue> button on the Run Window, Operator Window, and the Continue Remote input.

If you execute the Cont command during WaitRecover status (waiting for the recover after safeguard open), it will turn on all the robot motors and execute the recover motion. Then, the program will be resumed. If you just want to turn on motors and execute recover motion, use the Recover command.

<table>
<thead>
<tr>
<th>CAUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ When executing Cont command from a program, you must understand the command specification and confirm that the system has the proper conditions for the Cont command. Improper use such as continuous execution of a command within a loop may deteriorate the system safety.</td>
</tr>
</tbody>
</table>

See Also

Pause, Recover

Cont Statement Example

```c
Function main
    Xqt 2, monitor, NoPause
    Do
        Jump P1
        Jump P2
    Loop
    Fend

Function monitor
    Do
        If Sw(pswitch) = On then
            Pause
            Wait Sw(pswitch) = Off and Sw(cswitch) = On
            Cont
            EndIf
    Loop
    Fend
```
Copy Statement

Copies a file to another location.

**Syntax**

```
Copy source, destination
```

**Parameters**

- **source**: Pathname and filename of the source location of the file to copy. See ChDisk for the details.
- **destination**: Pathname and filename of the destination to copy the specified source file to. See ChDisk for the details.

**Description**

Copies the specified `source` filename to the specified `destination` filename.

The same pathname and filename may not be specified for both source and destination files. An error occurs if the destination already exists.

**Note**

A network path is available.

Wildcard characters (*, ?) are not allowed in specified filenames.

When used in the Command window, quotes and comma may be omitted.

**See Also**

ChDir, MkDir

**Copy Command Example**

The following example is done from the Command window.

```
> copy TEST.DAT TEST2.DAT

> Copy TEST.DAT c: 'NG
!! Error: 7203 Access is denied.
> Copy TEST.DAT c:\ 'OK

> }
```
Cos Function

Returns the cosine of a numeric expression.

Syntax

\[
\text{Cos} \ (\text{number})
\]

Parameters

\text{number} \quad \text{Numeric expression in Radians.}

Return Values

Numeric value in radians representing the cosine of the numeric expression \text{number}.

Description

Cos returns the cosine of the numeric expression. The numeric expression (\text{number}) must be in radian units. The value returned by the Cos function will range from -1 to 1

To convert from degrees to radians, use the DegToRad function.

See Also

Abs, Atan, Atan2, Int, Mod, Not, Sgn, Sin, Sqr, Str$, Tan, Val

Cos Function Example

The following example shows a simple program which uses Cos.

\begin{verbatim}
Function costest
    Real x
    Print "Please enter a value in radians"
    Input x
    Print "COS of \( x \) is \( \cos(x) \)
Fend
\end{verbatim}

The following examples use Cos from the Command window.

Display the cosine of 0.55:

```
>print \text{cos}(0.55)
0.852524522059506
>
```

Display cosine of 30 degrees:

```
>print \text{cos}(\text{DegToRad}(30))
0.866025403784439
>
```
CP Statement

Sets CP (Continuous Path) motion mode.

Syntax
CP { On | Off }

Parameters
On | Off  The keyword On is used to enable path motion. The keyword Off is used to disable CP mode.

Description
CP (Continuous Path) motion mode can be used for the Arc, Arc3, Go, Jump, Jump3, Jump3CP, JumpTLZ, and Move robot motion instructions.

When CP mode is On, each motion command executes the next statement as deceleration starts. Continuous path motion will continue regardless of whether the CP parameter is specified in each motion command or not.
When CP is Off, this function is active only when the CP parameter is specified in each motion command.

![Graph showing speed vs. time for normal motion and path motion.]

When CP is On, path motion will continue without full deceleration between two CP motions (Arc, Arc3, Jump3, Jump3CP, JumpTLZ, and Move), or two PTP motions (Go, Jump). In contrast, full deceleration will occur between a CP motion and a PTP motion.

CP will be set to Off in the following cases

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

See Also
CP Function, Arc, Arc3, Go, Jump, Jump3, Jump3CP, JumpTLZ, Move

CP Statement Example
CP On
Move P1
Move P2
CP Off

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 183
CP Function

Returns status of path motion.

Syntax

CP

Return Values

0 = Path motion off, 1 = Path motion on.

See Also

CP Statement

CP Function Example

    If CP = Off Then
        Print "CP is off"
    EndIf
CP_Offset Statement

Sets the offset time to start the subsequent motion command when executing CP On.

Syntax
(1) CP_Offset [ On [, OffsetTime ] ]
(2) CP_Offset Off

Parameters
On | Off
On: Enables the motion command start offset function in CP On. If omitted, current setting will be displayed.
Off: Disables the motion command start offset function in CP On.

OffsetTime
Specify the offset time to start the subsequent command in CP On by a real value from 10 to 24 (unit: ms). If omitted, the default value (10 ms) will be set.

Description
CP_Offset is available for following commands.
Move, Arc, Arc3, CVMove

If the CP parameter is added to CP On or motion commands, the subsequent command will be executed at the same time as the prior motion starts decelerating. As a result, the motions become a path motion as shown below, where deceleration of the first command and acceleration of the subsequent command overlap.
At this moment, the start of deceleration for the first command and the start of acceleration for the subsequent command are not strictly simultaneous due to the processing overhead time for starting the statement. Therefore, the speed declines at the switching point in the path motion, and the motion will not be constant velocity.
CP_Offset solves this problem by accelerating the starting time of the subsequent motion command.

By setting CP_Offset on, the processing start of the subsequent motion command will be accelerated by the time specified for the OffsetTime parameter, and deceleration start of the actual robot and acceleration start of the subsequent command will be synchronized. As a result, the constant velocity can be improved. The OffsetTime parameter is set by default. Adjust the parameter according to your application. Especially when the subsequent motion command has “!Parallel Processing!”, the overhead time required for the motion start gets longer. Therefore, set the OffsetTime parameter higher than the default value, approximately 16 ms.

To set the OffsetTime parameter for CP_Offset, measure the speed of the tool center point for the target motion by using TCPSpeed. Setting an appropriate value for the OffsetTime parameter improved the motion at the switching point to be close to constant.
TCPSpeed increases when OffsetTime is too large, and TCPSpeed decreases when OffsetTime is too small. Adjustment of CP_Offset must be done in actual system. Appropriate adjustment cannot be done in the simulator because the processing time to start the command differs from the actual controller.
CP_Offset Statement

Sample program for measurement using TCPSpeed

```plaintext
Function main
    Motor On
    Power High
    SpeedS 250; AccelS 1500
    Speed 50; Accel 50, 50
    Go XY(300, 500, 500, 90, 0, 180)
    CP_Offset On
    Xqt printTcPSpeed
    Move XY(0, 500, 500, 90, 0, 180) CP
    Move XY(-300, 500, 500, 90, 0, 180)
    Quit printTcPSpeed
    CP_Offset Off
Fend

Function printTcPSpeed
    Do
        Print TCPSpeed
    Loop
Fend

Example of OffsetTime adjustment

This command is not intended for PTP motion. In PTP motion, the motion will be an usual path motion.

CP_Offset is off when any of the following conditions occur:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or Quit All stops tasks
```
See Also
   CP_Offset Function, CP, Move, Arc, Arc3, CVMove

CP_Offset Statement Example

   CP_Offset  On
   Move P1
   Move P2
   CP_Offset  Off
CP_Offset Function

Returns the offset time to start the subsequent motion command when executing CP On.

Syntax

CP_Offset

Return Values

Real number representing the offset time to start the motion command.

See Also

CP_Offset Statement

CP_Offset Function Example

If CP_Offset = 0 Then
  Print "CP_Offset is off"
EndIf
**Ctr Function**

Returns the counter value of the specified Hardware Input counter.

**Syntax**

```
Ctr(bitNumber)
```

**Parameters**

- `bitNumber`  
  Number of the Hardware Input bit set as a counter. Only 16 counters can be active at the same time.

**Return Values**

The current count of the specified Hardware Input Counter. (Integer expression from 0-65535)

**Description**

Ctr works with the CTReset statement to allow Hardware inputs to be used as counters.

Each time a hardware input specified as a counter is switched from the Off to On state that input causes the counter to increment by 1.

The Ctr function can be used at any time to get the current counter value for any counter input. Any of the Hardware Inputs can be used as counters. However, only 16 counters can be active at the same time.

**Counter Pulse Input Timing Chart**

<table>
<thead>
<tr>
<th></th>
<th>High (ON)</th>
<th>Low (OFF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timer</td>
<td>4 msec or longer</td>
<td>4 msec or longer</td>
</tr>
</tbody>
</table>

**See Also**

CTReset

**Ctr Function Example**

The following example shows a sample of code which could be used to get a hardware input counter value.

```
CTReset 3 ' Reset counter for input 3 to 0
On 0   ' Turn an output switch on
Wait Ctr(3) >= 5
Off 0  ' When 5 input cycles are counted for Input 3 turn switch off (output 0 off)
```
CTReset Statement

Resets the counter value of the specified input counter and enables the input to be a counter input.

Syntax

```
CTReset(bitNumber)
```

Parameters

- `bitNumber` Number of the input bit set as a counter. This must be an integer expression representing a valid input bit. Only 16 counters can be active at the same time.

Description

CTReset works with the CTR function to allow inputs to be used as counters. CTReset sets the specified input bit as a counter and then starts the counter. If the specified input is already used as a counter, it is reset and started again.

Notes

Turning Off Power and Its Effect on Counters

Turning off main power releases all counters.

Using the Ctr Function

Use the Ctr Function to retrieve current Hardware Input counter values.

See Also

Ctr

CTReset Statement Example

The following example shows a sample of code which could be used to get a hardware input counter value.

```
CTReset 3  ' Reset Counter 3 to 0
On 0  ' Turn an output switch on
Wait Ctr(3) >= 5
Off 0  ' When 5 input cycles are counted for Input 3 turn switch off (output 0 off)
```
CtrlDev Function

Returns the current control device number.

Syntax
CtrlDev

Return Values
21 PC
22 Remote I/O
26 Remote Ethernet
29 Remote RS232C
20 TP3

See Also
CtrlInfo Function

CtrlDev Function Example

Print "The current control device is: ", CtrlDev
CtrlInfo Function

Returns controller information.

Syntax

CtrlInfo (index)

Parameters

index Integer expression that represents the index of the information to retrieve.

Description

The following table shows the information that is available from the CtrlInfo function:

<table>
<thead>
<tr>
<th>Index</th>
<th>Bit</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>N/A</td>
<td></td>
<td>Obtained for compatibility. Use index 9 to get the firmware version of the controller.</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>Controller status</td>
</tr>
<tr>
<td>0</td>
<td>&amp;H1</td>
<td>Ready state</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>&amp;H2</td>
<td>Start state</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>&amp;H4</td>
<td>Pause state</td>
<td></td>
</tr>
<tr>
<td>3-7</td>
<td></td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>&amp;H100</td>
<td>Estop state</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>&amp;H200</td>
<td>Safeguard open</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>&amp;H400</td>
<td>Error state</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>&amp;H800</td>
<td>Critical error state</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>&amp;H1000</td>
<td>Warning</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>&amp;H2000</td>
<td>WaitRecover state (Waiting for recover from safeguard open)</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>&amp;H4000</td>
<td>Recover state (Recovering from the safeguard open)</td>
<td></td>
</tr>
<tr>
<td>15-31</td>
<td></td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>&amp;H1</td>
<td>Enable switch is on</td>
</tr>
<tr>
<td>0-31</td>
<td></td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>&amp;H1</td>
<td>Teach mode circuit problem detected</td>
</tr>
<tr>
<td>1</td>
<td>&amp;H2</td>
<td>Safeguard circuit problem detected</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>&amp;H4</td>
<td>Estop circuit problem detected</td>
<td></td>
</tr>
<tr>
<td>3-31</td>
<td></td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>N/A</td>
<td></td>
<td>0 – Normal mode</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1 – Dry run mode</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>N/A</td>
<td>Control device:</td>
<td></td>
</tr>
<tr>
<td>21-22</td>
<td></td>
<td>RC+ – Remote</td>
<td></td>
</tr>
<tr>
<td>26-29</td>
<td></td>
<td>Remote Ethernet – Remote RS232C</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>N/A</td>
<td>Number of defined robots</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>N/A</td>
<td>Operation mode:</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>0 – Program mode</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1 – Auto mode</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>N/A</td>
<td>Undefined</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>N/A</td>
<td>Firmware version of the Controller</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Major No.*1000000 + Minor No.*10000 + Rev No.*100 + Build No. (Example) Version 1.6.2.4 is 1060204</td>
<td></td>
</tr>
</tbody>
</table>
### Return Values

Long value of the desired data

### See Also

RobotInfo, TaskInfo

### CtrlInfo Function Example

Print "The controller version: ", CtrlInfo(6)
CurDir$ Function

Returns a string representing the current directory.

Syntax
CurDir$

Return Values
A string that includes the current drive and path.

See Also
ChDir, CurDrive$, CurDisk$

CurDir$ Function Example

    Print "The current directory is: ", CurDir$
CurDisk$ Function

Returns a string representing the current disk.

Syntax
CurDisk$

Return Values
A string that contains the current disk letter.

See Also
ChDisk, CurDir$, CurDrive$

CurDisk$ Function Example

    Print "The current disk is: ", CurDisk$
CurDrive$ Function

Returns a string representing the current drive.

Syntax

CurDrive$

Return Values

A string that contains the current drive letter.

See Also

ChDrive, CurDir$, CurDisk$

CurDrive$ Function Example

Print "The current drive is: ", CurDrive$
CurPos Function

Returns the current target position of the specified robot.

Syntax
CurPos

Return Values
A robot point representing the current target position of the specified robot.

See Also
InPos, FindPos, RealPos

CurPos Function Example

Function main

    Xqt showPosition
    Do
         Jump P0
         Jump P1
     Loop
    Fend

    Function showPosition
    Do
        P99 = CurPos
        Print CX(P99), CY(P99)
    Loop
    Fend
Curve Statement

Defines the data and points required to move the arm along a curved path. Many data points can be defined in the path to improve precision of the path.

Syntax

Curve fileName, closure, mode, numAxes, pointList

Parameters

fileName A string expression for the name of the file in which the point data is stored. The specified fileName will have the extension .CVT appended to the end so no extension is to be specified by the user. When the Curve instruction is executed, file will be created.

You cannot specify a file path and fileName doesn’t have any effect from ChDisk. See ChDisk for the details.

closure Specifies whether or not the defined Curve is Closed or left Open at the end of the curved motion. This parameter must be set to one of two possible values, as shown below.

C - Closed Curve
O - Open Curve

When specifying the open curve, the Curve instruction creates the data to stop the arm at the last point of the specified point series. When specifying the closed curve, the Curve instruction creates the data required to continue motion through the final specified point and then stopping motion after returning the arm to the starting point of the specified point series for the Curve instruction.

mode Specifies whether or not the arm is automatically interpolated in the tangential direction of the U-Axis. It can also specify the ECP number in the upper four bits.

<table>
<thead>
<tr>
<th>Mode Setting</th>
<th>Tangential Correction</th>
<th>ECP Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hexadecimal</td>
<td>Decimal</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>No</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>32</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>160</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>176</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>192</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>208</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>224</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>240</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>No</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>34</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>162</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>178</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>194</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>210</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>226</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>242</td>
<td>15</td>
<td>15</td>
</tr>
</tbody>
</table>
When specifying tangential correction, Curve uses only the U-Axis coordinate of the starting point of the point series. Tangential correction continuously maintains tool alignment tangent to the curve in the XY plane. It is specified when installing tools such as cutters that require continuous tangential alignment. When specifying a closed curve (using the closure parameter) with Automatic Interpolation in the tangential direction of the U-Axis, the U-Axis rotates 360 degrees from the start point. Therefore, before executing the CVMove instruction, set the U-Axis movement range using the Range instruction so the 360 degree rotation of the U-Axis does not cause an error.

When using ECP, specify the ECP number in the upper four bits.

When generating a curve considering the additional axis position included in the point data, specify the ninth bit as 1. For example, when using no orientation offset or ECP and generating a curve considering the additional axis position, specify &H100.

When generating a curve for the additional axis, join the continuous point data of S axis and T axis separately from the robot coordinate system.

However if the additional axis is consisted of the PG axis, it doesn’t generate a curve with the continuous point but creates the data to move to the final point.

**numAxes**

Integer number 2, 3, 4, or 6 which specifies the number of axes controlled during the curve motion as follows:

2 - Generate a curve in the XY plane with no Z Axis movement or U Axis rotation. (except for 6-Axis robots (including N series))

3 - Generate a curve in the XYZ space with no U axis rotation. (except for 6-Axis robots (including N series))

4 - Generate a curve in the XYZ space with U-Axis rotation. (except for 6-Axis robots (including N series))

6 - Generate a curve in the XYZ space with U, V, and W axes rotation (6-Axis robots (including N series) only).

The axes not selected to be controlled during the Curve motion maintain their previous encoder pulse positions and do not move during Curve motion.

**pointList**

\{ point expression | P(start:finish) \} [...]

This parameter is actually a series of Point Numbers and optional output statements either separated by commas or an ascended range of points separated by a colon. Normally the series of points are separated by commas as shown below:

Curve "MyFile", O, 0, 4, P1, P2, P3, P4

Sometimes the user defines a series of points using an ascending range of points as shown below:

Curve "MyFile", O, 0, 4, P(1:4)

In the case shown above the user defined a curve using points P1, P2, P3, and P4. output command is optional and is used to control output operation during curve motion. The command can be On or Off for digital outputs or memory outputs. Entering an output command following any point number in the point series causes execution of the output command when the arm reaches the point just before the output command. A maximum of 16 output commands may be included in one Curve statement. In the example below, the "On 2" command is executed just as the arm reaches the point P2, then the arm continues to all points between and including P3 and P10.

Curve "MyFile", C, 0, 4, P1, P2, ON 2, P(3:10)
Curve Statement

**Description**
Curve creates data that moves the manipulator arm along the curve defined by the point series `pointList` and stores the data in a file on the controller. The `CVMove` instruction uses the data in the file created by `Curve` to move the manipulator in a continuous path type fashion.

The curve file is stored in the Compact Flash inside of the controller. Therefore, `Curve` starts writing into the Compact Flash. Frequent writing into the Compact Flash will shorten the Compact Flash lifetime. We recommend using `Curve` only for saving the point data.

`Curve` calculates independent X, Y, Z, U, V, W coordinate values for each point using a cubic spline function to create the trajectory. Therefore, if points are far apart from each other or the orientation of the robot is changed suddenly from point to point, the desired trajectory may not be realized.

It is not necessary to specify speeds or accelerations prior to executing the `Curve` instruction. Arm speed and acceleration parameters can be changed any time prior to executing `CVMove` by using the `SpeedS` or `AccelS` instructions.

Points defined in a local coordinate system may be used in the series to locate the curve at the desired position. By defining all of the specified points in the point series for the `Curve` instruction as points with local attributes, the points may be changed as points on the local coordinate system by the `Local` instruction following the `Curve` instruction.

**Notes**

**Use tangential correction when possible**
It is recommended that you use tangential correction whenever possible, especially when using `CVMove` in a continuous loop through the same points. If you do not use tangential correction, the robot may not follow the correct path at higher speeds.

**Open Curve Min and Max Number of Points Allowed**
Open Curves may be specified by using from 3 to 200 points.

**Closed Curve Min and Max Number of Points Allowed**
Closed Curves may be specified by using from 3 to 50 points.

**Potential Error**

**Attempt to Move Arm Outside Work Envelope**
The `Curve` instruction cannot check the movement range for the defined curve path. This means that a user defined path may cause the robot arm to move outside the normal work envelope. In this case an “out of range” error will occur.

**See Also**
`AccelS` Function, `Arc`, `CVMove`, `ECP`, `Move`, `SpeedS`

**Curve Statement Example**
The following example designates the free curve data file name as `MYCURVE.CVT`, creates a curve tracing P1-P7, switches ON output port 2 at P2, and decelerates the arm at P7.

```
Set up curve
> curve "mycurve", O, 0, 4, P1, P2, On 2, P(3:7)

Move the arm to P1 in a straight line
> jump P1

Move the arm according to the curve definition called “mycurve”
> cvmove "mycurve"
```
CVMove Statement

Performs the continuous spline path motion defined by the Curve instruction.

**Syntax**

```
CVMove fileName [CP] [searchExpr] [SYNC]
```

**Parameters**

- `fileName` String expression for the file name. This file must be previously created by the Curve instruction and stored on a PC hard disk.
  
  You cannot specify a file path and fileName doesn’t have any effect from ChDisk. See ChDisk for the details.

- `CP` Optional. Specifies continuous path motion after the last point.

- `searchExpr` Optional. A Till or Find expression.
  
  - **Till**
  
  ```
  Till Sw(expr) = \{On | Off\}
  ```
  
  - **Find**
  
  ```
  Find Sw(expr) = \{On | Off\}
  ```

- `SYNC` Reserves a motion command. A robot will not move until the SyncRobots gives instructions.

**Description**

CVMove performs the continuous spline path motion defined by the data in the file `fileName`, which is located in the controller memory. The file must be previously created with the Curve command. Multiple files may exist at the same time on the system. If the file name does not have an extension, .CVT is added automatically.

The user can change the speed and acceleration for the continuous path motion for CVMove by using the SpeedS and AccelS instructions.

When the Curve instruction has been previously executed using points with Local definitions, you can change the operating position by using the Local instruction.

When executing CVMove, be careful that the robot doesn’t collide with peripheral equipment. When you attempt to change the hand orientation of the 6-axis robot (including N series) between adjacent points suddenly, due to the nature of cubic spline function, the 6-axis robot may start changing its orientation from the previous and following points and move in an unexpected trajectory. Verify the trajectory thoroughly prior to a CVMove execution and be careful that the robot doesn’t collide with peripheral equipment.

Specify points closely each other and at equal interval. Do not change the hand orientation between adjacent points suddenly.

The CP parameter causes acceleration of the next motion command to start when the deceleration starts for the current motion command. In this case the robot will not stop at the destination coordinate and will continue to move to the next point.

**See Also**

AccelS Function, Arc, Curve, Move, SpeedS, Till, TillOn

**CVMove Statement Example**

The following example designates the free curve data file name as MYCURVE.CVT, creates a curve tracing P1-P7, switches ON output port 2 at P2, and decelerates the arm at P7.

```
Set up curve
> curve "mycurve", 0, 0, 4, P1, P2, On 2, P(3:7)

Move the arm to P1 in a straight line
> jump P1

Move the arm according to the curve definition called mycurve
> cvmove "mycurve"
```
Sets the coordinate value of a point data.
CV, CW are for only 6-axis robots (including N series).
CR is only for Joint type robots.
CS, CT are only for robots with additional axes.

Syntax
```
CX(point) = value
CY(point) = value
CZ(point) = value
CU(point) = value
CV(point) = value
CW(point) = value
CR(point) = value
CS(point) = value
CT(point) = value
```

Parameters
- `point`  
  P_number or P(expr) or point label.
- `value`  
  Real expression representing the new coordinate value in millimeters.

See Also
- CX, CY, CZ, CU, CV, CW, CR, CS, CT Functions

**CX, CY, CZ, CU, CV, CW, CR, CS, CT Statements Example**

```
CX(pick) = 25.34
```
CX, CY, CZ, CU, CV, CW, CR, CS, CT Functions

Retrieves a coordinate value from a point
CV, CW functions are only for 6-axis robots (including N series).
CS, CT are only for robots with additional axes.

Syntax

<table>
<thead>
<tr>
<th>Function</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>CX</td>
<td>CX (point)</td>
</tr>
<tr>
<td>CY</td>
<td>CY (point)</td>
</tr>
<tr>
<td>CZ</td>
<td>CZ (point)</td>
</tr>
<tr>
<td>CU</td>
<td>CU (point)</td>
</tr>
<tr>
<td>CV</td>
<td>CV (point)</td>
</tr>
<tr>
<td>CW</td>
<td>CW (point)</td>
</tr>
<tr>
<td>CR</td>
<td>CR (point)</td>
</tr>
<tr>
<td>CS</td>
<td>CS (point)</td>
</tr>
<tr>
<td>CT</td>
<td>CT (point)</td>
</tr>
</tbody>
</table>

Parameters

point  
Point expression.

Return Values

Returns the specified coordinate value. The return values for CX, CY, CZ are real numbers in millimeters.
The return values for CU, CV, CW are real numbers in degrees.
Return values of CS, CT functions: Real values in mm or deg. It depends on the additional axis setting.

Description

Used to retrieve an individual coordinate value from a point.

To obtain the coordinate from the current robot position, use Here for the point parameter.

See Also

CX, CY, CZ, CU, CV, CW, CR, CS, CT Statements

CX, CY, CZ, CU, CV, CW, CR, CS, CT Functions Example

The following example extracts the X axis coordinate value from point “pick” and puts the coordinate value in the variable x.

```plaintext
Function cxtest
  Real x
  x = CX(pick)
  Print "The X Axis Coordinate of point 'pick' is", x
Fend
```
Date Statement

Displays the date.

Syntax
Date

Return Values
The current date is displayed.

See Also
Time, Date$

Date Statement Example
Example from the command window.

    > Date
    2009/08/01
Date$ Function

Returns the system date.

Syntax

Date$

Return Values

A string containing the date in the format yyyy/mm/dd.

See Also

Date, Time, Time$

Date$ Function Example

Print "Today's date: ", Date$
Declare Statement

Declares an external function in a dynamic link library (DLL).

Syntax

\[
\text{Declare } \text{funcName}, \text{“dllFile”}, \text{“alias”}, \text{argList}] \text{ As type}
\]

Parameters

- **funcName**: The name of the function as it will be called from your program.
- **dllFile**: The path and name of the library file. This must be a literal string (characters delimited by quotation marks). You may also use a macro defined by `#define`. If there is no path specified, then RC+ will look for the file in the current project directory. If not found, then it is assumed that the file is in the Windows system32 directory. The file extension can be omitted, but is always assumed to be .DLL.
- **alias**: Optional. The actual name of the function in the DLL or the function index. The name is case sensitive. The alias must be a literal string (characters delimited by quotation marks). If you use an index, you must use a `#` character before the index. If omitted, a function name specified by `funcName` can be used as a name of function in DLL.
- **arglist**: Optional. List of the DLL arguments. See syntax below.

\[
\text{[ ByRef | ByVal }] \text{ varName } [()] \text{ As type}
\]

- **ByRef**: Optional. Specify ByRef when you refer to the variable to be seen by the calling function. In this case, the argument change in a function can be reflected to the variable of the calling side. You can change the values received as a reference.
- **ByVal**: Optional. Specify ByVal when you do not want any changes in the value of the variable to be seen by the calling function. This is the default.
- **varName**: Required. Name of the variable representing the argument; follows standard variable naming conventions. If you use an array variable as argument, you must specify ByRef.
- **type**: Required. You must declare the type of argument.

Description

Use Declare to call DLL functions from the current program. Declare must be used outside of functions.

The Declare statement checks that the DLL file and function exist at compile time.

Passing Numeric Variables ByVal

\[
\text{SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (a As Long) As Long}
\]

\[
\text{VC++ } \text{long _stdcall MyDllFunc(long a);}\]

Passing String Variables ByVal

\[
\text{SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (a$ As String) As Long}
\]

\[
\text{VC++ } \text{long _stdcall MyDllFunc(char *a);}\]
Passing Numeric Variables ByRef

SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (ByRef a As Long) As Long
VC++ long _stdcall MyDllFunc(long *a);

Passing String Variables ByRef

SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (ByRef a$ As String) As Long
VC++ long _stdcall MyDllFunc(char *a);

When you pass a string using ByRef, you can change the string in the DLL. Maximum string length is 255 characters. You must ensure that you do not exceed the maximum length.

Passing Numeric Arrays ByRef

SPEL: Declare MyDLLFunc, "mystuff.dll", "MyDLLFunc", (ByRef a() As Long) As Long
VC++ long _stdcall MyDllFunc(long *a);

Returning Values from DLL Function

The DLL function can return a value for any data type, including String. However, for a string, you must return a pointer to a string allocated in the DLL function. And the function name must end in a dollar sign, as with all SPEL+ string variables and functions. Note that the alias doesn't have a dollar sign suffix.

For example:

Declare ReturnLong, "mystuff.dll", "ReturnLong", As Long
Declare ReturnString$, "mystuff.dll", "ReturnString", As String

Function main

    Print "ReturnLong = ", ReturnLong
    Print "ReturnString$ = ", ReturnString$
    Fend

See Also

Function...Fend

Declare Statement Example

' Declare a DLL function. Since there is no path specified, the file can be in the current project directory or in the Windows system32 directory
Declare MyDLLTest, "mystuff.dll", "MyDLLTest" As Long

Function main

    Print MyDLLTest
    Fend

' Declare a DLL function with two integer arguments and use a #define to define the DLL file name
#define MYSTUFF "mystuff.dll"
Declare MyDLLCall, MYSTUFF, "MyTestFunc", (var1 As Integer, var2 As Integer) As Integer

' Declare a DLL function using a path and index.
Declare MyDLLTest, "c:\mydlls\mystuff.dll",="#1" As Long
DegToRad Function

Converts degrees to radians.

Syntax

```
DegToRad(degrees)
```

Parameters

- `degrees` Real expression representing the degrees to convert to radians.

Return Values

A double value containing the number of radians.

See Also

ATan, ATan2, RadToDeg Function

DegToRad Function Example

```
s = Cos(DegToRad(x))
```
Del Statement

Deletes one or more files.

**Syntax**

```
Del fileName
```

**Parameters**

*fileName*  
The path and name of the file(s) to delete. The filename should be specified with an extension. See ChDisk for the details.

**Description**

Deletes the specified file(s).

**Del Statement Example**

Example from the command window.

```plaintext
> Del TEST.PTS  ' Deletes the point file from the current directory.

> Del c:\TEST.PTS  ' NG
!! Error: 7213 The file specified by path does not exist.
> Del c:\TEST.PTS  ' OK
```
DeleteDB Statement

Deletes data from the table in the opened database.

Syntax

```
DeleteDB #databaseNum,tableNumber [, condition]
```

Parameters

- `databaseNum`: Specify the database number (integer from 501 to 508) specified in OpenDB.
- `tableNumber`: Specify the table name whose data will be deleted.
- `condition`: Specify the condition to delete the data. Compound condition can be specified by using AND and OR. If the condition is not specified, all data in the table will be deleted.

Description

Deletes the data matched to the delete condition from the specified table in the opened database. If the database is an Excel book, this command cannot be executed.

Note

- Connection of PC with installed RC+ is required.

See Also

OpenDB, CloseDB, SelectDB, UpdateDB
**DiffToolOrientation Function**

Returns the angle between the coordinate axes of Tool coordinate systems in order to show difference between Tool orientations of two specified points.

**Syntax**

```
DiffToolOrientation(pointData1, pointData2, axisNumber)
```

**Parameters**

- `pointData1`: Specify the first point data.
- `pointData2`: Specify the second point data.
- `axisNumber`: Specify the coordinate axis of Tool coordinate system.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>COORD_X_PLUS</td>
<td>1:</td>
</tr>
<tr>
<td>COORD_Y_PLUS</td>
<td>2:</td>
</tr>
<tr>
<td>COORD_Z_PLUS</td>
<td>3:</td>
</tr>
<tr>
<td>+X axis</td>
<td></td>
</tr>
<tr>
<td>+Y axis</td>
<td></td>
</tr>
<tr>
<td>+Z axis</td>
<td></td>
</tr>
</tbody>
</table>

**Return Values**

Angle (real value, from 0 to 180 degrees)

**Description**

Returns the angle (real value, from 0 to 180 degrees) between the specified coordinate axes of the Tool coordinate systems which indicates the difference between Tool orientations of two specified points. The results are not affected by the order of parameters, `pointData1` and `pointData2`. The results are also not affected by positional relation (coordinate values of X, Y, and Z) between the origin points of the two points.

**DiffToolOrientation Function Example**

```
' Displays the angle between Tool coordinate Z axes of Point 1 and 2.
Print DiffToolOrientation(P1, P2, COORD_Z_PLUS)
```
DispDev Statement

Sets the current display device.

Syntax

\[ \text{DispDev}(deviceID) \]

Parameters

\( deviceID \)  
The device ID for the desired display device.

- 21 RC+
- 24 TP (TP1 only)
- 20 TP3

The following parameters are also available.

- 21 DEVID_SELF
- 24 DEVID_TP
- 20 DEVID_TP3

See Also

DispDev Function

DispDev Statement Example

\[ \text{DispDev DEVID_TP} \]
DispDev Function

Returns the current display device.

Syntax

DispDev

Return Values

Integer value containing the deviceID.
21 RC+
24 TP (TP1 only)
20 TP3

See Also

DispDev Statement

DispDev Function Example

Print "The current display device is ", DispDev
Dist Function

Returns the distance between two robot points.

Syntax

\[
\text{Dist}(\text{point1}, \text{point2})
\]

Parameters

\text{point1, point2} \quad \text{Specifies two robot point expressions.}

Return Values

Returns the distance between both points (real value in mm).

Description

Even if you are using the additional axis, only the robot travel distance is returned. It doesn’t include the travel distance of additional axis while you use the additional axis as running axis. For the Joint type robot, the return value of this function means nothing.

See Also

CU, CV, CW, CX, CY, CZ

Dist Function Example

\[
\text{Real distance}
\]

\[
\text{distance} = \text{Dist}(P1, P2)
\]
Do...Loop Statement

Repeats a block of statements while a condition is True or until a condition becomes True.

Syntax

\[
\text{Do } \begin{cases} \text{ While } & \text{ condition } \\ \text{ Until } & \text{ condition } \end{cases} \\
\text{[statements]} \\
\text{[Exit Do]} \\
\text{[statements]} \\
\text{Loop}
\]

Or, you can use this syntax:

\[
\text{Do} \\
\text{[statements]} \\
\text{[Exit Do]} \\
\text{[statements]} \\
\text{Loop } \begin{cases} \text{ While } & \text{ condition } \\ \text{ Until } & \text{ condition } \end{cases}
\]

The Do Loop statement syntax has these parts:

<table>
<thead>
<tr>
<th>Part</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>condition</td>
<td>Optional. Numeric expression or string expression that is True or False. If condition is Null, condition is treated as False.</td>
</tr>
<tr>
<td>statements</td>
<td>One or more statements that are repeated while, or until, condition is True.</td>
</tr>
</tbody>
</table>

Description

Any number of Exit Do statements may be placed anywhere in the Do...Loop as an alternate way to exit a Do...Loop. Exit Do is often used after evaluating some condition, for example, If...Then, in which case the Exit Do statement transfers control to the statement immediately following the Loop.

When used within nested Do...Loop statements, Exit Do transfers control to the loop that is one nested level above the loop where Exit Do occurs.

Note

DO NOT use XQT command repeatedly in Loop statements.

Do not use XQT command repeatedly in Loop statements such as Do...Loop. The controller may freeze up. If you use Loop statements repeatedly, make sure to add Wait command (Wait 0.1).

See Also

For...Next, Select...Send

Do...Loop Statement Example

```
Do While Not Lof(1)
    Line Input #1, tLine$
    Print tLine$
Loop
```
Double Statement

Declares variables of type Double. (8 byte double precision number).

Syntax

```plaintext
Double varName [(subscripts)] [, varName [(subscripts)]]...
```

Parameters

- **varName**: Variable name which the user wants to declare as type Double.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows 
  (ubound1, [ubound2], [ubound3])
  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.
  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.
  When specifying the upper bound value, make sure the number of total elements is within the range shown below:
  - Local variable: 2,000
  - Global Preserve variable: 4,000
  - Global variable and module variable: 100,000

Description

Double is used to declare variables as type Double. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions. Valid number of digits for Double is 14.

See Also

- Boolean, Byte, Global, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UInt64, UShort

Double Statement Example

The following example shows a simple program which declares some variables using Double.

```plaintext
Function doubletest
    Double var1
    Double A(10) 'Single dimension array of double
    Double B(10, 10) 'Two dimension array of double
    Double C(5, 5, 5) 'Three dimension array of double
    Double arrayvar(10)
    Integer i
    Print "Please enter a Number:" 
    Input var1 
    Print "The variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter a Number:" 
        Input arrayvar(i) 
        Print "Value Entered was ", arrayvar(i)
    Next i 
Fend
```
ECP Statement

Selects or displays the current ECP (external control point).

Syntax

(1) ECP  ECPNumber
(2) ECP

Parameters

ECPNumber  Optional. Integer expression from 0 to 15 representing which of 16 ECP definitions to use with subsequent motion instructions. ECP 0 makes the ECP selection invalid.

Return Values

Displays current ECP when used without parameters.

Description

ECP selects the external control point specified by the ECPnumber (ECPNumber).

Note

This command will only work if the External Control Point option is active.

Power Off and Its Effect on the ECP Selection

Turning main power off clears the ECP selection.

See Also

ECPS

ECP Statement Example

>ecpset 1, 100, 200, 0, 0
>ecp 1
ECP Function

Returns the current ECP (external control point) number.

**Syntax**

```
ECP
```

**Return Values**

Integer containing the current ECP number.

**Note**

This command will only work if the External Control Point option is active.

**See Also**

ECP Statement

**ECP Function Example**

```
Integer savECP

savECP = ECP
ECP 2
Call Dispense
ECP savECP
```
ECPClr Statement

Clears (undefines) an external control point.

Syntax

```plaintext
ECPClr  ECPNumber
```

Parameters

- **ECPNumber**: Integer expression representing which of the 15 external control points to clear (undefine). (ECP0 is the default and cannot be cleared.)

Description

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Note

This command will only work if the External Control Point option is active.

See Also

Arm, ArmClr, ArmSet, ECPSel, Local, LocalClr, Tool, TLSet

ECPClr Statement Example

```plaintext
ECPClr 1
```
ECPDef Function

Returns ECP definition status.

Syntax

```
ECPDef (ECPNumber)
```

Parameters

- **ECPNumber**: Integer expression representing which ECP to return status for.

Return Values

- True if the specified ECP has been defined, otherwise False.

See Also

- Arm, ArmClr, ArmSet, ECPSet, Local, LocalClr, Tool, TLClr, TLSet

ECPDef Statement Example

```
Function DisplayECPDef(ecpNum As Integer)

    If ECPDef(ecpNum) = False Then
        Print "ECP ", ecpNum, "is not defined"
    Else
        Print "ECP ", ecpNum, ": ",
        Print ECPSet(ecpNum)
    EndIf
EndFunction
```
**ECPSet Statement**

Defines or displays an external control point.

**Syntax**

(1) \texttt{ECPSet \ ECPNum, ECPPoint}
(2) \texttt{ECPSet \ ECPNum}
(3) \texttt{ECPSet}

**Parameters**

- \textit{ECPNum} \hspace{1cm} Integer number from 1 to 15 representing which of 15 external control points to define.
- \textit{ECPPoint} \hspace{1cm} \textit{P}\textit{number} or \textit{P(expr)} or point label or point expression.

**Return Values**

- When parameters are omitted, displays the current ECPSet definitions.
- When only the ECP number is specified, displays the specified ECPSet definitions.

**Description**

Defines an external control point.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

**Note**

This command will only work if the External Control Point option is active.

**ECPSet Statement Example**

\begin{verbatim}
  ECPSet 1, P1
  ECPSet 2, 100, 200, 0, 0
\end{verbatim}
ECPSet Function

Returns a point containing the external control point definition for the specified ECP.

Syntax

\[
\text{ECPSet}(ECP\text{Number})
\]

Parameters

\textit{ECPNumber} \quad \text{Integer expression representing the number of the ECP to retrieve.}

Return Values

A point containing the ECP definition.

Note

This command will only work if the External Control Point option is active.

See Also

ECPSet Statement

ECPSet Function Example

\[
P1 \ = \ \text{ECPSet}(1)
\]
ElapsedTime Function

Returns the elapsed time since the takt time measurement timer starts in seconds.

Syntax

ElapsedTime

Return Values

An actual value representing an elapsed time of a takt time measurement timer. (Unit: second)
Valid range is from 0 to approx. 1.7E+31. Timer resolution is 0.001 seconds.

Description

Returns an elapsed time since the takt time measurement timer starts. Unlike the Tmr function, the ElapsedTime function does not count the time while the program is halted.

The takt time measurement timer can be reset by using ResetElapsedTime statement.

Real overhead

ResetElapsedTime
overHead = ElapsedTime

See Also

ResetElapsedTime, Tmr Function

ElapsedTime Function Example

ResetElapsedTime  ' Resets the takt time measurement timer
For i = 1 To 10  ' Executes 10 times
    GoSub Cycle
Next
Print ElapsedTime / 10  ' Measures a takt time and displays it
Elbow Statement

Sets the elbow orientation of a point.

Syntax
(1) `Elbow point [, value]`
(2) `Elbow`

Parameters
- `point`  
  `Pnumber` or `P(expr)` or point label.
- `value`  
  Integer expression.  
  `1` = Above (/A)  
  `2` = Below (/B)

Return Values
When both parameters are omitted, the elbow orientation is displayed for the current robot position. If `value` is omitted, the elbow orientation for the specified point is displayed.

See Also
Elbow Function, Hand, J4Flag, J6Flag, Wrist

Elbow Statement Example

```
Elbow P0, Below
Elbow pick, Above
Elbow P(myPoint), myElbow

P1 = 0.000, 490.000, 515.000, 90.000, -40.000, 180.000

Elbow P1, Above
Go P1

Elbow P1, Below
Go P1
```
Elbow Function

Returns the elbow orientation of a point.

Syntax

```
Elbow [(point)]
```

Parameters

- **point**: Optional. Point expression. If `point` is omitted, then the elbow orientation of the current robot position is returned.

Return Values

- 1: Above (/A)
- 2: Below (/B)

See Also

Elbow Statement, Hand, Wrist, J4Flag, J6Flag

Elbow Function Example

```
Print Elbow(pick)
Print Elbow(P1)
Print Elbow
Print Elbow(P1 + P2)
```
Eof Function

Returns end of file status.

**Syntax**

Eof (fileNumber)

**Parameters**

fileNumber  Integer number from 30 to 60 or expression representing the file number to check.

**Return Values**

True if file pointer is at end of file, otherwise False.

**Description**

Eof is functional only if the file is opened for reading mode. An error occurs if the file was opened with the AOpen or WOpen statements.

**See Also**

Lof

**Eof Function Example**

```
Integer fileNum
String data$

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
Do While Not Eof(fileNum)
    Line Input #fileNum, data$
    Print "data = ", data$
Loop
Close #fileNum
```
Era Function

Returns the joint number for which an error occurred.

Syntax

Era[\(\text{taskNum}\)]

Parameters

- **taskNum**: Integer expression representing a task number from 0 to 32. Task number omission or “0” specifies the current task.

Return Values

The joint number that caused the error in the range 0 to 9 as described below:

- 0 - The current error was not caused by a servo axis.
- 1 - The error was caused by joint number 1
- 2 - The error was caused by joint number 2
- 3 - The error was caused by joint number 3
- 4 - The error was caused by joint number 4
- 5 - The error was caused by joint number 5
- 6 - The error was caused by joint number 6
- 7 - The error was caused by joint number 7
- 8 - The error was caused by joint number 8 (additional S axis)
- 9 - The error was caused by joint number 9 (additional T axis)

Description

Era is used when an error occurs to determine if the error was caused by one of the robot joints and to return the number of the joint which caused the error. If the current error was not caused by any joint, Era returns “0”.

When the event “Error during Auto Mode” occurs, normal task and NoPause task in AUTO mode stop execution and end the task.

If the target task has already ended when using this function for NoEmgAbort task or background task, “Error 2261” is occurred. Use OnErr to acquire information before the task ends.

See Also

Erl, Err, ErrMsg$, Ert, OnErr, Trap

Era Function Example

```plaintext
Function main
    OnErr Goto eHandler
    Do
        Call PickPlace
        Loop
    Exit Function

    eHandler:
        Print "The Error code is ", Err
        Print "The Error Message is ", ErrMsg$(Err)
        errTask = Ert
        If errTask > 0 Then
            Print "Task number in which error occurred is ", errTask
            Print "The line where the error occurred is Line ", Erl(errTask)
            If Era(errTask) > 0 Then
                Print "Joint which caused the error is ", Era(errTask)
            EndIf
        EndIf
EndIf
Fend
```
EResume Statement

Resumes execution after an error-handling routine is finished.

Syntax

```
EResume [{ label | Next }]
```

Description

**EResume**

If the error occurred in the same procedure as the error handler, execution resumes with the statement that caused the error. If the error occurred in a called procedure, execution resumes at the Call statement in the procedure containing the error handler.

**EResume Next**

If the error occurred in the same procedure as the error handler, execution resumes with the statement immediately following the statement that caused the error. If the error occurred in a called procedure, execution resumes with the statement immediately following the Call statement that last in the procedure containing the error handler.

**EResume { label }**

If the error occurred in the same procedure as the error handler, execution resumes at the statement containing the label.

See Also

OnErr

**EResume Statement Example**

```vbnet
Function main
    Integer retry

    OnErr GoTo eHandler
    Do
        RunCycle
    Loop
    Exit Function

    eHandler:
    Select Err
    Case MyError
        retry = retry + 1
        If retry < 3 Then
            EResume ' try again
        Else
            Print "MyError has occurred ", retry, " times"
        EndIf
    EndIf
    Send
    Fend
```
**Erf$ Function**

Returns the name of the function in which the error occurred.

**Syntax**

\[
\text{Erf$[\{\text{taskNumber}\}]}
\]

**Parameters**

- \text{taskNumber} Integer expression representing a task number from 0 to 32. Task number omission or “0” specifies the current task.

**Return Values**

The name of the function where the last error occurred.

**Description**

Erf$ is used with OnErr. Erf$ returns the function name in which the error occurred. Using Erf$ combined with Err, Ert, Erl and Era the user can determine much more about the error which occurred. When the event “Error during Auto Mode” occurs, normal task and NoPause task in AUTO mode stop execution and end the task. If the target task has already ended when using this function for NoEngAbort task or background task, “Error 2261” is occurred. Use OnErr to acquire information before the task ends.

**See Also**

Era, Erl, Err, ErrMsg$, Ert, OnErr

**Erf$ Function Example**

The Following items are returned in the program example below.

- In which task the error occurred (Ert function)
- In which function the error occurred (Erf$ function)
- Where the error occurred (Erl function)
- On which joint the error occurred (Era function)

Function main

OnErr Goto eHandler

Do
    Call PickPlace
Loop
Exit Function

eHandler:
    Print "The Error code is ", Err
    Print "The Error Message is ", ErrMsg$(Err)
    errTask = Ert
    If errTask > 0 Then
        Print "Task number in which error occurred is ", errTask
        Print "Function at which error occurred is ", Erf$(errTask)
        Print "The line where the error occurred is Line ", Erl(errTask)
        If Era(errTask) > 0 Then
            Print "Joint which caused the error is ", Era(errTask)
        EndIf
    EndIf
EndIf
End
Erl Function

Returns the line number in which the error occurred.

Syntax

\[
\text{Erl}(\text{taskNumber})
\]

Parameters

- **taskNumber**: Integer expression representing a task number from 0 to 32. Task number omission or ”0” specifies the current task.

Return Values

The line number where the last error occurred.

Description

Erl is used with OnErr. Erl returns the line number in which the error occurred. Using Erl combined with Err, Ert and Era the user can determine much more about the error which occurred. When the event “Error during Auto Mode” occurs, normal task and NoPause task in AUTO mode stop execution and end the task. If the target task has already ended when using this function for NoEmgAbort task or background task, “Error 2261” is occurred. Use OnErr to acquire information before the task ends.

See Also

- Era, Erf$, Err, ErrMsg$, Ert, OnErr

Erl Function Example

The Following items are returned in the program example below.
- In which task the error occurred (Ert function)
- Where the error occurred (Erl function)
- What error occurred (Err function)
- On which joint the error occurred (Era function)

```plaintext
Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
    Loop
  Exit Function
EndFunction

eHandler:
  Print "The Error code is ", Err
  Print "The Error Message is ", ErrMsg$(Err)
  errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
EndIf
Fend
```
Err Function

Returns the most recent error status.

Syntax

```
Err [ (taskNumber) ]
```

Parameters

- **taskNumber**: Optional. Integer expression representing a task number from 0 to 32. “0” specifies the current task.

Return Values

Returns a numeric error code in integer form.

Description

Err allows the user to read the current error code. This along with the SPEL+ Error Handling capabilities allows the user to determine which error occurred and react accordingly. Err is used with OnErr.

To get the controller error, use SysErr function.

When the event “Error during Auto Mode” occurs, normal task and NoPause task in AUTO mode stop execution and end the task.

If the target task has already ended when using this function for NoEmgAbort task or background task, “Error 2261” is occurred. Use OnErr to acquire information before the task ends.

See Also

Era, Erf$, Erfl, ErrMsg$, EResume, Ert, OnErr, Return, SysErr

Err Function Example

The following example shows a simple utility program which checks whether points P0-P399 exist. If the point does not exist, then a message is printed on the screen to let the user know this point does not exist. The program uses the CX instruction to test each point for whether or not it has been defined. When a point is not defined control is transferred to the error handler and a message is printed on the screen to tell the user which point was undefined.

```
Function errtest
    Integer i, errnum
    Real x

    OnErr GoTo eHandle
    For i = 0 To 399
        x = CX(P(i))
    Next i
    Exit Function
    
    '*********************************************
    '* Error Handler                           *
    '*********************************************
eHandle:
    errnum = Err
    ' Check if using undefined point
    If errnum = 78 Then
        Print "Point number P", i, " is undefined!"
    Else
        Print "ERROR: Error number ", errnum, " Occurred."
    EndIf
    EResume Next
Fend
```
Erbb Function

Returns the robot number which the error occurred.

Syntax
Erbb

Return Values
Returns the robot number which the error occurred.

Description
Erbb finds and returns the robot number where the error occurred. If the robot is not the cause of the error, “0” will be returned.

See Also
Era, Erl, Err, ErrMsg$, OnErr, Trap

Erbb Function Example
The Following items are returned in the program example below.
- In which task the error occurred (Ert function)
- Where the error occurred (Erl function)
- What error occurred (Err function)
- On which joint the error occurred (Era function)
- On which robot the error occurred (Erbb function)

Function main
OnErr Goto eHandler
Do
 Call PickPlace
 Loop
 Exit Function
eHandler:
Print "The Error code is ", Err
Print "The Error Message is ", ErrMsg$(Err)
eErrTask = Ert
If errTask > 0 Then
 Print "Task number in which error occurred is ", errTask
 Print "The line where the error occurred is Line ", Erl(errTask)
 If Era(errTask) > 0 Then
 Print "Joint which caused the error is ", Era(errTask)
 EndIf
Print "Robot number in which error occurred is ", errb
EndIf
Fend
ErrMsg$ Function

Returns the error message which corresponds to the specified error number.

Syntax
ErrMsg$(errNumber, langID)

Parameters
errNumber Integer expression containing the error number to get the message for.
langID Optional. Integer expression containing the language ID based on the following values.
0 - English
1 - Japanese
2 - German
3 - French
4 – Simplified Chinese
5 – Traditional Chinese
If omitted, English is used.

Return Values
Returns the error message which is described in the Error Codes table.

See Also
Era, Erl, Err, Ert, OnErr, Trap

ErrMsg$ Function Example
The Following items are returned in the program example below.
In which task the error occurred (Ert function)
Where the error occurred (Erl function)
On which joint the error occurred (Era function)

Function main
OnErr Goto eHandler
Do
Call PickPlace
Loop
Exit Function
eHandler:
Print "The Error code is ", Err
Print "The Error Message is ", ErrMsg$(Err)
errTask = Ert
If errTask > 0 Then
Print "Task number in which error occurred is ", errTask
Print "The line where the error occurred is Line ", Erl(errTask)
If Era(errTask) > 0 Then
Print "Joint which caused the error is ", Era(errTask)
EndIf
EndIf
Fend
Error Statement

Generates a user error.

Syntax

(1) Error taskNumber, errorNumber
(2) Error errorNumber

Parameters

- taskNumber: Optional. Integer expression representing a task number from 0 to 32. “0” specifies the current task.
- errorNumber: Integer expression representing a valid error number. User error numbers range is from 8000 to 8999.

Description

Use the Error statement to generate system or user defined errors. You can define user error labels and descriptions by using the User Error Editor in the EPSON RC+ development environment.

See Also

Era, Erl, Err, OnErr

Error Statement Example

```plaintext
#define ER_VAC 8000

If Sw(vacuum) = Off Then
    Error ER_VAC
EndIf
```
ErrorOn Function

Returns the error status of the controller.

Syntax

    ErrorOn

Return Values

    True if the controller is in error status, otherwise False.

Description

    ErrorOn function is used only for NoEmgAbort task (special task using NoEmgAbort at Xqt) and background task.

See Also

    ErrorOn, SafetyOn, SysErr, Wait, Xqt

ErrorOn Function Example

    The following example shows a program that monitors the controller error and switches the I/O On/Off according to the error number when error occurs.

Notes

    Forced Flag

    This program example uses Forced flag for On/Off command. Be sure that the I/O outputs change during error, or at Emergency Stop or Safety Door Open when designing the system.

    After Error Occurrence

    As this program, finish the task promptly after completing the error handling.

Function main

    Xqt ErrorMonitor, NoEmgAbort :
    :
    Fend

Function ErrorMonitor

    Wait ErrorOn
    If 4000 < SysErr Then
        Print "Motion Error = ", SysErr
        Off 10, Forced
        On 12, Forced
    Else
        Print "Other Error = ", SysErr
        Off 11, Forced
        On 13, Forced
    EndIf
    Fend
Ert Function

Returns the task number in which an error occurred.

Syntax

Ert

Return Values

The task number in which the error occurred.

Description

Ert is used when an error occurs to determine in which task the error occurs.
Ert returns the number as follows:
No task with error (0), normal task (1 to 32), back ground task (65 to 80), TRAP task (257 to 267).

See Also

Era, Erl, Err, ErrMsg$, OnErr, Trap

Ert Function Example

The Following items are returned in the program example below.
  In which task the error occurred (Ert function)
  Where the error occurred (Erl function)
  On which joint the error occurred (Era function)

Function main
  OnErr Goto eHandler
  Do
    Call PickPlace
   Loop
  Exit Function
  eHandler:
    eHandler:
Print "The Error code is ", Err
Print "The Error Message is ", ErrMsg$(Err)
errTask = Ert
  If errTask > 0 Then
    Print "Task number in which error occurred is ", errTask
    Print "The line where the error occurred is Line ", Erl(errTask)
    If Era(errTask) > 0 Then
      Print "Joint which caused the error is ", Era(errTask)
    EndIf
  EndIf
EndIf
End
EStopOn Function

Returns the Emergency Stop status.

Syntax
EstopOn

Return Values
True if the status is Emergency Stop, otherwise False.

Description
EStopOn function is used only for NoEmgAbort task (special task using NoEmgAbort at Xqt).

See Also
ErrorOn, SafetyOn, Wait, Xqt

EstopOn Function Example
The following example shows a program that monitors the Emergency Stop and switches the I/O On/Off when Emergency Stop occurs.

Notes
Forced Flag
This program example uses Forced flag for On/Off command.
Be sure that the I/O outputs change during error, or at Emergency Stop or Safeguard Open when designing the system.

Error Handling
As this program, finish the task promptly after completing the error handling.

Outputs OFF during Emergency Stop
As this program example, when the task executes I/O On/Off after the Emergency Stop, uncheck the [Setup]-[System Configuration]-[Controller]-[Preferences]-[Outputs off during emergency stop] check box. If this check box is checked, the execution order of turn Off by the controller and turn On using the task are not guaranteed.

Function main

    Xqt EStopMonitor, NoEmgAbort
    :
    :  
    Fend

Function EStopMonitor
    Wait EStopOn
    Print "EStop !!!"
    Off 10, Forced
    On 12, Forced
    Fend
Eval Function

Executes a Command window statement from a program and returns the error status.

Syntax
Eval( command [, reply$ ] )

Parameters
command A string expression containing a command you want to execute.
reply$ Optional. A string variable that contains the reply from the command.
If the command is in the error status, it will return “!Error: error code”.
If the reply is over 255 characters, the extra characters will be truncated.

Return Values
The error code returned from executing the command.
Even if the command execution results in an error, the function itself will not be an error. Also, the system log doesn’t record it.
When the command is completed successfully, it returns “0”.

Description
You can execute any command (executable commands from Command window) from communication port such as TCP/IP by using Eval. It takes more time to execute this function than by using a normal statement.

Use the reply$ parameter to retrieve the reply from the command. For example, if the command was “Print Sw(1)”, then reply$ would be a “1” or “0”.

See Also
Error Codes

Eval Function Example
This example shows how to execute a command being read over RS-232. After the command is executed, the error code is returned to the host. For example, the host could send a command like “motor on”.

    Integer errCode
    String cmd$
    OpenCom #1
    Do
        Line Input #1, cmd$
        errCode = Eval(cmd$)
        Print #1, errCode
    Loop
Exit Statement

Exits a loop construct or function.

Syntax

Exit { Do | For | Function }

Description

The Exit statement syntax has these forms:

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exit Do</td>
<td>Provides a way to exit a Do...Loop statement. It can be used only inside a Do...Loop statement. Exit Do transfers control to the statement following the Loop statement. When used within nested Do...Loop statements, Exit Do transfers control to the loop that is one nested level above the loop where Exit Do occurs.</td>
</tr>
<tr>
<td>Exit For</td>
<td>Provides a way to exit a For loop. It can be used only in a For...Next loop. Exit For transfers control to the statement following the Next statement. When used within nested For loops, Exit For transfers control to the loop that is one nested level above the loop where Exit For occurs.</td>
</tr>
<tr>
<td>Exit Function</td>
<td>Immediately exits the Function procedure in which it appears. Execution continues with the statement following the statement that called the Function.</td>
</tr>
</tbody>
</table>

See Also

Do...Loop, For...Next, Function...Fend

Exit Statement Example

```plaintext
For i = 1 To 10
  If Sw(i) = On Then
    Exit For
  EndIf
  Jump P(i)
Next i
```
**ExportPoints Statement**

Exports a point file to the specified path.

**Syntax**

```plaintext
ExportPoints fileName, destination
```

**Parameters**

- `fileName`  String expression containing the specific file to be exported. The extension must be “.pts”. You cannot specify a file path and `fileName` doesn’t have any effect from ChDisk. See ChDisk for the details.

- `destination` Specify the path and file name to save the file. The extension must be “.pts”. See ChDisk for the details.

**Description**

ExportPoints copies a specified point file to a folder on the PC. If the file already exists in the folder, it will be overwritten.

**Potential Errors**

- **File Does Not Exist**
  If the specified path does not exist, an error will occur.

- **A Path Cannot be Specified**
  If `fileName` contains a path, an error will occur.

**See Also**

- Dir, LoadPoints, SavePoints, FileExists, FolderExists

**ExportPoints Statement Example**

```plaintext
Function main
    LoadPoints "robot1.pts"
    :
    SavePoints "robot1.pts"
    If FolderExists("c:\mypoints\") Then
        ExportPoints "robot1.pts", "c:\mypoints\modell.pts"
    EndIf
EndFend
```
FbusIO_GetBusStatus Function

Returns the status of the specified Fieldbus.

Syntax
FbusIO_GetBusStatus(busNumber)

Parameters
   busNumber   Integer expression representing the Fieldbus system number. This number must be 16. This is the ID for the bus connected to the Fieldbus master board on the PC side of the controller.

Return Values
   0 - OK
   1 - Disconnected
   2 - Power off

Description
FbusIO_GetBusStatus can be used to verify the general status of the Fieldbus.

Note
This command will only work if the Fieldbus Master option is active.

See Also
FbusIO_GetDeviceStatus, FbusIO_SendMsg

FbusIO_GetBusStatus Function Example

   Long  sts
   sts = FbusIO_GetBusStatus(16)
FbusIO_GetDeviceStatus Function

Returns the status of the specified Fieldbus device.

Syntax

FbusIO_GetDeviceStatus(busNumber, deviceID)

Parameters

busNumber       Integer expression representing the Fieldbus system number. This number must be 16. This is the ID for the bus connected to the Fieldbus master board on the PC side of the controller.

deviceID        Integer expression representing the Fieldbus ID of the device.

Return Values

0 - OK
1 - Disconnected
2 - Power off
3 - Synchronization error. Device is booting, or has incorrect baud rate.

Description

FbusIO_GetDeviceStatus can be used to verify the general status of a Fieldbus device.

Note

This command will only work if the Fieldbus Master option is active.

See Also

FbusIO_GetBusStatus, FbusIO_SendMsg

FbusIO_GetDeviceStatus Function Example

Long sts
sts = FbusIO_GetDeviceStatus(16, 10)
**FbusIO_SendMsg Statement**

Sends an explicit message to a Fieldbus device and returns the reply.

**Syntax**

\[
\text{FbusIO\_SendMsg}(\text{busNumber, deviceID, msgParam, sendData(), recvData()})
\]

**Parameters**

- **busNumber**: Integer expression representing the Fieldbus system number. This number must be 16. This is the ID for the bus connected to the Fieldbus master board on the PC side of the controller.
- **deviceID**: Integer expression representing the Fieldbus ID of the device.
- **msgParam**: Integer expression for the message parameter. Not used with DeviceNet.
- **sendData**: Array of type Byte containing data that is sent to the device. This array must be dimensioned to the number of bytes to send. If there are no bytes to send, specify 0.
- **recvData**: Array of type Byte that contains the data received from the device. This array will automatically be redimensioned to the number of bytes received.

**Description**

FBusIO_SendMsg is used to query one Fieldbus device. Refer to the device manufacturer for information on messaging support.

**Note**

This command will only work if the Fieldbus Master option is active.

**See Also**

FbusIO\_GetBusStatus, FbusIO\_GetDeviceStatus
FbusIO_SendMsg Statement Example

' Send explicit message to DeviceNet device
Byte sendData(5)
Byte recvData(0)
Integer i

sendData(0) = &H0E  ' Command
sendData(1) = 1    ' Class
sendData(3) = 1    ' Instance
sendData(5) = 7    ' Attribute
' msgParam is 0 for DeviceNet
FbusIO_SendMsg 16, 1, 0, sendData(), recvData()
' Display the reply
For i = 0 to UBound(recvData)
    Print recvData(i)
Next i

' Send message to Profibus device
Byte recvData(0)
Integer i

' msgParam is the service number
FbusIO_SendMsg 16, 1, 56, 0, recvData()
' Display the reply
For i = 0 to UBound(recvData)
    Print recvData(i)
Next i
**FileDateTime$ Function**

Returns the date and time of a file.

**Syntax**

```
FileDateTime$(filename)
```

**Parameters**

- `filename` A string expression containing the file name to check. The drive and path can also be included. If only file name is specified, the file in the current directory is displayed. See ChDisk for the details.

**Note**

A network path is available.

**Return Values**

Returns the date and time of the last update in the following format:

```
m/d/yyyy hh:mm:ss
```

**See Also**

FileExists, FileLen

**FileDateTime$ Function Example**

```plaintext
String myPath$
myPath$ = "c:\TEST\TEST.DAT"

If FileExists(myPath$) Then
    Print "Last access date and time: ", FileDateTime$(myPath$)
    Print "Size: ", FileLen(myPath$)
EndIf
```
FileExists Function

Checks if a file exists.

Syntax

FileExists ( filename )

Parameters

fileName A string expression containing the file name to check. The drive and path can also be included. If only the file name is specified, the file is checked in the current directory. See ChDisk for the details.

Note

A network path is available.

Return Values

True if the file exists, False if not.

See Also

FolderExists, FileLen, FileDateTime$

FileExists Function Example

String myPath$
myPath$ = "c:\TEST\TEST.DAT"

If FileExists (myPath$) Then
    Print "Last access date and time: ", FileDateTime$(myPath$)
    Print "Size: ", FileLen(myPath$)
EndIf
FileLen Function

Returns the length of a file.

Syntax

FileLen ( filename )

Parameters

filename

A string expression containing the file name to check. This includes both drive name and path name.
If only the file name is specified, the file is checked in the current directory.
See ChDisk for the details.

Note

A network path is available.

Return Values

Returns the number of bytes in the file.

See Also

FileDateTime$, FileExists

FileLen Function Example

String myPath$
myPath$ = "c:\TEST\TEST.DAT"

If FileExists(myPath$) Then
    Print "Last access date and time: ", FileDateTime$(myPath$)
    Print "Size: ", FileLen(myPath$)
EndIf
Find Statement

Specifies or displays the condition to store coordinates during motion.

Syntax

```
Find [condition]
```

Parameters

- **condition**: Input status specified as a trigger

```
[Event] comparative operator ( =, <>, >=, >, <, <=) [Integer expression]
```

The following functions and variables can be used in the Event:

**Functions**:
- Sw, In, InW, Oport, Out, OutW, MemSw, MemIn, MemInW, Ctr
- GetRobotInsideBox, GetRobotInsidePlane, AIO_In, AIO_InW, AIO_Out, AIO_OutW

**Variables**:
- Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort global
- preserve variable, Global variable, module variable

In addition, using the following operators you can specify multiple event conditions.

**Operator**:
- And, Or, Xor

**Example**:

```
Find Sw(5) = On
Find Sw(5) = On And Sw(6) = Off
```

Description

Find statement can be used by itself or as a modifier of a motion command.

The Find condition must include at least one of the functions above.

When variables are included in the Find condition, their values are computed when setting the Find condition. No use of variable is recommended. Otherwise, the condition may be an unintended condition. Multiple Find statements are permitted. The most recent Find condition remains current.

When parameters are omitted, the current Find definition is displayed.

Notes

**Find Setting at Main Power On**

At power on, the Find condition is:

```
Find Sw(0) = On  'Input bit 0 is on
```

**Use of PosFound Function to Verify Find**

Use PosFound function to verify if the Find condition has been satisfied after executing a motion command using Find modifier.

**Use Variables in Event Condition Expression**

- Available variables are Integer type (Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)
- Array variables are not available
- Local variables are not available
- If a variable value cannot satisfy the event condition for more than 0.01 seconds, the system cannot retrieve the change in variables.
- Up to 64 can wait for variables in one system (including the ones used in the event condition expressions such as Wait). If it is over 64, an error occurs during the project build.
- If you try to transfer a variable waiting for variables as a reference with Byref, an error occurs.
- When a variable is included in the right side member of the event condition expression, the value is calculated when starting the motion command. We recommend not using variables in an integer expression to avoid making unintended conditions.
See Also
FindPos, Go, Jump, PosFound

Find Statement Example

```plaintext
Find Sw(5) = On
Go P10 Find
If PosFound Then
  Go FindPos
Else
  Print "Cannot find the sensor signal."
EndIf
```
FindPos Function

Returns a robot point stored by Fine during a motion command.

Syntax

FindPos

Return Values

A robot point that was stored during a motion command using Find.

See Also

Find, Go, Jump, PosFound, CurPos, InPos

FindPos Function Example

Find Sw(5) = On
Go P10 Find
If PosFound Then
    Go FindPos
Else
    Print "Cannot find the sensor signal."
EndIf
Fine Statement

Specifies and displays the positioning accuracy for target points.

Syntax

(1) Ultra Sqi(E, axis1, axis2, axis3, axis4, axis5, axis6, axis7, axis8, axis9)
(2) Ultra Siq

Parameters

axis1 Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 1st joint.
axis2 Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 2nd joint.
axis3 Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 3rd joint.
axis4 Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 4th joint.
axis5 Optional. Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 5th joint. Only for the 6-axis robot (including N series).
axis6 Optional. Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 6th joint. Only for the 6-axis robot (including N series).
axis7 Optional. Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 7th joint. Only for the Joint type 7-axis robot.
axis8 Optional. Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 7th joint. Only for the additional S axis.
axis9 Optional. Integer expression ranging from (0 to 65535) which represents the allowable positioning error for the 7th joint. Only for the additional T axis.

* For C8, C12 series Manipulators, the allowable positioning error is from 0 to 131070.

Return Values

When used without parameters, Fine displays the current fine values for each axis.

Description

Fine specifies, for each joint, the allowable positioning error for detecting completion of any given move.

This positioning completion check begins after the CPU has completed sending the target position pulse to the servo system. Due to servo delay, the robot will not yet have reached the target position. This check continues to be executed every few milliseconds until each joint has arrived within the specified range configuration. Positioning is considered complete when all axes have arrived within the specified ranges. Once positioning is complete program control is passed to the next statement, however, servo system keeps the control of the robot target position.

When relatively large ranges are used with the Fine instruction, the positioning will be confirmed relatively early in the move, and executes the next statement.

The default Fine settings depend on the robot type. Refer to your robot manual for details.
Notes

Cycle Times and the Fine Instruction
The Fine value does not affect the acceleration or deceleration control of the manipulator arm. However, smaller Fine values can cause the system to run slower because it may take the servo system extra time (a few milliseconds) to get within the acceptable position range. Once the arm is located within the acceptable position range (defined by the Fine instruction), the CPU executes the next user instruction.

Initialization of Fine (by Motor On, SLock, SFree)
When any of the following commands is used, the Fine value will be initialized to the default: SLock, SFree, Motor instructions.
Make sure that you reset Fine values after one of the above commands is executed.

Potential Error
If Fine positioning is not completed within about 2 seconds, Error 4024 will occur. This error normally means the servo system balance needs to be adjusted. (Call your distributor for assistance)

See Also
Accel, AccelR, AccelS, Arc, Go, Jump, Move, Speed, SpeedR, SpeedS, Pulse, FineDist, FineStatus

Fine Statement Example
The examples below show the Fine statement used in a program function, and used from the monitor window.

```plaintext
Function finetest
  Fine 5, 5, 5, 5  ' reduces precision to +/- 5 Pulse
  Go P1
  Go P2
Fend

> Fine 10, 10, 10, 10
> Fine 10, 10, 10, 10
```
Fine Function

Returns Fine setting for a specified joint.

Syntax

`Fine(joint)`

Parameters

| joint | Integer expression representing the joint number for which to retrieve the Fine setting. The additional S axis is 8 and T axis is 9. |

Return Values

Real value.

See Also

`Accel, AccelS, Arc, Go, Jump, Move, Speed, SpeedS, Pulse`

Fine Function Example

This example uses the Fine function in a program:

```c
Function finetst
    Integer a
    a = Fine(1)
Fend
```
**FineDist Statement**

Specifies and displays the positioning error limits. The unit of the setting value is “mm”.

**Syntax**

(1) `FineDist value`

(2) `FineDist`

**Parameters**

`value` Positioning allowance ranges from 0.001[mm] to 10[mm].

**Return Values**

If the parameter is not specified, FineDist displays the current set value.

**Fine and FineDist**

The difference between Fine and FineDist is the unit of the positioning check. Fine statement sets the positioning check value in pulse, and the positioning check is performed on each axis. FineDist statement sets the positioning check value in mm, and the positioning check is performed in the coordinate system of Tool number 0. Fine and FineDist can be used at the same time. If Fine and FineDist are used in the program as shown below, the positioning check will be performed by FineDist. (If the order of Fine and FineDist is reversed, Fine will perform the positioning check.)

```
Function test
  Fine 5, 5, 5, 5
  FineDist 0.1

  Go P1
  Go P2
Fend
```
Note

Initialization of Fine (by Motor On, SLock, SFree, Till)
When any of the following commands is used, the FineDist value will be initialized to the default and the positioning check will be performed by Fine:
   SLock, SFree, Motor, Till
Make sure to reset the FineDist value after any of the above commands is executed.

Potential Error
If FineDist positioning is not completed within about 2 seconds, Error 4024 will occur. This error normally means the servo system balance needs to be adjusted.

See Also
Accel, AccelR, AccelS, Arc, Go, Jump, Move, Speed, SpeedR, SpeedS, Pulse, Fine, FineStatus

FineDist Statement Example
The example below show the FineDist statement used in a program function, and used from the monitor window.

```
Function fineDisttest
   Fine 0.1 'Set precision to +/- 0.1 mm
   Go P1
   Go P2
Fend

> FineDist 0.1
> FineDist 0.1
> FineDist 0.1
```
FineStatus Function

Returns whether Fine or FineDist is used by an integer.

**Syntax**

```
FineStatus
```

**Return Values**

Returns whether Fine is used or FineDist is used by an integer.

- 0 = Fine is used
- 1 = FineDist is used

**See Also**

Fine, FineDist

**FineStatus Function Example**

```
Print FineStatus
```
Fix Function

Returns the integer portion of a real number.

**Syntax**

```
Fix(number)
```

**Parameters**

- `number`  
  Real expression containing number to fix.

**Return Values**

An integer value containing the integer portion of the real number.

**See Also**

Int

**Fix Function Example**

```
>print Fix(1.123)
1
>
```
Flush Statement

Writes a file's buffer into the file.

Syntax

Flush #fileName

Parameters

#fileName Integer value from 30 to 63 or expression

Description

Writes a file's buffer into the specified file.
Flush cannot be used if the file was opened with ROpen.

Flush Statement Example

Integer fileNum, i

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Flush #fileNum
Close #fileNum
FmtStr Statement

Formats a numeric expression or date/time expression.

Syntax
FmtStr expFormat, strFormat, stringVar

Parameters
- **expFormat**: Numeric expression or date/time expression to be formatted.
- **strFormat**: Specify date/time expression in “yyyymm/dd”.
- **stringVar**: Format specification string.

Description
Returns the formatted string according to the **strFormat**.

### Numeric Format Specifiers

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>None</strong></td>
<td>Display the number with no formatting.</td>
</tr>
<tr>
<td><strong>(0)</strong></td>
<td>Digit placeholder. Display a digit or a zero. If the expression has a digit in the position where “0” appears in the format string, display it; otherwise, display a zero in that position. If the number has fewer digits than there are “0” (on either side of the decimal) in the format expression, display leading or trailing “0”. If the number has more digits to the right of the decimal separator than there are “0” to the right of the decimal separator in the format expression, round the number to as many decimal places as there are “0”. If the number has more digits to the left of the decimal separator than there are “0” to the left of the decimal separator in the format expression, display the extra digits without modification.</td>
</tr>
<tr>
<td><strong>(#)</strong></td>
<td>Digit placeholder. Display a digit or nothing. If the expression has a digit in the position where “#” appears in the format string, display it; otherwise, display nothing in that position. This symbol works like the 0 digit placeholder, except that leading and trailing “0” aren't displayed if the number has the same or fewer digits than there are “#” characters on either side of the decimal separator in the format expression.</td>
</tr>
<tr>
<td><strong>(.)</strong></td>
<td>Decimal placeholder. In some locales, a comma is used as the decimal separator. The decimal placeholder determines how many digits are displayed to the left and right of the decimal separator. If the format expression contains only number signs to the left of this symbol, numbers smaller than 1 begin with a decimal separator. To display a leading zero displayed with fractional numbers, use “0” as the first digit placeholder to the left of the decimal separator. The actual character used as a decimal placeholder in the formatted output depends on the Number Format recognized by your system.</td>
</tr>
</tbody>
</table>
| **(,)**   | Thousand separator. In some locales, a period is used as a thousand separator. The thousand separator separates thousands from hundreds within a number that has four or more places to the left of the decimal separator. Standard use of the thousand separator is specified if the format contains a thousand separator surrounded by digit placeholders (0 or #). Two adjacent thousand separators or a thousand separator immediately to the left of the decimal separator (whether or not a decimal is specified) means "scale the number by dividing it by 1000, rounding as needed." For example, you can use the format string "##0,0," to represent 100 million as “100”.
   Numbers smaller than 1 million are displayed as “0”. Two adjacent thousand separators in any position other than immediately to the left of the decimal separator are treated simply as specifying the use of a thousand separator. The actual character used as the thousand separator in the formatted output depends on the Number Format recognized by your system. |

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 259
### Date/Time Expression Specifiers

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(:)</td>
<td>Time separator. In some locals, other characters may be used. The time separator separates hours, minutes, and seconds when time values are formatted. The actual character used as the time separator in the formatted output depends on the Windows settings.</td>
</tr>
<tr>
<td>(/)</td>
<td>Date separator. In some locals, other characters may be used. The date separator separates day, month, and year when date values are formatted. The actual character used as the date separator in the formatted output depends on the Windows settings.</td>
</tr>
<tr>
<td>c</td>
<td>Display the date in “ddddd” and time in “ttttt”, in this order. If the date serial number does not have a fraction, it only displays the date. If the timing information does not have the integer, it only displays the timing information.</td>
</tr>
<tr>
<td>d</td>
<td>Display the date with the day in the lead without “0”. (1 to 31)</td>
</tr>
<tr>
<td>dd</td>
<td>Display the date with the day in the lead with “0”. (01 to 31)</td>
</tr>
<tr>
<td>ddd</td>
<td>Displays the abbreviation of the day of the week. (Sun to Sat)</td>
</tr>
<tr>
<td>dddd</td>
<td>Displays the unabbreviated day of the week. (Sunday to Saturday)</td>
</tr>
<tr>
<td>ddddd</td>
<td>Displays the day, month, and year in the format of the short data display settings of the Windows. Default setting of the short data display format is m/d/yy.</td>
</tr>
<tr>
<td>ddddddd</td>
<td>Displays the serial values of the date as day, month, and year in the long data display setting of the Windows. Default setting of the long data display is mmmm dd, yyyy.</td>
</tr>
<tr>
<td>w</td>
<td>Displays the day of the week with a number. (1: Sunday ~ 7: Saturday)</td>
</tr>
<tr>
<td>ww</td>
<td>Displays the number of weeks in a year with a number (1 to 54).</td>
</tr>
<tr>
<td>m</td>
<td>Display the month with the day in the lead without “0”. (1 to 12) Even if this character is placed right after “h” or “hh”, this does not display “minute”. To display “minute”, use “n” or “nn”.</td>
</tr>
<tr>
<td>mm</td>
<td>Display the month with the day in the lead with “0”.(01 to 12) Even if this character is placed right after “h” or “hh”, this does not display “minute”. To display “minute”, use “n” or “nn”.</td>
</tr>
<tr>
<td>mmm</td>
<td>Displays the abbreviated month name (Jan to Dec)</td>
</tr>
<tr>
<td>mmmm</td>
<td>Displays the unabbreviated month name (January to December) .</td>
</tr>
<tr>
<td>q</td>
<td>Displays the number of quarters in a year (1 to 4)</td>
</tr>
<tr>
<td>y</td>
<td>Displays the day of a year. (1 to 366)</td>
</tr>
<tr>
<td>yyyy</td>
<td>Displays the year in 2 digits.(00 to 99)</td>
</tr>
<tr>
<td>yyyy</td>
<td>Displays the year in 4 digits. (100 to 9999)</td>
</tr>
<tr>
<td>h</td>
<td>Displays the time in 24-hour clock without “0” at the beginning.(0 to 23)</td>
</tr>
<tr>
<td>hh</td>
<td>Displays the time in 24-hour clock with “0” at the beginning.(00 to 23)</td>
</tr>
<tr>
<td>n</td>
<td>Displays the minute without “0” at the beginning. (0 to 59)</td>
</tr>
<tr>
<td>nn</td>
<td>Displays the minute with “0” at the beginning.(00 to 59)</td>
</tr>
<tr>
<td>s</td>
<td>Displays the second without “0” at the beginning.(0 to 59)</td>
</tr>
<tr>
<td>ss</td>
<td>Displays the second with “0” at the beginning. (00 to 59)</td>
</tr>
<tr>
<td>t t t t t</td>
<td>Displays the time (hour, minute, second) with the time separator of Windows setting. If the “initial zero” option is used, the time before 10:00am/pm are displayed with “0” at the beginning. Default time format of the Windows is h:mm:ss.</td>
</tr>
<tr>
<td>AM/PM</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with AM/PM (uppercase).</td>
</tr>
<tr>
<td>am/pm</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with am/pm (lowercase).</td>
</tr>
<tr>
<td>A/P</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with A/P (uppercase).</td>
</tr>
<tr>
<td>a/p</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with a/p (lowercase).</td>
</tr>
</tbody>
</table>
AMPM Displays the time in 12-hour clock. For the morning, displays AM with a string and for the afternoon, displays the PM with a string each with the Windows format setting. Both uppercases and lowercases can be used for AM/PM if the specified string matches the Windows setting. Default Windows setting is AM/PM.

Note

Mixture of numeric format specifiers and time/date specifiers
An error occurs if both numeric format specifier and time/date specifier are specified.

See Also
Left$, Right$, Str$

FmtStr Statement Example

Function SaveData

    String d$, f$, t$

    ' Make file name in the format
    ' month, day, hour, minute
    d$ = Date$
    t$ = Time$
    d$ = d$ + " " + t$
    FmtStr d$, "mmddhhnn", f$
    f$ = f$ + ".dat"
    WOpen f$ as #30
    Print #30, "data"
    Close #30
Fend
FmtStr$ Function

Format a numeric expression.

Syntax

FmtStr$ (expFormat, strFormat)

Parameters

expFormat  Numeric expression or date/time expression to be formatted. Specify date/time expression in "yyyy/mm/dd".

strFormat  Format specification string.

Return Values

A string containing the formatted expression.

Description

Use FmtStr$ to format a numeric expression into a string.

Numeric Format Specifiers

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Display the number with no formatting.</td>
</tr>
<tr>
<td>(0)</td>
<td>Digit placeholder. Display a digit or a zero. If the expression has a digit in the position where &quot;0&quot; appears in the format string, display it; otherwise, display a zero in that position. If the number has fewer digits than there are “0” (on either side of the decimal) in the format expression, display leading or trailing “0”. If the number has more digits to the right of the decimal separator than there are “0” to the right of the decimal separator in the format expression, round the number to as many decimal places as there are “0”. If the number has more digits to the left of the decimal separator than there are “0” to the left of the decimal separator in the format expression, display the extra digits without modification.</td>
</tr>
<tr>
<td>(#)</td>
<td>Digit placeholder. Display a digit or nothing. If the expression has a digit in the position where “#” appears in the format string, display it; otherwise, display nothing in that position. This symbol works like the 0 digit placeholder, except that leading and trailing “0” aren't displayed if the number has the same or fewer digits than there are “#” characters on either side of the decimal separator in the format expression.</td>
</tr>
<tr>
<td>(.)</td>
<td>Decimal placeholder. In some locales, a comma is used as the decimal separator. The decimal placeholder determines how many digits are displayed to the left and right of the decimal separator. If the format expression contains only number signs to the left of this symbol, numbers smaller than 1 begin with a decimal separator. To display a leading zero displayed with fractional numbers, use “0” as the first digit placeholder to the left of the decimal separator. The actual character used as a decimal placeholder in the formatted output depends on the Number Format recognized by your system.</td>
</tr>
<tr>
<td>(,)</td>
<td>Thousand separator. In some locales, a period is used as a thousand separator. The thousand separator separates thousands from hundreds within a number that has four or more places to the left of the decimal separator. Standard use of the thousand separator is specified if the format contains a thousand separator surrounded by digit placeholders (0 or #). Two adjacent thousand separators or a thousand separator immediately to the left of the decimal separator (whether or not a decimal is specified) means &quot;scale the number by dividing it by 1000, rounding as needed.&quot; For example, you can use the format string &quot;##0,&quot;, to represent 100 million as “100&quot;. Numbers smaller than 1 million are displayed as “0&quot;. Two adjacent thousand separators in any position other than immediately to the left of the decimal separator are treated simply as specifying the use of a thousand separator. The actual character used as the thousand separator in the formatted output depends on the Number Format recognized by your system.</td>
</tr>
</tbody>
</table>
### FmtStr$ Function

#### Date/Time Expression Specifiers

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>:)</td>
<td>Time separator. In some locals, other characters may be used. The time separator separates hours, minutes, and seconds when time values are formatted. The actual character used as the time separator in the formatted output depends on the Windows settings.</td>
</tr>
<tr>
<td>(/)</td>
<td>Date separator. In some locals, other characters may be used. The date separator separates day, month, and year when date values are formatted. The actual character used as the date separator in the formatted output depends on the Windows settings.</td>
</tr>
<tr>
<td>c</td>
<td>Display the date in “ddddd” and time in “ttttt”, in this order. If the date serial number does not have a fraction, it only displays the date. If the timing information does not have the integer, it only displays the timing information.</td>
</tr>
<tr>
<td>d</td>
<td>Display the date with the day in the lead without “0”. (1 to 31)</td>
</tr>
<tr>
<td>dd</td>
<td>Display the date with the day in the lead with “0”. (01 to 31)</td>
</tr>
<tr>
<td>ddd</td>
<td>Displays the abbreviation of the day of the week. (Sun to Sat)</td>
</tr>
<tr>
<td>dddd</td>
<td>Displays the unabbreviated day of the week. (Sunday to Saturday)</td>
</tr>
<tr>
<td>ddddd</td>
<td>Displays the day, month, and year in the format of the short data display settings of the Windows. Default setting of the short data display format is m/d/yy.</td>
</tr>
<tr>
<td>ddddddd</td>
<td>Displays the serial values of the date as day, month, and year in the long data display setting of the Windows. Default setting of the long data display is mmmm dd, yyyy.</td>
</tr>
<tr>
<td>w</td>
<td>Displays the day of the week with a number. (1: Sunday ~ 7: Saturday)</td>
</tr>
<tr>
<td>ww</td>
<td>Displays the number of weeks in a year with a number (1 to 54).</td>
</tr>
<tr>
<td>m</td>
<td>Display the month with the day in the lead without “0”. (1 to 12) Even if this character is placed right after “h” or “hh”, this does not display “minute”. To display “minute”, use “n” or “nn”.</td>
</tr>
<tr>
<td>mm</td>
<td>Display the month with the day in the lead with “0”. (01 to 12) Even if this character is placed right after “h” or “hh”, this does not display “minute”. To display “minute”, use “n” or “nn”.</td>
</tr>
<tr>
<td>mmm</td>
<td>Displays the abbreviated month name (Jan to Dec)</td>
</tr>
<tr>
<td>mmmm</td>
<td>Displays the unabbreviated month name (January to December)</td>
</tr>
<tr>
<td>q</td>
<td>Displays the number of quarters in a year (1 to 4)</td>
</tr>
<tr>
<td>y</td>
<td>Displays the day of a year. (1 to 366)</td>
</tr>
<tr>
<td>yy</td>
<td>Displays the year in 2 digits. (00 to 99)</td>
</tr>
<tr>
<td>yyyy</td>
<td>Displays the year in 4 digits. (100 to 9999)</td>
</tr>
<tr>
<td>h</td>
<td>Displays the time in 24-hour clock without “0” at the beginning. (0 to 23)</td>
</tr>
<tr>
<td>hh</td>
<td>Displays the time in 24-hour clock with “0” at the beginning. (00 to 23)</td>
</tr>
<tr>
<td>n</td>
<td>Displays the minute without “0” at the beginning. (0 to 59)</td>
</tr>
<tr>
<td>nn</td>
<td>Displays the minute with “0” at the beginning. (00 to 59)</td>
</tr>
<tr>
<td>s</td>
<td>Displays the second without “0” at the beginning. (0 to 59)</td>
</tr>
<tr>
<td>ss</td>
<td>Displays the second with “0” at the beginning. (00 to 59)</td>
</tr>
<tr>
<td>t t t t t</td>
<td>Displays the time (hour, minute, second) with the time separator of Windows setting. If the “initial zero” option is used, the time before 10:00am/pm are displayed with “0” at the beginning. Default time format of the Windows is h:mm:ss.</td>
</tr>
<tr>
<td>AM/PM</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with AM/PM (uppercase).</td>
</tr>
<tr>
<td>am/pm</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with am/pm (lowercase).</td>
</tr>
<tr>
<td>A/P</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with A/P (uppercase).</td>
</tr>
<tr>
<td>a/p</td>
<td>Displays the time in 12-hour clock and displays morning and afternoon with a/p (lowercase).</td>
</tr>
</tbody>
</table>
FmtStr$ Function

**AMPM**
Displays the time in 12-hour clock. For the morning, displays AM with a string and for
the afternoon, displays the PM with a string each with the Windows format setting. Both
uppercases and lowercases can be used for AM/PM if the specified string matches the
Windows setting. Default Windows setting is AM/PM.

---

**Note**

**Mixture of numeric format specifiers and time/date specifiers**
An error occurs if both numeric format specifier and time/date specifier are specified.

**See Also**
Left$, Right$, Str$

---

**FmtStr$ Function Example**

```vbscript
Function SendDateCode
    String d$, f$
    f$ = FmtStr$(10, "000.00")
    OpenCom #1
    Print #1, f$
    CloseCom #1
End
```

---

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
FolderExists Function

Checks if a folder exists.

Syntax

FolderExists(pathName)

Parameters

pathName

A string expression containing the path of the folder to check. The drive can also be included. See ChDisk for the details.

Note

- This function is executable only with the PC disk.

Return Values

True if the folder exists, False if not.

See Also

FileExists, MkDir

FolderExists Function Example

If Not FolderExists("c:\TEST") Then
    MkDir "c:\TEST"
EndIf
For...Next Statement

The For...Next instructions are used together to create a loop where instructions located between For and Next are executed multiple times as specified by the user.

Syntax

```
For  var = initValue To finalValue  [Step increment ]
    statements
Next [var]
```

Parameters

- **var**
  - The counting variable used with the For...Next loop. This variable is normally defined as an integer but may also be defined as a Real variable.
- **initValue**
  - The initial value for the counter var.
- **finalValue**
  - The final value of the counter var. Once this value is met, the For...Next loop is complete and execution continues starting with the statement following the Next instruction.
- **increment**
  - An optional parameter which defines the counting increment for each time the Next statement is executed within the For...Next loop. This variable may be positive or negative. However, if the value is negative, the initial value of the variable must be larger than the final value of the variable. If the increment value is left out the system automatically increments by “1”.
- **statements**
  - Any valid SPEL+ statements can be inserted inside the For...Next loop.

Description

For...Next executes a set of statements within a loop a specified number of times. The beginning of the loop is the For statement. The end of the loop is the Next statement. A variable is used to count the number of times the statements inside the loop are executed.

The first numeric expression (initValue) is the initial value of the counter. This value may be positive or negative as long as the finalValue variable and Step increment correspond correctly.

The second numeric expression (finalValue) is the final value of the counter. This is the value which once reached causes the For...Next loop to terminate and control of the program is passed on to the next instruction following the Next instruction.

Program statements after the For statement are executed until a Next instruction is reached. The counter variable (var) is then incremented by the Step value defined by the increment parameter. If the Step option is not used, the counter is incremented by “1 (one)”.

The counter variable (var) is then compared with the final value. If the counter is less than or equal to the final value, the statements following the For instruction are executed again. If the counter variable is greater than the final value, execution branches outside of the For...Next loop and continues with the instruction immediately following the Next instruction.
Notes

Negative Step Values:
If the value of the Step increment (increment) is negative, the counter variable (var) is decremented (decreased) each time through the loop and the initial value must be greater than the final value for the loop to work.

Variable Following Next is Not Required:
The variable name following the Next instruction may be omitted. However, for programs that contain nested For...Next loops, it is recommended to include the variable name following the Next instruction to aid in quickly identifying loops.

When a variable exits the loop, the value is not a final value.

Function forsample
Integer i
For i = 0 To 3
Next
Print i  ' Displays 4
Fend

When you exit the loop by GoTo without using Exit For
Error 2020 will occur when you repeatedly execute the program which exits the loop by GoTo command, not Exit For command. Be sure to use Exit For command to exit the loop.

See Also
Do...Loop

For...Next Statement Example

Function fornext
Integer counter
For counter = 1 to 10
Go Pctr
Next counter

For counter = 10 to 1 Step -1
Go Pctr
Next counter
Fend
Force_Calibrate Statement

Sets zero offsets for all axes for the current force sensor.

Syntax

```
Force_Calibrate
```

Parameters

```
On | Off
```

Torque Control can be either On or Off.

Description

You should call Force_Calibrate for each sensor when your application starts. This will account for the weight of the components mounted on the sensor.

Note

This command will only work if the Force Sensing option is active.

See Also

```
Force_Sensor
```

Force_Calibrate Statement Example

```
Force_Calibrate
```
Force_ClearTrigger

Cleans all trigger conditions for the current force sensor.

**Syntax**

```
Force_ClearTrigger
```

**Description**

Use Force_ClearTrigger to clear all conditions for the current force sensor’s trigger.

**Note**

This command will only work if the Force Sensing option is active.

**See Also**

Force_Sensor, Force_SetTrigger

**Force_ClearTrigger Statement Example**

```
Force_ClearTrigger
```
**Force_GetForces Statement**

Returns the forces and torques for all force sensor axes in an array.

**Syntax**

```plaintext
Force_GetForces array()
```

**Parameters**

- `array()`: Real array with upper bound of 6.

**Return Values**

The array elements are filled in as follows:

<table>
<thead>
<tr>
<th>Index</th>
<th>Axis</th>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X Force</td>
<td>FORCE_XFORCE</td>
</tr>
<tr>
<td>2</td>
<td>Y Force</td>
<td>FORCE_YFORCE</td>
</tr>
<tr>
<td>3</td>
<td>Z Force</td>
<td>FORCE_ZFORCE</td>
</tr>
<tr>
<td>4</td>
<td>X Torque</td>
<td>FORCE_XTORQUE</td>
</tr>
<tr>
<td>5</td>
<td>Y Torque</td>
<td>FORCE_YTORQUE</td>
</tr>
<tr>
<td>6</td>
<td>Z Torque</td>
<td>FORCE_ZTORQUE</td>
</tr>
</tbody>
</table>

**Description**

Use `Force_GetForces` to read all force and torque values at once.

**Note**

This command will only work if the Force Sensing option is active.

**See Also**

- `Force_GetForce` Function

**Force_GetForces Statement Example**

```plaintext
Real fValues(6)
Force_GetForces fValues()
```
Force_GetForce Function

Returns the force for a specified axis.

Syntax

```
Force_GetForce (axis)
```

Parameters

- **axis**: Integer expression representing the axis.

<table>
<thead>
<tr>
<th>Axis</th>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>X Force</td>
<td>FORCE_XFORCE</td>
<td>1</td>
</tr>
<tr>
<td>Y Force</td>
<td>FORCE_YFORCE</td>
<td>2</td>
</tr>
<tr>
<td>Z Force</td>
<td>FORCE_ZFORCE</td>
<td>3</td>
</tr>
<tr>
<td>X Torque</td>
<td>FORCE_XTORQUE</td>
<td>4</td>
</tr>
<tr>
<td>Y Torque</td>
<td>FORCE_YTORQUE</td>
<td>5</td>
</tr>
<tr>
<td>Z Torque</td>
<td>FORCE_ZTORQUE</td>
<td>6</td>
</tr>
</tbody>
</table>

Return Values

Returns a real value.

Description

Use Force_GetForce to read the current force setting for one axis. The units are determined by the type of force sensor.

Note

This command will only work if the Force Sensing option is active.

See Also

- Force_GetForces

Force_GetForce Function Example

```
Print Force_GetForce(1)
```
Force_Sensor Statement

Sets the current force sensor for the current task.

**Syntax**

```
Force_Sensor sensorNumber
```

**Parameters**

- **sensorNumber** Integer expression representing the sensor number.

**Description**

When using multiple force sensors on the same system, you must set the current force sensor before using other force sensing commands.

If your system has only one sensor, then you don't need to use Force_Sensor because the default sensor number is 1.

**Note**

This command will only work if the Force Sensing option is active.

**See Also**

Force_Sensor Function

**Force_Sensor Statement Example**

```
Force_Sensor 1
```
Force_Sensor Function

Returns the current force sensor for the current task.

Syntax

\[ \text{Force\_Sensor} \]

Description

Force_Sensor returns the current sensor number for the current task. When a task starts, the sensor number is automatically set to 1.

Note

This command will only work if the Force Sensing option is active.

See Also

Force_Sensor

Force_Sensor Function Example

\[ \text{var} = \text{Force\_Sensor} \]
Force_SetTrigger Statement

Sets the force trigger for the Till command.

Syntax

**Force_SetTrigger** *axis*, *Threshold*, *CompareType*

Parameters

- **axis**: Integer expression containing the desired force sensor axis.
  - **Axis**: Constant Value
    - X Force: FORCE_XFORCE 1
    - Y Force: FORCE_YFORCE 2
    - Z Force: FORCE_ZFORCE 3
    - X Torque: FORCE_XTORQUE 4
    - Y Torque: FORCE_YTORQUE 5
    - Z Torque: FORCE_ZTORQUE 6

- **Threshold**: Real expression containing the desired threshold in units for the sensor being used.

- **CompareType**: Comparison Constant Value
  - Less than or equal: FORCE_LESS 0
  - Greater than or equal: FORCE_GREATER 1

Description

To stop motion with a force sensor, you must set the trigger for the sensor, then use Till Force in your motion statement.

You can set the trigger with multiple axes. Call Force_SetTrigger for each axis. To disable an axis, set the threshold at 0.

Note

This command will only work if the Force Sensing option is active.

See Also

Force_Calibrate

Force_SetTrigger Statement Example

```
' Set trigger to stop motion when force is less than −1 on Z axis.
Force_SetTrigger 3, -1, 0
SpeedS 3
AccelS 5000
Move Place Till Force
```
FreeFile Function

Returns / reserves a file number that is currently not being used.

Syntax
FreeFile

Return Values
Integer between 30 and 63.

See Also
AOpen, BOpen, ROpen, UOpen, WOpen, Close

FreeFile Function Example

```plaintext
Integer fileNum, i, j

fileNum = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
   Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
   Input #fileNum, j
   Print "data = ", j
Next i
Close #fileNum
```
A function is a group of program statements which includes a Function statement as the first statement and an Fend statement as the last statement.

**Syntax**

```plaintext
Function funcName [[argList]] [As type(function)]
statements
Fend
```

**Parameters**

- **funcName**
  The name which is given to the specific group of statements bound between the Function and Fend instructions. The function name must contain alphanumeric characters and may be up to 64 characters in length. Underscores are also allowed.

- **argList**
  Optional. List of variables representing arguments that are passed to the Function procedure when it is called. Multiple variables are separated by commas.

The arglist argument has the following syntax:

```plaintext
[ {ByRef | ByVal} ] varName [[ ()] ] As type(argument)
```

- **ByRef**
  Optional. Specify ByRef when you refer to the variable to be seen by the calling function. In this case, the argument change in a function can be reflected to the variable of the calling side.

- **ByVal**
  Optional. Specify ByVal when you do not want any changes in the value of the variable to be seen by the calling function. This is the default.

- **varName [[ ()] ]**
  Required. Name of the variable representing the argument; follows standard variable naming conventions. If you use an array variable as argument, you should specify ByRef and add empty parentheses “()” representing the array after the variable name.

- **As type (argument)**
  Required. You must declare the type of argument.

- **As type (function)**
  Use this parameter if you want to obtain return values. You must declare the type of return values.

**Return Values**

Value whose data type is specified with the As clause at the end of the function declaration (As type(function)).

**Description**

The Function statement indicates the beginning of a group of SPEL+ statements. To indicate where a function ends we use the Fend statement. All statements located between the Function and Fend statements are considered part of the function.

The Function...Fend combination of statements could be thought of as a container where all the statements located between the Function and Fend statements belong to that function. Multiple functions may exist in one program file.

If you want to use the return value, assign the value to the variable name which has the same name as the function and then terminate the function.

**See Also**

Call, Fend, Halt, Quit, Return, Xqt
Function...Fend Statement Example

<Example 1>
The following example shows 3 functions which are within a single file. The functions called task2 and task3 are executed as background tasks while the main task called main executes in the foreground.

```spel
Function main
  Xqt 2, task2   ' Execute task2 in background
  Xqt 3, task3   ' Execute task3 in background
  ' ...more statements here
Fend

Function task2
  Do
    On 1
    On 2
    Off 1
    Off 2
  Loop
Fend

Function task3
  Do
    On 10
    Wait 1
    Off 10
  Loop
Fend
```

<Example 2>
In the following example, the pressure control sequence for peripherals is supplied as an argument and the result sent to the external device is displayed as a return value.

```spel
Function main
  Integer iResult
  Real Sequence1(200)
  .
  iResult = PressureControl(ByRef Sequence1()) ' Argument is array
  Print "Result:", iResult
  .
Fend

Function PressureControl(ByRef Array1() As Real) As Integer
  (Control pressure for peripherals according to Array1)
  PressureControl = 3 ' Return value
  .
Fend
```
GetCurrentUser$ Function

Returns the current EPSON RC+ user.

Syntax
  GetCurrentUser$

Return Values
  String containing the current user logID.

Note
  This command will only work if the Security option is active.

See Also
  LogIn Statement

GetCurrentUser$ Function Example

  String currUser$
  currUser$ = GetCurrentUser$
GetRobotInsideBox Function

Returns a robot which is in the approach check area.

Syntax
GetRobotInsideBox(AreaNum)

Parameters
AreaNum Integer value (1 to 15) representing the approach check area you want to return the status for.

Return Values
Return the robot that is in the approach check area specified with AreaNum in bit.
Bit 0 : Robot 1 ……… Bit 15 : Robot 16
If the robot doesn’t configure the approach check area, bit is always 0.
For example, Robot 1, Robot 3 are in the approach check area, bit 0, bit 2 will be On and 5 will be returned.

See Also
Box, InsideBox

GetRobotInsideBox Function Example
The following program uses the GetRobotInsideBox function. Wait for the status that no robots are in the approach check area.

Function WaitNoBox
Wait GetRobotInsideBox(1) = 0

Wait for the status that Robot 2 is only one in the approach check area.

Function WaitInBoxRobot2
Wait GetRobotInsideBox(1) = &H2

The following program uses the GetRobotInsideBox function in the parallel processing of the motion command. When a robot is in the specific approach check area while it is running, it turns ON the I/O. One robot is connected to the controller in this case.

Function Main
Motor On
Power High
Speed 30; Accel 30, 30

Go P1 !D0; Wait GetRobotInsideBox(1) = 1; On 1!

Fend

Note
D0 must be described.
GetRobotInsidePlane Function

Returns a robot which is in the approach check plane.

Syntax
GetRobotInsidePlane (PlaneNum)

Parameters
PlaneNum  Integer value (1 to 15) representing the approach check plane you want to return the status for.

Return Values
Returns the number of the robot that is in the approach check plane specified with PlaneNum in bit.
Bit 0 : Robot 1 ……. Bit 15 : Robot 16
If the robot doesn’t configure the approach check plane, it always returns bit 0.
For example, Robot 1, Robot 3 are in the approach check plane, bit 0, bit 2 will be On and 5 will be returned.

See Also
InsidePlane, Plane

GetRobotInsidePlane Function Example
The following program uses the GetRobotInsidePlane function.
Wait for the status that no robots are in the approach check plane.

Function WaitNoPlane
  Wait GetRobotInsidePlane(1) = 0

Wait for the status Robot 2 is only one in the approach check plane.

Function WaitInPlaneRobot2
  Wait GetRobotInsidePlane(1) = &H2

The following program uses the GetRobotInsidePlane function in the parallel processing of the motion command. When a robot is in the specific approach check plane while it is running, it turns ON the I/O. One robot is connected to the controller in this case.

Function Main
  Motor On
  Power High
  Speed 30; Accel 30, 30
  Go P1 !D0; Wait GetRobotInsidePlane(1) = 1; On 1!
  Fend

Note
D0 must be described.
Global Statement

Declares variables with the global scope. Global variables can be accessed from anywhere.

Syntax

\[
\text{Global [ Preserve ] dataType varName \{[subscripts]\} \[, \ varName \{[subscripts]\}\} , ...]
\]

Parameters

- **Preserve**: If Preserve is specified, then the variable retains its values. The values are cleared by project changes. If Preserve is omitted, the variable doesn’t retain its values.
- **dataType**: Data type including Boolean, Byte, Double, Int32, Integer, Long, Real, Short, String, UByte, UInt32, or UShort.
- **varName**: Variable name. Names may be up to 32 characters in length.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows:
  \[(ubound1, [ubound2], [ubound3])\]
  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.
  The elements in each dimension of an array are numbered from 0 to the upper bound value.
  The total available number of array elements for global variables is 10000 for strings and 100000 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.
  To calculate the total elements used in an array, use the following formula.
  (If a dimension is not used, substitute 0 for the ubound value.)
  total elements = (ubound1 + 1) * (ubound2 + 1) * (ubound3 + 1)

Description

Global variables are variables which can be used in more than 1 file within the same project. They are cleared whenever a function is started from the Run window or Operator window unless they are declared with the Preserve option.

When declared in Preserve option, the variable retains the value at turning off the controller.

Global Preserve variables can be used with the RC+ Connectivity option.

It is recommended that global variable names begin with a "g_" prefix to make it easy to recognize globals in a program. For example:

\[
\text{Global Long g_PartsCount}
\]

See Also

Boolean, Byte, Double, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UInt64, UShort
Global Statement

Global Statement Example

The following example shows 2 separate program files. The first program file defines some global variables and initializes them. The second file then also uses these global variables.

FILE1 (MAIN.PRG)

```
Global Integer g_Status
Global Real g_MaxValue

Function Main
  g_Status = 10
  g_MaxValue = 1.1
  .
  Fend
```

FILE2 (TEST.PRG)

```
Function Test
  Print "status1 =", g_Status
  Print "MaxValue =", g_MaxValue
  .
  Fend
```
Go Statement

Moves the arm using point to point motion from the current position to the specified point or X, Y, Z, U, V, W position. The Go instruction can move any combination of 1-6 joints at the same time.

Syntax

Go destination [CP] [LJM [orientationFlag]] [PerformMode modeNumber] [searchExpr] ![...!] [SYNC]

Parameters

- **destination**: The target destination of the motion using a point expression.
- **CP**: Optional. Specifies continuous path motion.
- **LJM**: Optional. Convert the target destination using LJM function.
- **orientationFlag**: Optional. Specifies a parameter that selects an orientation flag for LJM function.
- **PerformMode modeNumber**: Optional. Specify the robot performance mode.
- **searchExpr**: Optional. A Till or Find expression.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode_Standard</td>
<td>1</td>
<td>Sets the Standard mode</td>
</tr>
<tr>
<td>Mode_High_Speed</td>
<td>2</td>
<td>Sets the High-speed mode</td>
</tr>
<tr>
<td>Mode_Low_Oscillation</td>
<td>3</td>
<td>Sets the Low-oscillation mode</td>
</tr>
</tbody>
</table>

- **Till | Find**
- **Till Sw(expr) = {On | Off}**
- **Find Sw(expr) = {On | Off}**

- **SYNC**: Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Go simultaneously moves all joints of the robot arm using point to point motion. The destination for the Go instruction can be defined in a variety of ways:

- Using a specific point to move to. For example: Go P1.
- Using an explicit coordinate position to move to. For example: Go XY(50, 400, 0, 0).
- Using a point with a coordinate offset. For example: Go P1 +X(50).
- Using a point but with a different coordinate value. For example: Go P1 :X(50).

The path is not predictable because each joint interpolates between the current point and the target point. Be careful of the interference with peripherals.

The Speed instruction determines the arm speed for motion initiated by the Go instruction. The Accel instruction defines the acceleration.

With CP parameter, the arm can accelerate for the next motion command while the arm starts decelerating to a stop. In this case, the arm is not positioned at the target point.

With LJM parameter, the arm moves to the point into where the target point is converted using LJM function, with the current point as reference point.

Go LJM (P1, Here, 1) can be Go P1 LJM 1.
At this point, the original point data P1 does not change.
Go Statement

LJM parameter is available for the 6-axis (including N series) and RS series robots.
When using orientationFlag with the default value, it can be omitted.

Go P1 LJM

Deceleration motion and acceleration motion of different modes can be combined when PerformMode is set
while the path motion is enabled. Some combinations are not available depending on operation modes. For
details, refer to PerformMode Statement.

Notes

Difference between Go and Move

The Move instruction and the Go instruction each cause the robot arm to move. However, the primary
difference between the 2 instructions is that the Go instruction causes point to point motion whereas the Move
instruction causes the arm to move in a straight line. The Go instruction is used when the user is primarily
concerned with the orientation of the arm when it arrives on point. The Move instruction is used when it is
important to control the path of the robot arm while it is moving.

Difference between Go and Jump

The Jump instruction and the Go instruction each cause the robot arm to move in a point to point type fashion.
However, the JUMP instruction has 1 additional feature. Jump causes the robot end effector to first move up
to the LimZ value, then in a horizontal direction until it is above the target point, and then finally down to
the target point. This allows Jump to be used to guarantee object avoidance and more importantly to improve
cycle times for pick and place motions.

Proper Speed and Acceleration Instructions with Go

The Speed and Accel instructions are used to specify the speed and acceleration of the manipulator during
motion caused by the Go instruction. Pay close attention to the fact that the Speed and Accel instructions
apply to point to point type motion (like that for the Go instruction) while linear and circular interpolation
motion uses the SpeedS and AccelS instructions.

Using Go with the Optional Till Modifier

The optional Till modifier allows the user to specify a condition to cause the robot to decelerate to a stop at
an intermediate position prior to completing the motion caused by the Go instruction. If the Till condition is
not satisfied, the robot travels to the target position. The Go with Till modifier can be used in 2 ways as
described below:

(1) Go with Till Modifier
Checks if the current Till condition becomes satisfied. If satisfied, this command completes by
decelerating and stopping the robot at an intermediate position prior to completing the motion
caused by the Go instruction.

(2) Go with Till Modifier, Sw(Input bit number) Modifier, and Input Condition
This version of the Go with Till modifier allows the user to specify the Till condition on the same
line with the Go instruction rather than using the current definition previously defined for Till. The
condition specified is simply a check against one of the inputs. This is accomplished through using
the Sw instruction. The user can check if the input is On or Off and cause the arm to stop based on
the condition specified. This feature works almost like an interrupt where the motion is interrupted
(stopped) once the Input condition is met. If the input condition is never met during the robot motion
then the arm successfully arrives on the point specified by destination.

Using Go with the Optional Find Modifier

The optional Find modifier allows the user to specify a condition to cause the robot to record a position
during the motion caused by the Go instruction. The Go with Find modifier can be used in 2 ways as
described below:

(1) Go with Find Modifier:
Checks if the current Find condition becomes satisfied. If satisfied, the current position is stored in
the special point FindPos.

(2) Go with Find Modifier, Sw(Input bit number) Modifier, and Input Condition:
This version of the Go with Find modifier allows the user to specify the Find condition on the same
line with the Go instruction rather than using the current definition previously defined for Find. The
condition specified is simply a check against one of the inputs. This is accomplished through using
the Sw instruction. The user can check if the input is On or Off and cause the current position
Go Statement

to be stored in the special point FindPos.

**Go Instruction Always Decelerates to a Stop**

The Go instruction always causes the arm to decelerate to a stop prior to reaching the final destination of the move.

---

**Potential Error**

**Attempt to Move Outside of Robots Work Envelope**

When using explicit coordinates with the Go instruction, you must make sure that the coordinates defined are within the robot's valid work envelope. Any attempt to move the robot outside of the valid work envelope will result in an error.

---

**See Also**

!...! Parallel Processing, Accel, Find, Jump, Move, Pass, P#= (Point Assignment), PerformMode, Pulse, Speed, Sw, Till

**Go Example**

The example shown below shows a simple point to point move between points P0 and P1. Later in the program the arm moves in a straight line toward point P2 until input #2 turns on. If input #2 turns on during the Go, then the arm decelerates to a stop prior to arriving on point P2 and the next program instruction is executed.

Function sample

```plaintext
Integer i

Home
Go P0
Go P1
For i = 1 to 10
   Go P(i)
Next i
Go P2 Till Sw(2) = On
If Sw(2) = On Then
   Print "Input #2 came on during the move and"
   Print "the robot stopped prior to arriving on"
   Print "point P2."
Else
   Print "The move to P2 completed successfully."
   Print "Input #2 never came on during the move."
EndIf
Fend
```

Some syntax examples from the command window are shown below:

```plaintext
>Go Here +X(50)       ' Move only in the X direction 50 mm from current position
>Go P1               ' Simple example to move to point P1
>Go P1 :U(30)         ' Move to P1 but use +30 as the position for the U joint to move to
>Go P1 /L             ' Move to P1 but make sure the arm ends up in lefty position
>Go XY(50, 450, 0, 30) ' Move to position X=50, Y=450, Z=0, U=30

<Another Coding Example>
Till Sw(1) = Off And Sw(2) = On   ' Specifies Till conditions for inputs 1 & 2
Go P1 Till
Go P2 Till Sw(2) = On            ' Stop if current Till condition defined on previous line is met
Go P3 Till
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 285
GoSub transfers program control to a subroutine. Once the subroutine is complete, program control returns back to the line following the GoSub instruction which initiated the subroutine.

**Syntax**

```plaintext
GoSub { label }

{ label;}
statements
Return
```

**Parameters**

*label*  
When the user specifies a label, the program execution will jump to the line on which this label resides. The label can be up to 32 characters in length. However, the first character must be an alphabet character (not numeric).

**Description**

The GoSub instruction causes program control to branch to the user specified statement label. The program then executes the statement on that line and continues execution through subsequent line numbers until a Return instruction is encountered. The Return instruction then causes program control to transfer back to the line which immediately follows the line which initiated the GoSub in the first place. (i.e. the GoSub instruction causes the execution of a subroutine and then execution returns to the statement following the GoSub instruction.) Be sure to always end each subroutine with Return. Doing so directs program execution to return to the line following the GoSub instruction.

**Potential Errors**

**Branching to Non-Existent Statement**

If the GoSub instruction attempts to branch control to a non-existent label then an Error 3108 will be issued.

**Return Found Without GoSub**

A Return instruction is used to "return" from a subroutine back to the original program which issued the GoSub instruction. If a Return instruction is encountered without a GoSub having first been issued then an Error 2383 will occur. A standalone Return instruction has no meaning because the system doesn't know where to Return to.

**See Also**

GoTo, OnErr, Return
GoSub Statement Example

The following example shows a simple function which uses a GoSub instruction to branch to a label and execute some I/O instructions then return.

```
Function main
  Integer var1, var2
  GoSub checkio  'GoSub using Label
    On 1
    On 2
    Exit Function
checkio:           'Subroutine starts here
  var1 = In(0)
  var2 = In(1)
  If var1 = 1 And var2 = 1 Then
    On 1
  Else
    Off 1
  EndIf
  Return         'Subroutine ends here
Fend
```
GoTo Statement

The GoTo instruction causes program control to branch unconditionally to a designated statement label.

Syntax

```
GoTo {label}
```

Parameters

`label`  
Program execution will jump to the line on which the label resides. The label can be up to 32 characters. However, the first character must be an alphabetic character (not numeric).

Description

The GoTo instruction causes program control to branch to the user specified label. The program then executes the statement on that line and continues execution from that line on. GoTo is most commonly used for jumping to an exit label because of an error.

Note

Using Too Many GoTo's

Please be careful with the GoTo instruction since using too many GoTo's in a program can make the program difficult to understand. The general rule is to try to use as few GoTo instructions as possible. Some GoTo's are almost always necessary. However, jumping all over the source code through using too many GoTo statements is an easy way to cause problems.

See Also

`GoSub`, `OnErr`

GoTo Statement Example

The following example shows a simple function which uses a GoTo instruction to branch to a line label.

```pascal
Function main
  If Sw(1) = Off Then
    GoTo mainAbort
  EndIf
  Print "Input 1 was On, continuing cycle"
  .
  Exit Function

mainAbort:
  Print "Input 1 was OFF, cycle aborted!"
Fend
```
Halt Statement

Temporarily suspends execution of a specified task.

Syntax

Halt taskIdentifier

Parameters

- taskIdentifier: Task name or integer expression representing the task number. A task name is the function name used in an Xqt statement or a function started from the Run window or Operator window. If an integer expression is used, the range is from 1 to 16 for normal tasks and from 257 to 261 for trap tasks.

Description

Halt temporarily suspends the task being executed as specified by the task name or number.

To continue the task where it was left off, use Resume. To stop execution of the task completely, use Quit. To display the task status, click the Task Manager Icon on the EPSON RC+ Toolbar to run the Task manager.

Halt also stops the task when the specified task is NoPause task, NoEngAbort task (special task using NoPause or NoEngAbort at Xqt), trap tasks, or the background tasks. However, stopping these tasks needs enough consideration. Normally, Halt is not recommended for the special task.

See Also

Quit, Resume, Xqt

Halt Statement Example

The example below shows a function named “flicker” that is started by Xqt, then is temporarily stopped by Halt and continued again by Resume.

```c
Function main
    Xqt flicker        ' Execute flicker function
    Do
        Wait 3           ' Execute task flicker for 3 seconds
        Halt flicker
        Wait 3           ' Halt task flicker for 3 seconds
        Resume flicker
    Loop
Fend

Function flicker
    Do
        On 1
        Wait 0.2
        Off 1
        Wait 0.2
    Loop
Fend
```
Hand Statement

Sets the hand orientation of a point.

Syntax
(1) Hand point [Lefty | Righty]
(2) Hand

Parameters
point Pnumber or P(expr) or point label.
Lefty | Righty Hand orientation.

Return Values
When both parameters are omitted, the hand orientation is displayed for the current robot position. If Lefty | Righty is omitted, the hand orientation for the specified point is displayed.

See Also
Elbow, Hand Function, J4Flag, J6Flag, Wrist, J1Flag, J2Flag

Hand Statement Example

Hand P0, Lefty
Hand pick, Righty
Hand P(myPoint), myHand

Pl = -364.474, 120.952, 469.384, 72.414, 1.125, -79.991

Hand Pl, Righty
Go Pl

Hand Pl, Lefty
Go Pl
Hand Function

Returns the hand orientation of a point.

Syntax

```plaintext
Hand ([point])
```

Parameters

- `point` Optional. Point expression. If `point` is omitted, then the hand orientation of the current robot position is returned.

Return Values

1. Righty (/R)
2. Lefty (/L)

See Also

Elbow, Wrist, J4Flag, J6Flag, J1Flag, J2Flag

Hand Function Example

```plaintext
Print Hand(pick)
Print Hand(P1)
Print Hand
Print Hand(P1 + P2)
```
HealthCalcPeriod

Set and display a period calculating “remaining months” of parts consumption commands information.

Syntax
(1) HealthCalcPeriod Period
(2) HealthCalcPeriod

Parameters
Period Specify a period to calculate in integer (1~7). (Unit: day) Default value is “7”.

Return Values
Display the current setting value of HealthCalcPeriod if omitting the parameter.

Description
Remaining months of parts consumption commands information is automatically calculated based on the past operating condition. HealthCalcPeriod command sets and displays an operating period for this calculation.

If setting a period longer, remaining months which suppress the influence of variations is calculated. However, take a time to display “remaining time” correctly after changing the motion or speed.

Setting value of HealthCalcPeriod is applied to all robot, joint, and part controlled by executed controller.

Note
Setting Period
Period which is set in HealthCalcPeriod command is running period of the controller.
Be careful that is not the same as actual time.

Calculation of “remaining months” and “consumption rate” when clearing.
Calculate remaining months every day in spite of the setting value of HealthCalcPeriod until exceeds the setting period for the first time after clearing the “Parts consumption commands information” of EPSON RC+, “remaining months” and “consumption rate” of HealthCtrlReset or HealthRBReset

See Also
HealthCalcPeriod Function, HealthCtrlInfo, HealthRBInfo, HealthCtrlReset, HealthRBReset

HealthCalcPeriod on Functional Example
> HealthCalcPeriod 3
> HealthCalcPeriod
 3
**HealthCalcPeriod Function**

Returns “remaining months” calculating period of the parts consumption commands information which is currently set.

**Syntax**

```
HealthCalcPeriod
```

**Return Values**

Returns calculating period in integer. (Unit: day)

**See Also**

HealthCalcPeriod

**HealthCalcPeriod on Functional Example**

Example to display the calculating period.

```
Print "period is", HealthCalcPeriod
```
HealthCtrlAlarmOn Function

Returns the status of the parts consumption alarm for the specified Controller parts.

Syntax

HealthCtrlAlarmOn(partType)

Parameters

partType  Integer expression (1) or the following constant representing the parts you want to obtain the alarm status.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_CONTROLLER_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
</tbody>
</table>

Return Values

True if the parts consumption alarm is occurring for the specified parts, otherwise False.

The parts consumption alarm occurs when the parts consumption rate obtained by HealthRateCtrlInfo exceeds 100%.

See Also

HealthCtrlInfo, HealthRateCtrlInfo

HealthCtrlAlarmOn Function Example

The example below determines if the parts consumption alarm is occurring for the Controller batteries.

```
Function PrintAlarm
    If HealthCtrlAlarmOn(HEALTH_CONTROLLER_TYPE_BATTERY) = True Then
        Print "Controller Battery NG"
    Else
        Print "Controller Battery OK"
    EndIf
EndFunction
```
HealthCtrlInfo Statement

Displays the remaining months before the recommended replacement time for the specified Controller parts.

Syntax

```plaintext
HealthCtrlInfo partType
```

Parameters

`partType`  
Integer expression (1) or the following constant representing the parts you want to obtain the remaining months before the recommended replacement time.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_CONTROLLER_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
</tbody>
</table>

Description

Displays the remaining months before the recommended replacement time for the specified Controller parts. The remaining months are calculated based on the parts consumption rate from the past usage and the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller.

Notes

Since the remaining months are calculated based on the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller, they cannot be calculated properly in the following cases:

- If the command is executed when the operating time is less than every operation of a period which is set in HealthCalcPeriod
- If the command is executed after the long-term operation stop period of the robot.
- If the command is executed after the parts consumption alarm is reset due to the parts replacement.
- If the time and date on the Controller is changed.

In above cases, execute the command after operating the Controller more than twice of setting period in HealthCalcPeriod to display the accurate value.

See Also

HealthCtrlAlarmOn, HealthRateCtrlInfo

HealthCtrlInfo Statement Example

The example below displays the remaining months before the recommended replacement time for the Controller batteries.

```plaintext
> HealthCtrlInfo HEALTH_CONTROLLER_TYPE_BATTERY BATTERY 240.000
> 
```
HealthCtrlInfo Function

Returns the remaining months before the recommended replacement time for the specified Controller parts.

Syntax

```
HealthCtrlInfo(partType)
```

Parameters

**partType**

Integer expression (1) or the following constant representing the parts you want to obtain the remaining months before the recommended replacement time.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_CONTROLLER_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
</tbody>
</table>

Return Values

Real number representing the remaining months before the recommended replacement time. (Unit: month)

Description

The remaining months are calculated based on the parts consumption rate from the past usage and the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller.

Notes

Since the remaining months are calculated based on the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller, they cannot be calculated properly in the following cases:
- If the command is executed when the operating time is less than every operation of a period which is set in HealthCalcPeriod.
- If the command is executed after the long-term operation stop period of the robot.
- If the command is executed after the parts consumption alarm is reset due to the parts replacement.
- If the time and date on the Controller is changed.

In above cases, execute the command after operating the Controller more than twice of setting period in HealthCalcPeriod to display the accurate value.

See Also

HealthCtrlAlarmOn, HealthRateCtrlInfo

HealthCtrlInfo Function Example

The example below outputs the alarm when the recommended replacement time is in less than one month.

```
Function AlarmCheck
    Real month
    month = HealthCtrlInfo(HEALTH_CONTROLLER_TYPE_BATTERY)
    If month < 1 Then
        Print "Alarm ON"
    EndIf
EndFunction
```
**HealthCtrlRateOffset Statement**

Sets the offset for the consumption rate of the specified parts.

**Syntax**

`HealthCtrlRateOffset partType, offset`

**Parameters**

- `partType` Integer expression (1) or the following constant representing the controller related parts.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH(Controller) TYPE BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
</tbody>
</table>

- `offset` Integer expression that specifies the offset value added to the consumption rate. (Unit: %)

**Description**

Sets the offset for the consumption rate of the specified parts.

**See Also**

HealthRBAlarmOn, HealthRateRBInfo, HealthRBInfo

**HealthCtrlRateOffset Statement Example**

The following is the example to add 10% to the parts consumption rate of the Controller batteries.

```
> HealthRBRateOffset HEALTH_CONTROLLER_TYPE_BATTERY, 10
>```

```
HealthCtrlReset Statement

Clears the remaining months before the recommended replacement time and the consumption rate for the specified parts.

Syntax

```
HealthCtrlReset partType
```

Parameters

<table>
<thead>
<tr>
<th>partType</th>
<th>Integer expression (1) or the following constant representing the controller related parts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_CONTROLLER_TYPE_BATTERY</td>
<td>1</td>
</tr>
</tbody>
</table>

Description

Clears the remaining months before the recommended replacement time and the consumption rate for the specified parts.
The warnings are also canceled.

See Also

HealthCtrlAlarmOn, HealthRateCtrlInfo, HealthCtrlInfo

HealthCtrlReset Statement Example

```
> HealthCtrlReset HEALTH_CONTROLLER_TYPE_BATTERY
> 
```
HealthCtrlWarningEnable

Enable or disable the parts consumption alarm notification of specified part related to the Controller.

Syntax
HealthCtrlWarningEnable partType [, On/Off]

Parameters

<table>
<thead>
<tr>
<th>partType</th>
<th>Integer expression or the following constant representing the controller.</th>
</tr>
</thead>
<tbody>
<tr>
<td>On/Off</td>
<td>On: Enable the parts consumption alarm notification.</td>
</tr>
<tr>
<td></td>
<td>Off: Disable the parts consumption alarm notification</td>
</tr>
</tbody>
</table>

Return Values
If On/Off parameters are omitted, the current On/Off settings are displayed.

Description
When the parts consumption alarm of the specified part occurs, set whether to notify the parts consumption alarm.

Notes
If the parts consumption alarm of the specified part is disabled, the parts consumption alarm will not be notified when the recommended replacement time is passed. Be careful to set when using this command.

See Also
HealthCtrlAlarmOn

HealthCtrlWarningEnable Example
Example to disable the parts consumption alarm of batteries of the controller.

    > HealthCtrlWarningEnable HEALTH_CONTROLLER_TYPE_BATTERY, Off

Example to display the parts consumption alarm settings of batteries of the controller.

    > HealthCtrlWarningEnable HEALTH_CONTROLLER_TYPE_BATTERY
    BATTERY Off
    >
HealthCtrlWarningEnable Function

Returns the setting status of the parts consumption alarm notification of specified part related to the Controller.

Syntax

```
HealthCtrlWarningEnable(partType)
```

Parameters

- `partType` Integer expression or the following constant representing the controller.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_CONTROLLER_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
</tbody>
</table>

Return Values

Returns the setting values of the parts consumption alarm in integer.

- 1: On
- 0: Off

See Also

HealthCtrlAlarmOn

HealthCtrlWarningEnable Function Example

Example to display the parts consumption alarm of batteries of the controller.

```
Print HealthCtrlWarningEnable(HEALTH_CONTROLLER_TYPE_BATTERY)
```
HealthRateCtrlInfo Function

Returns the consumption rate of the specified Controller parts.

Syntax

```
HealthRateCtrlInfo(partType)
```

Parameters

**partType**  
Integer expression (1) or the following constant representing the parts you want to obtain the remaining months before the recommended replacement time.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_CONTROLLER_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
</tbody>
</table>

Return Values

Real number representing the current parts consumption rate, when regarding the recommended replacement time as 100%. (Unit: %)

Description

This command calculates the parts consumption rate based on the actual operating condition.

Notes

The recommended replacement time is the recommended time to replace the parts calculated based on statistics.  
Replacement may be required before the consumption rate reaches 100%.  
In addition, the parts will not become immediately unusable when the consumption rate reaches 100%.  
However, it is recommended to replace the parts soon as the possibility of breakage increases after the consumption rate reaches 100%.

See Also

HealthCtrlAlarmOn, HealthCtrlInfo,

HealthRateCtrlInfo Function Example

The example below outputs the alarm when the consumption rate for the Controller batteries reaches 90%.

```
Function AlarmCheck
    Real HealthRate

    HealthRate = HealthRateCtrlInfo (HEALTH_CONTROLLER_TYPE_BATTERY)
    If HealthRate > 90 Then
        Print "Alarm ON"
    EndIf
EndIf
Fend
```
HealthRateRBInfo Function

Returns the consumption rate for the specified robot parts.

Syntax

**HealthRateRBInfo**(robotNumber, partType, jointNumber)

Parameters

- **robotNumber** Integer expression (1-16) representing the robot number you want to obtain the parts consumption rate.
- **partType** Integer expression (1-6) or the following constants representing the parts you want to obtain the consumption rate.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

- **jointNumber** Integer expression (1-9) representing the joint you want to obtain the parts consumption rate.

This command is unavailable for the additional axes.

Return Values

Real number representing the current parts consumption rate, when regarding the recommended replacement time as 100%. (Unit: %)

Returns “-1” when the robot does not have the specified parts.

Description

This command calculates the parts consumption rate based on the actual operating condition.

Notes

The recommended replacement time is the recommended time to replace the parts calculated based on statistics.
Replacement may be required before the consumption rate reaches 100%.
In addition, the parts will not become immediately unusable when the consumption rate reaches 100%.
However, it is recommended to replace the parts soon as the possibility of breakage increases after the consumption rate reaches 100%.

See Also

HealthRBAAlarmOn, HealthRBInfo
**HealthRateRBInfo Function Example**

The example below outputs the alarm when the consumption rate of the Joint #3 reduction gear unit on the robot 1 reaches 90%.

```spel
Function AlarmCheck
  Real HealthRate

  HealthRate = HealthRateRBInfo(1, HEALTH_ROBOT_TYPE_GEAR, 3)
  If HealthRate > 90 Then
    Print "Alarm ON"
  EndIf
EndFend
```
HealthRBAlarmOn Function

Returns the status of the parts consumption alarm for the specified robot parts.

Syntax

`HealthRBAlarmOn(robotNumber, partType, jointNumber)`

Parameters

- **robotNumber**
  Integer expression (1-16) representing the robot number you want to obtain the alarm status.

- **partType**
  Integer expression (1-6) or the following constants representing the parts you want to obtain the alarm status.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

- **jointNumber**
  Integer expression (1-9) representing the joint you want to obtain the alarm status. When the batteries are selected for `partType`, the same value will be returned when any joint is specified because the batteries are common to all joints. This command is unavailable for the additional axes.

Return Values

True if the parts consumption alarm is occurring for the specified parts, otherwise False.

The parts consumption alarm occurs when the parts consumption rate obtained by HealthRateRBInfo exceeds 100%.

Returns “-1” when the robot does not have the specified parts.

See Also

HealthRBInfo, HealthRateRBInfo

HealthRBAlarmOn Function Example

The example below determines if the parts consumption alarm is occurring for the grease on the Joint #3 of the robot 1.

```plaintext
Function PrintAlarm4
    If HealthRBAlarmOn(1, HEALTH_ROBOT_TYPE_GREASE, 3) = True Then
        Print "Robot1 Joint3 Grease NG"
    Else
        Print "Robot1 Joint3 Grease OK"
    EndIf
EndFunction
```
HealthRBAnalysis Statement

Simulates and displays the usable months for the specified parts in a particular robot operation cycle.

Syntax

HealthRBAnalysis robotNumber, partType[, jointNumber]

Parameters

<table>
<thead>
<tr>
<th>parameter</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>robotNumber</td>
<td>Integer expression (1-16) representing the robot number.</td>
</tr>
<tr>
<td>partType</td>
<td>Integer expression or the following constants representing the robot parts.</td>
</tr>
<tr>
<td>jointNumber</td>
<td>Integer expression (1-6) representing the joint. If the joint number is not specified, returns values for all the joints. This command is unavailable for the additional axes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_ALL</td>
<td>0</td>
<td>Specifies all parts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE BALL_SREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

Description

Simulates and displays the usable months for the specified parts in a particular robot operation cycle. This command calculates and displays how many months the parts can be used if they are new and used for 24 hours a day. The past usage is not considered.

Returns “-1” when the specified parts are not installed on the specified joint.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode (including the virtual controller).

See Also

HealthRBStart, HealthRBStop

HealthRBAnalysis Statement Example

The example below displays the usable months for all parts of all joints on SCARA robot.

```
> HealthRBAnalysis 1, HEALTH_ROBOT_TYPE_ALL
BELT       -1.000, -1.000, 38.689, 95.226
GREASE     -1.000, -1.000, 21.130, -1.000
MOTOR      240.000, 240.000, 240.000, 240.000
GEAR       240.000, 240.000, 240.000, 240.000
BALL_SREW_SPLINE -1.000, -1.000, 240.000, -1.000
>```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 305
The example below displays the usable months for the reduction gear units of all joints on SCARA robot.

```
> HealthRBAnalysis 1, HEALTH_ROBOT_TYPE_GEAR
GEAR              240.000, 224.357, -1.000, -1.000
```

The example below displays the usable months for the Joint #2 motor on 6-axis robot.

```
> HealthRBAnalysis 1, HEALTH_ROBOT_TYPE_MOTOR, 2
MOTOR              224.357
```
HealthRBAnalysis Function

Returns the usable months for the specified parts in a particular robot operation cycle.

Syntax

HealthRBAnalysis(robotNumber, partType, jointNumber)

Parameters

robotNumber Integer expression (1-16) representing the robot number.
partType Integer expression (2-6) or the following constants representing the robot parts.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

jointNumber Integer expression (1-6) representing the joint. This command is unavailable for the additional axes.

Return Values

Real number representing the usable months.
Returns “-1” when the specified parts are not installed on the specified joint.

Description

Simulates the usable months for the specified parts in a particular robot operation cycle. This command calculates how many months the parts can be used if they are new and used for 24 hours a day. The past usage is not considered.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode.

See Also

HealthRBStart, HealthRBStop
HealthRBAnalysis Function

HealthRBAnalysis Function Example

Function RobotPartAnalysis
    Real month

    Robot 1
    HealthRBStart 1
    Motor On
    Go P0
    Go P1
    Motor Off
    HealthRBStop 1

    month = HealthRBAnalysis(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
    Print "Ball Screw Spline analysis =", Str$(month)
    Fend
HealthRBDistance Statement

Displays the driving (rotation) amount of the motor of the specified joint.

Syntax

```
HealthRBDistance [robotNumber] [jointNumber]
```

Parameters

- **robotNumber**: Optional. Integer expression (1-16) representing the robot number. If omitted, the current robot number will be used.
- **jointNumber**: Integer expression (1-6) representing the joint. If the joint number is not specified, returns values for all the joints. This command is unavailable for the additional axes.

Description

Calculates and displays the driving (rotation) amount of the motor of the specified joint in robot operation from HealthRBStart to HealthRBStop. The past usage is not considered.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode.

See Also

HealthRBStart, HealthRBStop

HealthRBDistance Statement Example

The example below displays the driving amount of the Joint #1 of SCARA robot.

```plaintext
> HealthRBDistance 1, 1
1.000
> 
```
HealthRBDistance Function

Returns the driving (rotation) amount of the motor of the specified joint.

Syntax

HealthRBDistance([robotNumber,] jointNumber)

Parameters

robotNumber  Optional. Integer expression (1-16) representing the robot number.
If omitted, the current robot number will be used.

jointNumber  Integer expression (1-6) representing the joint. This command is unavailable for the
additional axes.

Return Values

Real number representing the driving amount.

Description

Returns the driving (rotation) amount of the motor of the specified joint in robot operation from
HealthRBStart to HealthRBStop. The past usage is not considered.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode.

See Also

HealthRBStart, HealthRBStop

HealthRBDistance Function Example

Function RobotPartAnalysis
  Real healthDistance

  Robot 1
  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  healthDistance = HealthRBDistance(1,1)
  Print "Distance =", Str$(healthDistance)
Fend
HealthRBInfo Statement

Displays the remaining months before the recommended replacement time for the specified robot parts.

Syntax

```
HealthRBInfo robotNumber, partType[, jointNumber]
```

Parameters

- **robotNumber**: Integer expression (1-16) representing the robot number you want to obtain the remaining months before the recommended replacement time.
- **partType**: Integer expression (0-6) or the following constants representing the parts you want to obtain the remaining months before the recommended replacement time.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_ALL</td>
<td>0</td>
<td>Specifies all parts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

- **jointNumber**: Integer expression (1-9) representing the joint you want to obtain the remaining months before the recommended replacement time. When the batteries are selected for `partType`, the same value will be returned when any joint is specified because the batteries are common to all joints. If the joint number is not specified, returns values for all the joints. This command is unavailable for the additional axes.

Description

Displays the remaining months before the recommended replacement time for the specified robot parts.

The remaining months are calculated based on the parts consumption rate from the past usage and the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller.

Returns “-1” when the robot joint does not have the specified parts.

Notes

Since the remaining months are calculated based on the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller, they cannot be calculated properly in the following cases:
- If this command is executed when the operating time is less than every operation of a period which is set in HealthCalcPeriod.
- If this command is executed after the long-term operation stop period of the robot.
- If this command is executed after the parts consumption alarm is reset after the parts replacement.
- If the time and date on the Controller is changed.

In above cases, execute the command after operating the Controller more than twice of setting period in HealthCalcPeriod to display the accurate value.

See Also

HealthRBAlarmOn, HealthRateRBInfo
HealthRBInfo Statement Example

The example below displays the remaining months for all parts of all joints on the robot 1.

```
> HealthRBInfo 1, HEALTH_ROBOT_TYPE_ALL
BATTERY        240.000
BELT           -1.000,  -1.000,  38.689,  95.226
GREASE         -1.000,  -1.000,  21.130,  -1.000
MOTOR          240.000,  240.000,  240.000,  240.000
GEAR           240.000,  224.357,  -1.000,  -1.000
BALL_SCREW_SPLINE -1.000,  -1.000,  240.000,  -1.000
>
```

The example below displays the remaining months for the reduction gear units of all joints on the robot 1.

```
> HealthRBInfo 1, HEALTH_ROBOT_TYPE_GEAR
GEAR           240.000,  224.357,  -1.000,  -1.000
>
```

The example below displays the remaining months for the Joint #2 motor of the robot 1.

```
> HealthRBInfo 1, HEALTH_ROBOT_TYPE_MOTOR, 2
MOTOR          224.357
>
```
HealthRBInfo Function

Returns the remaining months before the recommended replacement time for the specified robot parts.

Syntax

HealthRBInfo(robotNumber, partType, jointNumber)

Parameters

- **robotNumber**: Integer expression (1-16) representing the robot number you want to obtain the remaining months before the recommended replacement time.
- **partType**: Integer expression (1-6) or the following constants representing the parts you want to obtain the remaining months before the recommended replacement time.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

- **jointNumber**: Integer expression (1-9) representing the joint you want to obtain the remaining months before the recommended replacement time. When the batteries are selected for partType, the same value will be returned when any joint is specified because the batteries are common to all joints. This command is unavailable for the additional axes.

Return Values

Real number representing the remaining months before the recommended replacement time. (Unit: month) Returns “-1” when the robot does not have the specified parts.

Description

The remaining months are calculated based on the parts consumption rate from the past usage and the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller.

Notes

Since the remaining months are calculated based on the amount of change in the consumption rate obtained every operation of a period which is set in HealthCalcPeriod of the Controller, they cannot be calculated properly in the following cases:

- If this command is executed when the operating time is less than every operation of a period which is set in HealthCalcPeriod.
- If this command is executed after the long-term operation stop period of the robot.
- If this command is executed after the parts consumption alarm is reset after the parts replacement.
- If the time and date on the Controller is changed.

In above cases, execute the command after operating the Controller more than twice of setting period in HealthCalcPeriod to display the accurate value.
HealthRBInfo Function

See Also
HealthRBAalarmOn, HealthRateRBInfo

HealthRBInfo Function Example
The example below outputs the alarm when the recommended replacement time for the Joint #3 ball screw spline on the robot 1 is in less than one month.

```plaintext
Function AlarmCheck
    Real month

    month = HealthRBInfo(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
    If month < 1 Then
        Print "Alarm ON"
    EndIf
EndIf
```

Fend
HealthRBRateOffset Statement

Sets the offset for the consumption rate of the specified parts.

Syntax

HealthRBRateOffset robotNumber, partType, jointNumber, offset

Parameters

- **robotNumber**: Integer expression (1-16) representing the robot number.
- **partType**: Integer expression (1-6) or the following constants representing the robot part.
- **jointNumber**: Integer expression (1-6) representing the joint. When the batteries are selected for partType, the offset will be set when any joint is specified because the batteries are common to all joints. This command is unavailable for the additional axes.
- **offset**: Integer expression that specifies the offset value added to the consumption rate. (Unit: %)

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

Description

Sets the offset for the consumption rate of the specified parts and joints.

See Also

HealthRBAlarmOn, HealthRateRBInfo, HealthRBInfo

HealthRBRateOffset Example

The example below adds 10% to the consumption rate of the Joint #1 reduction gear unit on the robot 1.

```
HealthRBRateOffset 1, HEALTH_ROBOT_TYPE_GEAR, 1, 10
```

>
HealthRBReset Statement

Clears the remaining months before the recommended replacement time and the consumption rate for the specified parts.

Syntax

HealthRBReset robotNumber, partType, jointNumber

Parameters

robotNumber   Integer expression (1-16) representing the robot number.
partType      Integer expression (1-6) or the following constants representing the robot parts.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

jointNumber   Integer expression (1-6) representing the joint you want to obtain the remaining months before the recommended replacement time. When the batteries are selected for partType, the remaining months will be cleared when any joint is specified because the batteries are common to all joints. This command is unavailable for the additional axes.

Description

Clears the remaining months before the recommended replacement time and the consumption rate for the specified parts and joints.

The warnings are also canceled.

See Also

HealthRBAlarmOn, HealthRateRBInfo, HealthRBInfo

HealthRBReset Statement Example

> HealthRBReset 1,HEALTH_ROBOT_TYPE_GEAR,1
>
HealthRBSpeed Statement

Displays the average speed of the specified joint.

Syntax

HealthRBSpeed [robotNumber] [, jointNumber]

Parameters

robotNumber     Optional. Integer expression (1-16) representing the robot number.
                 If omitted, the current robot number will be used.

jointNumber     Integer expression (1-6) representing the joint. If the joint number is not specified, returns
                 values for all the joints. This command is unavailable for the additional axes.

Description

Returns the average of the absolute values for speed of the specified joint in robot operation from
HealthRBStart to HealthRBStop. The result is a real number from 0 to 1.
The maximum average speed is "1".
The value is "0" when the average value is 0.001 or less.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode.

See Also

HealthRBStart, HealthRBStop, AveSpeed

HealthRBSpeed Statement Example

The example below displays the speed of the Joint #1 of SCARA robot.

> HealthRBSpeed 1, 1
  0.100
>
HealthRBSpeed Function

Returns the average of the absolute values for speed of the specified joint.

Syntax

HealthRBSpeed ([robotNumber, ] jointNumber)

Parameters

robotNumber  Optional.  Integer expression (1-16) representing the robot number.  
If omitted, the current robot number will be used.

jointNumber  Integer expression (1-6) representing the joint.  This command is unavailable for the 
additional axes.

Return Values

The result is a real number from 0 to 1.

Description

Returns the average of the absolute values for speed of the specified joint in robot operation from 
HealthRBStart to HealthRBStop.  The result is a real number from 0 to 1.  The maximum average speed is 
"1".

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode (including the virtual controller).

See Also

HealthRBStart, HealthRBStop, AveSpeed

HealthRBSpeed Function Example

Function RobotPartAnalysis
  Real healthSpeed

  Robot 1
  HealthRBStart 1
  Motor On
  Go P0
  Go P1
  Motor Off
  HealthRBStop 1

  healthSpeed = HealthRBSpeed(1,1)
  Print "AveSpeed =", Str$(healthSpeed)
Fend
HealthRBStart Statement

Starts calculation of the usable months and elements for the parts in a particular robot operation cycle.

Syntax

HealthRBStart robotNumber

Parameters

robotNumber Integer expression (1-16) representing the robot number.

Description

Starts calculation of the usable months and elements (torque, speed, and driving amount) for the parts on
the specified robot in a particular robot operation cycle.

If this command is executed again when the calculation is already started, the previous calculation result
will be initialized.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode (including the virtual controller).

See Also

HealthRBAnalysis, HealthRBStop, HealthRBTRQ, HealthRBSpeed, HealthRBDistance

HealthRBStart Statement Example

Function RobotPartAnalysis
Real month

Robot 1

HealthRBStart 1
Motor On
Go P0
Go P1
Motor Off
HealthRBStop 1

month = HealthRBAnalysis(1, HEALTH_ROBOT_TYPE BALL SCREW SPLINE, 3)
Print "Ball Screw Spline analysis =", Str$(month)
Fend
HealthRBStop Statement

Stops calculation of the usable months and elements for the parts in a particular robot operation cycle.

Syntax

HealthRBStop robotNumber

Parameters

robotNumber Integer expression (1-16) representing the robot number.

Description

Stops calculation for the usable months and elements (torque, speed, and driving amount) of the parts on the specified robot in a particular robot operation cycle.

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode (including the virtual controller).
- Calculation automatically ends when one hour passes since calculation starts.
- If the command is executed after the automatic termination, an error will occur.
- If the command is executed without executing the HealthRBStart command, an error will occur.
- If the command is executed again without executing the HealthRBStart command after the previous HealthRBStop command, an error will occur.

See Also

HealthRBAnalysis, HealthRBStart, HealthRBTRQ, HealthRBSpeed, HealthRBDistance

HealthRBStop Statement Example

Function RobotPartAnalysis
    Real month

    Robot 1
    HealthRBStart 1
    Motor On
    Go P0
    Go P1
    Motor Off
    HealthRBStop 1

    month = HealthRBAnalysis(1, HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE, 3)
    Print "Ball Screw Spline analysis =", Str$(month)
Fend
HealthRBTRQ Statement

Displays the torque value which affects the life of the parts on the specified joint.

Syntax

HealthRBTRQ [robotNumber] [, jointNumber]

Parameters

robotNumber  Optional. Integer expression (1-16) representing the robot number. If omitted, the current robot number will be used.

jointNumber  Integer expression (1-6) representing the joint. If the joint number is not specified, returns values for all the joints. This command is unavailable for the additional axes.

Description

Displays the torque value which affects the life of the parts on the specified joint in robot operation from HealthRBStart to HealthRBStop. The result is a real number from 0 to 1. The maximum torque value is "1".

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode (including the virtual controller).

See Also

HealthRBStart, HealthRBStop, ATRQ

HealthRBTRQ Statement Example

The example below displays the torque value which affects the life of the parts on the Joint #1 of SCARA robot.

> HealthRBTRQ 1, 1
0.02
>


HealthRBTRQ Function

Returns the torque value which affects the life of the parts on the specified joint.

Syntax

HealthRBTRQ ([robotNumber,] jointNumber)

Parameters

robotNumber  Optional. Integer expression (1-16) representing the robot number.
If omitted, the current robot number will be used.

jointNumber  Integer expression (1-6) representing the joint. This command is unavailable for the additional axes.

Return Values

The result is a real number from 0 to 1.

Description

Returns the torque value which affects the life of the parts on the specified joint in robot operation from HealthRBStart to HealthRBStop. The result is a real value from 0 to 1. The maximum torque value is "1".

Notes

- This command does not function in Auto mode.
- This command does not function in dry run mode (including the virtual controller).

See Also

HealthRBStart, HealthRBStop, ATRQ

HealthRBTRQ Function Example

Function RobotPartAnalysis
    Real healthTRQ

    Robot 1

    HealthRBStart 1
    Motor On
    Go P0
    Go P1
    Motor Off
    HealthRBStop 1

    healthTRQ = HealthRBTRQ(1,1)
    Print "Torque =", Str$(healthTRQ)
Fend
Enable or disable the parts consumption alarm notification of specified part related to the robot.

Syntax
HealthRBWarningEnable robotNumber, partType [ , On/Off]

Parameters
- **robotNumber**: Integer expression (1-16) representing the robot number.
- **partType**: Integer expression (1-6) or the following constants representing the robot parts.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL(SCREW)_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

- **On/Off**: On: Enable the parts consumption alarm notification. Off: Disable the parts consumption alarm notification.

Return Values
If On/Off parameters are omitted, the current On/Off settings are displayed.

Description
When the parts consumption alarm of the specified part occurs, set whether to notify the parts consumption alarm.

Notes
If the parts consumption alarm of the specified part is disabled, the parts consumption alarm will not be notified when the recommended replacement time is passed. Be careful to set when using this command.

See Also
HealthRBAAlarmOn
HealthRBWarningEnable

**HealthRBWarningEnable Example**
Example to disable the parts consumption alarm of the grease part of SCARA robot 1.

> HealthRBWarningEnable 1, HEALTH_ROBOT_TYPE_GREASE, Off

Example to display the parts consumption alarm settings of the grease part of SCARA robot 1.

> HealthRBWarningEnable 1, HEALTH_ROBOT_TYPE_GREASE
GREASE Off
>
HealthRBWarningEnable Function

Returns the setting status of the parts consumption alarm notification of specified part related to the robot.

Syntax
HealthRBWarningEnable(robotNumber, partType)

Parameters
- **robotNumber**: Integer expression (1-16) representing the robot number you want to obtain the remaining months before the recommended replacement time.
- **partType**: Integer expression (1-6) or the following constants representing the parts you want to obtain the remaining months before the recommended replacement time.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEALTH_ROBOT_TYPE_BATTERY</td>
<td>1</td>
<td>Specifies the batteries.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BELT</td>
<td>2</td>
<td>Specifies the timing belts.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GREASE</td>
<td>3</td>
<td>Specifies the grease.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_MOTOR</td>
<td>4</td>
<td>Specifies the motors.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_GEAR</td>
<td>5</td>
<td>Specifies the reduction gear units.</td>
</tr>
<tr>
<td>HEALTH_ROBOT_TYPE_BALL_SCREW_SPLINE</td>
<td>6</td>
<td>Specifies the ball screw spline.</td>
</tr>
</tbody>
</table>

Return Values
Returns the setting values of the parts consumption alarm in integer.
1: On
0: Off

See Also
HealthRBAAlarmOn

HealthRBWarningEnable Function Example
Example to display the parts consumption alarm settings of the grease part of SCARA robot 1.

Print HealthRBWarningEnable(1, HEALTH_ROBOT_TYPE_GREASE )
Here Statement

Teach a robot point at the current position.

Syntax

Here point

Parameters

point Pnumber or P(expr) or point label.

Notes

The Here statement and Parallel Processing

You cannot use both of the Here statement and parallel processing in one motion command like this:

Go Here :Z(0) ! D10; MemOn 1 !

Be sure to change the program like this:

P999 = Here
Go P999 Here :Z(0) ! D10; MemOn 1 !

The Here statement and Multitask

If the Here statement is executed in a multitask function executed by Xqt while the robot is moved by Move, Go, etc., in the main task, the task will be stopped due to an error. Current robot position can be retrieved by CurPos.

Example

Function Xqt_PrintHere
    Do
        Print CurPOS
        Wait 0.1
    Loop
Fend
Function main
    Xqt 10, Xqt_PrintHere
    Go P0
Fend

See Also

Here Function, CurPos

Here Statement Example

Here P1
Here pick
Here Function

Returns current robot position as a point.

**Syntax**

Here

**Return Values**

A point representing the current robot position.

**Description**

Use Here to retrieve the current position of the current manipulator.

**See Also**

Here Statement

**Here Function Example**

\[ P1 = \text{Here} \]
Returns a string representing a specified number in hexadecimal format.

**Syntax**

`Hex$(number)`

**Parameters**

`number` Integer expression.

**Return Values**

Returns a string containing the ASCII representation of the number in hexadecimal format.

**Description**

Hex$ returns a string representing the specified number in hexadecimal format. Each character is from 0 to 9 or A to F. Hex$ is especially useful for examining the results of the Stat function.

**See Also**

Str$, Stat, Val

**Hex$ Function Example**

```
> print hex$(stat(0))
A00000
> print hex$(255)
FF
```
Hofs Statement

Displays or sets the offset pulses between the encoder origin and the home sensor.

Syntax

(1) `Hofs j1Pulses, j2Pulses, j3Pulses, j4Pulses [j5Pulses, j6Pulses] [j7Pulses] [j8Pulses, j9Pulses]

(2) `Hofs

Parameters

`j1Pulses`  Integer expression representing joint 1 offset pulses.
`j2Pulses`  Integer expression representing joint 2 offset pulses.
`j3Pulses`  Integer expression representing joint 3 offset pulses.
`j4Pulses`  Integer expression representing joint 4 offset pulses.
`j5Pulses`  For 6 axis robots (including N series). Integer expression representing joint 5 offset pulses.
`j6Pulses`  For 6 axis robots (including N series). Integer expression representing joint 6 offset pulses.
`j7Pulses`  For 7 axis robots. Integer expression representing joint 7 offset pulses.
`j8Pulses`  For additional S axis. Integer expression representing joint 8 (additional S axis) offset pulses.
`j9Pulses`  For additional T axis. Integer expression representing joint 9 (additional T axis) offset pulses.

Return Values

Displays current Hofs values when used without parameters.

Description

Hofs displays or sets the home position offset pulses. Hofs specifies the offset from the encoder 0 point (Z phase) to the mechanical 0 point.

Although the robot motion control is based on the zero point of the encoder mounted on each joint motor, the encoder zero point may not necessarily match the robot mechanical zero point. The Hofs offset pulse correction pulse is used to carry out a software correction to the mechanical 0 point based on the encoder 0 point.

Notes

**Hofs Values SHOULD NOT be Changed unless Absolutely Necessary**

The Hofs values are correctly specified prior to delivery. There is a danger that unnecessarily changing the Hofs value may result in position errors and unpredictable motion. Therefore, it is strongly recommended that Hofs values not be changed unless absolutely necessary.

**To Automatically Calculate Hofs Values**

To have Hofs values automatically calculated, move the arm to the desired calibration position, and execute Calib. The controller then automatically calculates Hofs values based on the CalPls pulse values and calibration position pulse values.

**Saving and Restoring Hofs**

Hofs can be saved and restored using the Save and Load commands in the [System Configuration] dialog-[Robot]-[Calibration] from the System Configuration menu.
See Also
   Calib, CalPls, Home, Hordr, MCal, SysConfig

Hofs Statement Example
These are simple examples on the monitor window that first sets the joint 1 home offset value to be -545, the joint 2 home offset value to be 514, and the joint 3 and the joint 4 Home offset values to be both 0. It then displays the current home offset values.

   > hofs  -545, 514, 0, 0

   > hofs
   -545, 514, 0, 0

   >
Hofs Function

Returns the offset pulses used for software zero point correction.

Syntax

Hofs(jointNumber)

Parameters

jointNumber

Integer expression representing the joint number to retrieve the Hofs value for. The additional S axis is 8 and T axis is 9.

Return Values

The offset pulse value (integer value, in pulses).

See Also

Calib, CalPls, Home, Hordr, MCal, SysConfig

Hofs Function Example

This example uses the Hofs function in a program:

```plaintext
Function DisplayHofs
    Integer i
    Print "Hofs settings:"
    For i = 1 To 4
        Print "Joint ", i, ", " = ", Hofs(i)
    Next i
End
```
Home Statement

Moves the robot arm to the user defined home position.

Syntax

Home

Description

Executes low speed Point to Point motion to the Home (standby) position specified by HomeSet, in the homing order defined by Hordr.

Normally, for SCARA robots (including RS series), the Z joint (J3) returns first to the HomeSet position, then the J1, J2 and J4 joints simultaneously return to their respective HomeSet coordinate positions. The Hordr instruction can change this order of the axes returning to their home positions.

Note

Home Status Output:

When the robot is in its Home position, the controller's system Home output is turned ON.

Potential Error

Attempting to Home without HomeSet Values Defined

Attempting to Home the robot without setting the HomeSet values will result in an Error 2228 being issued.

See Also

HomeClr, HomeDef, HomeSet, Hordr

Home Statement Example

The Home instruction can be used in a program such as this:

```plaintext
Function InitRobot
    Reset
    If Motor = Off Then
        Motor On
    EndIf
    Home
EndFunction
```

Or it can be issued from the Command window like this:

```
> home
>```
Clears the home position definition.

Syntax

```
HomeClr
```

Description

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also

```
HomeDef, HomeSet
```

HomeClr Function Example

This example uses the HomeClr function in a program:

```
Function ClearHome
    If HomeDef = True Then
        HomeClr
    EndIf
EndFunction
```
HomeDef Function

Returns whether home position has been defined or not.

Syntax

HomeDef

Return Values

True if home position has been defined, otherwise False.

See Also

HomeClr, HomeSet

HomeDef Function Example

This example uses the HomeDef function in a program:

```plaintext
Function DisplayHomeSet
    Integer i
    If HomeDef = False Then
        Print "Home is not defined"
    Else
        Print "Home values:"
        For i = 1 To 4
            Print "J", i, " = ", HomeSet(i)
        Next i
    EndIf
EndFunction
```
HomeSet Statement

Specifies and displays the Home position.

Syntax

(1) **HomeSet** *j1Pulses*, *j2Pulses*, *j3Pulses*, *j4Pulses*  

\[[, j5Pulses, j6Pulses\] [, j7Pulses\] [, j8Pulses, j9Pulses\]

(2) **HomeSet**

Parameters

- **j1Pulses**: The home position encoder pulse value for joint 1.
- **j2Pulses**: The home position encoder pulse value for joint 2.
- **j3Pulses**: The home position encoder pulse value for joint 3.
- **j4Pulses**: The home position encoder pulse value for joint 4.
- **j5Pulses**: Optional for 6-axis robots (including N series). The home position encoder pulse value for joint 5.
- **j6Pulses**: Optional for 6-axis robots (including N series). The home position encoder pulse value for joint 6.
- **j7Pulses**: Optional for Joint type 7-axis robots. The home position encoder pulse value for joint 7.
- **j8Pulses**: Optional for additional S axis. The home position encoder pulse value for joint 8 (additional S axis).
- **j9Pulses**: Optional for additional T axis. The home position encoder pulse value for joint 9 (additional T axis).

Return Values

Displays the pulse values defined for the current Home position when parameters are omitted.

Description

Allows the user to define a new home (standby) position by specifying the encoder pulse values for each of the robot joints.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Potential Errors

**Attempting to Home without HomeSet Values Defined:**

Attempting to Home the robot without setting the HomeSet values will result in an Error 2228 being issued.

**Attempting to Display HomeSet Values without HomeSet Values Defined:**

Attempting to display home position pulse values without HomeSet values defined causes an Error 2228.

See Also

Home, HomeClr, HomeDef, Hordr, Pls
HomeSet Statement

HomeSet Statement Example
  The following examples are done from the monitor window:

  > homeset 0,0,0,0  'Set Home position at 0,0,0,0
  > homeset
       0 0
       0 0

  > home            'Robot homes to 0,0,0,0 position

Using the Pls function, specify the current position of the arm as the Home position.

  > homeset P1s(1), P1s(2), P1s(3), P1s(4)
HomeSet Function

Returns pulse values of the home position for the specified joint.

Syntax

HomeSet(jointNumber)

Parameters

jointNumber Integer expression representing the joint number to retrieve the HomeSet value for. The additional S axis is 8 and T axis is 9.

Return Values

Returns pulse value of joint home position. When jointNumber is “0”, returns “1” when HomeSet has been set or “0” if not.

See Also

HomeSet Statement

HomeSet Function Example

This example uses the HomeSet function in a program:

Function DisplayHomeSet

    Integer i

    If HomeSet(0) = 0 Then
        Print "HomeSet is not defined"
    Else
        Print "HomeSet values:",
        For i = 1 To 4
            Print "J", i, " = ", HomeSet(i)
        Next i
    EndIf
EndFunction
Hordr Statement

Specifies or displays the order of the axes returning to their Home positions.

Syntax
(2) **Hordr**

Parameters
- **step1**: Bit pattern that defines which joints should home during the 1st step of the homing process.
- **step2**: Bit pattern that defines which joints should home during the 2nd step of the homing process.
- **step3**: Bit pattern that defines which joints should home during the 3rd step of the homing process.
- **step4**: Bit pattern that defines which joints should home during the 4th step of the homing process.
- **step5**: Bit pattern that defines which joints should home during the 5th step of the homing process.
- **step6**: Bit pattern that defines which joints should home during the 6th step of the homing process.
- **step7**: Bit pattern that defines which joints should home during the 7th step of the homing process.
- **step8**: Bit pattern that defines which joints should home during the 8th step of the homing process.
- **step9**: Bit pattern that defines which joints should home during the 9th step of the homing process.

Return Values
Displays current Home Order settings when parameters are omitted.

Description
Hordr specifies joint motion order for the Home command. (i.e. Defines which joint will home 1st, which joint will home 2nd, 3rd, etc.)

The purpose of the Hordr instruction is to allow the user to change the homing order. The homing order is broken into 4, 6, or 9 separate steps, depending on robot type. The user then uses Hordr to define the specific joints which will move to the Home position during each step. It is important to realize that more than one joint can be defined to move to the Home position during a single step. This means that all joints can potentially be homed at the same time. For SCARA robots (including RS series, 4 axis robots), it is recommended that the Z joint normally be defined to move to the Home position first (in Step 1) and then allow the other joints to follow in subsequent steps.

The Hordr instruction expects that a bit pattern be defined for each of the steps. Each joint is assigned a specific bit. When the bit is set to “1” for a specific step, then the corresponding joint will home. When the bit is cleared to “0”, then the corresponding axis will not home during that step. The joint bit patterns are assigned as follows:

<table>
<thead>
<tr>
<th>Joint:</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit Number:</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>Binary Code:</td>
<td>&amp;B0001</td>
<td>&amp;B0010</td>
<td>&amp;B0100</td>
<td>&amp;B1000</td>
<td>&amp;B100</td>
<td>&amp;B100</td>
<td>&amp;B100</td>
<td>&amp;B100</td>
<td>&amp;B100</td>
</tr>
<tr>
<td></td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
</tbody>
</table>
Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also
Home, HomeSet

Hordr Statement Example
Following are some command window examples for SCARA robots (including RS series, 4 axis robots):

This example defines the home order as J3 in the first step, J1 in second step, J2 in third step, and J4 in the fourth step. The order is specified with binary values.

```
> hordr &B0100, &B0001, &B0010, &B1000
```

This example defines the home order as J3 in the first step, then J1, J2 and J4 joints simultaneously in the second step. The order is specified with decimal values.

```
> hordr 4, 11, 0, 0
```

This example displays the current home order in decimal numbers.

```
> hordr
4, 11, 0, 0
>
```
Hordr Function

Returns Hordr value for a specified step.

Syntax

\[
\text{Hordr}(\text{stepNumber})
\]

Parameters

\text{stepNumber}  \quad \text{Integer expression representing which Hordr step to retrieve.}

Return Values

Integer containing the Hordr value for the specified step.

See Also

Home, HomeSet

Hordr Function Example

\[
\text{Integer a} \\
\text{a} = \text{Hordr}(1)
\]
Hour Statement

Displays the accumulated controller operating time.

Syntax

    Hour

Description

Displays the amount of time the controller has been turned on and running SPEL. (Accumulated Operating Time) Time is always displayed in units of hours.

See Also

    Time

Hour Statement Example

The following example is done from the Command window:

```
> hour
2560
>
```
Hour Function

Returns the accumulated controller operating time.

Syntax

Hour

Return Values

Returns accumulated operating time of the controller (real number, in hours).

See Also

Time

Hour Function Example

Print "Number of controller operating hours: ", Hour
If…Then…Else…EndIf Statement

Executes instructions based on a specified condition.

Syntax

(1) If condition Then
    stmtT1
    ...
[Else if condition Then]
    stmtT1
    ...
[Else]
    stmtF1
    ...
EndIf

(2) If condition Then stmtT1 [; stmtT2... ] [Else stmtF1 [; stmtF2... ]]

Parameters

condition Any valid test condition which returns a True (any number besides “0”) or False result (returned as a “0”). (See sample conditions below)
stmtT1 Executed when the condition is True. (Multiple statements may be put here in a blocked If...Then...Else style.)
stmtF1 Executed when the condition is False. (Multiple statements may be put here in a blocked If...Then...Else style.)

Description

(1) If...Then...Else executes stmtT1, etc. when the conditional statement is True. If the condition is False then stmtF1, etc. are executed. The Else portion of the If...Then...Else instruction is optional. If you omit the Else statement and the conditional statement is False, the statement following the EndIf statement will be executed. For blocked If...Then...Else statements the EndIf statement is required to close the block regardless of whether an Else is used or not.

(2) If...Then...Else can also be used in a non blocked fashion. This allows all statements for the If...Then...Else to be put on the same line. Please note that when using If...Then...Else in a non blocked fashion, the EndIf statement is not required. If the If condition specified in this line is satisfied (True), the statements between the Then and Else are executed. If the condition is not satisfied (False), the statements following Else are executed. The Else section of the If...Then...Else is not required. If there is no Else keyword then control passes on to the next statement in the program if the If condition is False.

The logical output of the conditional statement is any number excluding “1” when it is True, and “0” when it is false.
Notes

Sample Conditions:

- $a = b$ : $a$ is equal to $b$
- $a < b$ : $b$ is larger than $a$
- $a >= b$ : $a$ is greater than or equal to $b$
- $a <> b$ : $a$ is not equal to $b$
- $a > b$ : $b$ is smaller than $a$
- $a <= b$ : $a$ is less than or equal to $b$

Logical operations And, Or and Xor may also be used.

True in the Conditions:

Constant True is $-1$ and the type is Boolean, so you need to be careful when using it in a comparing condition with other type variable.

```
Function main
    Integer i
    i = 3
    If i = True Then
        Print "i=TRUE"
    EndIf
EndFunction
```

When you execute the program above, “i=TRUE” is displayed.
The judgement of condition including the Boolean type is done with “0” or “non-0”.
If the value of “i” is not “0”, it is considered that the condition is established and “i=TRUE” is displayed.

See Also

Else, Select...Case, Do...Loop

If/Then/Else Statement Example

<Single Line If...Then...Else>

The following example shows a simple function which checks an input to determine whether to turn a specific output on or off. This task could be a background I/O task which runs continuously.

```
Function main
    Do
        If Sw(0) = 1 Then On 1 Else Off 1
    Loop
EndFunction
```
<Blocked If...Then...Else>

The following example shows a simple function which checks a few inputs and prints the status of these inputs:

```
If Sw(0) = 1 Then Print "Input0 ON" Else Print "Input0 OFF"
If Sw(1) = 1 Then
  If Sw(2) = 1 Then
    Print "Input1 On and Input2 ON"
  Else
    Print "Input1 On and Input2 OFF"
  EndIf
Else
  If Sw(2) = 1 Then
    Print "Input1 Off and Input2 ON"
  Else
    Print "Input1 Off and Input2 OFF"
  EndIf
EndIf
```

<Other Syntax Examples>

```
If x = 10 And y = 3 Then GoTo 50
If test <= 10 Then Print "Test Failed"
If Sw(0) = 1 Or Sw(1) = 1 Then Print "Everything OK"
```
ImportPoints Statement

Imports a point file into the current project for the specified robot.

Syntax

```plaintext
ImportPoints sourcePath, filename [, robotNumber]
```

Parameters

- **sourcePath**: String expression containing the specific path and file to import into the current project. The extension can be ".pts" or ".pnt" (EPSON RC+ 3.x and 4.x format). See ChDisk for the details.

- **fileName**: String expression containing the specific file to be imported to in the current project for the current robot. The extension must be ".pts". You cannot specify a file path and fileName doesn’t have any effect from ChDisk. See ChDisk for the details.

- **robotNumber**: Optional. Integer expression that specifies which robot the point file should be associated with. If robotNumber = 0, then the point file is imported as a common point file. If robotNumber is omitted, the current robot number is used.

Description

ImportPoints copies a point file into the current project and adds it to the project files for the specified robot. The point file is then compiled and is ready for loading using the LoadPoints command. If the file already exists for the current robot, it will be overwritten and recompiled.

The point data is stored in the Compact Flash inside of the controller. Therefore, ImportPoints starts writing into the Compact Flash. Frequent writing into the Compact Flash will shorten the Compact Flash lifetime. We recommend using ImportPoints only for saving the point data.

Potential Errors

- **File Does Not Exist**: If `sourcePath` does not exist, an error will occur.
- **A Path Cannot be Specified**: If `fileName` contains a path, an error will occur.
- **Point file for another robot**: If `fileName` is a point file for another robot, an error will occur.

See Also

- LoadPoints, Robot, SavePoints

ImportPoints Statement Example

```plaintext
Function main
   Robot 1
   ImportPoints "c:\mypoints\modell.pts", "robot1.pts"
   LoadPoints "robot1.pts"
Fend
```
In Function

Returns the status of the specified Byte port. Each port contains 8 input channels.

Syntax

\[ \text{In}(\text{byteportNumber}) \]

Parameters

\text{byteportNumber} \quad \text{Integer number representing one eight bit port (one byte).}

Return Values

Returns an integer value between 0 and 255. The return value is 8 bits, with each bit corresponding to 1 input channel.

Description

\text{In} provides the ability to look at the value of 8 input channels at the same time. The \text{In} instruction can be used to store the 8 I/O channels status into a variable or it can be used with the \text{Wait} instruction to Wait until a specific condition which involves more than 1 I/O channel is met.

Since 8 channels are checked at a time, the return values range from 0 to 255. Please review the chart below to see how the integer return values correspond to individual input channels.

<table>
<thead>
<tr>
<th>Input Channel Result (Using Byte port #0)</th>
<th>Return Values</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>15</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Channel Result (Using Byte port #2)</th>
<th>Return Values</th>
<th>23</th>
<th>22</th>
<th>21</th>
<th>20</th>
<th>19</th>
<th>18</th>
<th>17</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td></td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

See Also

\text{InBCD}, \text{MemIn}, \text{MemOff}, \text{MemOn}, \text{MemSw}, \text{Off}, \text{On}, \text{OpBCD}, \text{Oport}, \text{Out}, \text{Sw}, \text{Wait}
In Function Example

For the example below let’s assume that input channels 20, 21, 22, and 23 are all connected to sensory devices such that the application should not start until each of these devices are returning an On signal indicating everything is OK to start. The program example gets the 8 input channels status of byte port 2 and makes sure that channels 20, 21, 22, and 23 are each On before proceeding. If they are not On (i.e. returning a value of 1) an error message is given to the operator and the task is stopped.

In the program, the variable “var1” is compared against the number 239 because in order for inputs 20, 21, 22, and 23 to all be On, then the result of In(2) will be 240 or larger. (We don’t care about Inputs 16, 17, 18, and 19 in this case so any values between 240-255 will allow the program to proceed.)

```
    Function main
        Integer var1
        var1 = In(2)   ' Get 8 input channels status of byte port 2
        If var1 > 239 Then
            Go P1
            Go P2
            ' Execute other motion statements here
            ',
        Else
            Print "Error in initialization!"
            Print "Sensory Inputs not ready for cycle start"
            Print "Please check inputs 20, 21, 22, and 23 for"
            Print "proper state for cycle start and then"
            Print "start program again"
        EndIf
    EndFend
```

We cannot set inputs from the command window but we can check them. For the examples shown below, we will assume that the Input channels 1, 5, and 15 are On. All other inputs are Off.

```
> print In(0)
  34
> print In(1)
  128
> print In(2)
  0
```
InBCD Function

Returns the input status of 8 inputs using BCD format. (Binary Coded Decimal)

Syntax

\[ \text{InBCD}(\text{portNumber}) \]

Parameters

- \( \text{portNumber} \): Integer number representing one eight bit port (one byte).

Return Values

Returns as a Binary Coded Decimal (0-9), the input status of the input port (0 to 99).

Description

InBCD simultaneously reads 8 input lines using the BCD format. The \( \text{portNumber} \) parameter for the InBCD instruction defines which group of 8 inputs to read where \( \text{portNumber} = 0 \) means inputs 0 to 7, \( \text{portNumber} = 1 \) means inputs 8 to 15, etc.

The resulting value of the 8 inputs is returned in BCD format. The return value may have 1 or 2 digits between 0 and 99. The 1st digit (or 10's digit) corresponds to the upper 4 outputs of the group of 8 outputs selected by \( \text{portNumber} \). The 2nd digit (or 1’s digit) corresponds to the lower 4 outputs of the group of 8 outputs selected by \( \text{portNumber} \).

Since valid entries in BCD format range from 0 to 9 for each digit, every I/O combination cannot be met. The table below shows some of the possible I/O combinations and their associated return values assuming that \( \text{portNumber} \) is 0.

<table>
<thead>
<tr>
<th>Return Values</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>02</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>03</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>08</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>09</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>10</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>11</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>99</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
</tbody>
</table>

Notice that the Binary Coded Decimal format only allows decimal values to be specified. This means that through using Binary Coded Decimal format it is impossible to retrieve a valid value if all inputs for a specific port are turned on at the same time when using the InBCD instruction. The largest value possible to be returned by InBCD is 99. In the table above it is easy to see that when 99 is the return value for InBCD, all inputs are not on. In the case of a return value of 99, inputs 0, 3, 4, and 7 are On and all the others are Off.
InBCD Function

Note

**Difference between InBCD and In**

The InBCD and In instructions are very similar in the SPEL+ language. However, there is one major difference between the two. This difference is shown below:

- The InBCD instruction uses the Binary Coded Decimal format for specifying the return value format for the 8 inputs. Since Binary Coded Decimal format precludes the values of &HA, &HB, &HC, &HD, &HE or &HF from being used, all combinations for the 8 inputs cannot be satisfied.
- The In instruction works very similarly to the InBCD instruction except that In allows the return value for all 8 inputs to be used. (i.e. 0 to 255 vs. 0 to 99 for InBCD) This allows all possible combinations for the 8 bit input groups to be read.

**See Also**

In, MemOff, MemOn, MemOut, MemSw, Off, On, OpBCD, Oport, Out, Sw, Wait

**InBCD Function Example**

Some simple examples from the Command window are as follows:

Assume that inputs 0, 4, 10, 16, 17, and 18 are all On (The rest of the inputs are Off).

```
> Print InBCD (0)
11
> Print InBCD (1)
04
> Print InBCD (2)
07
>
```
Inertia Statement

Specifies load inertia and eccentricity for current robot.

Syntax

\[
\text{Inertia} \ [ \text{loadInertia} ] [, \text{eccentricity} ]
\]

Parameters

- **loadInertia**: Optional. Real expression that specifies total moment of inertia in kgm2 around the center of the end effector joint, including end effector and part.
- **eccentricity**: Optional. Real expression that specifies eccentricity in mm around the center of the end effector joint, including end effector and part.

Return Values

When parameters are omitted, the current Inertia parameters are displayed.

Description

Use the Inertia statement to specify the total moment of inertia for the load on the end effector joint. This allows the system to more accurately compensate acceleration, deceleration, and servo gains for end effector joint. You can also specify the distance from the center of end effector joint to the center of gravity of the end effector and part using the `eccentricity` parameter.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also

Inertia Function

Inertia Statement Example

\[
\text{Inertia} \ 0.02, 1
\]
Inertia Function

Returns inertia parameter value.

Syntax

\[ \text{Inertia} \left( \text{paramNumber} \right) \]

Parameters

\( \text{paramNumber} \) Integer expression which can have the following values:

- 0: Causes function to return “1” if robot supports inertia parameters or “0” if not.
- 1: Causes function to return load inertia in kgm².
- 2: Causes function to return eccentricity in mm.

Return Values

Real value of the specified setting.

See Also

Inertia Statement

Inertia Function Example

\[
\begin{align*}
\text{Real loadInertia, eccentricity}\\
\text{loadInertia} = \text{Inertia}(1)\\
\text{eccentricity} = \text{Inertia}(2)
\end{align*}
\]
InPos Function

Returns the position status of the specified robot.

Syntax

InPos

Return Values

True if position has been completed successfully, otherwise False.

See Also

CurPos, FindPos, WaitPos

InPos Function Example

Function main

    P0 = XY(0, -100, 0, 0)
    P1 = XY(0, 100, 0, 0)

    Xqt MonitorPosition
    Do
        Jump P0
        Wait .5
        Jump P1
        Wait .5
    Loop

Fend

Function MonitorPosition

    Boolean oldInPos, pos

    Do
        Pos = InPos
        If pos <> oldInPos Then
            Print "InPos = ", pos
        EndIf
        oldInPos = pos
    Loop

Fend
Input Statement

Input Statement

Receives input data from the display device and stored in a variable(s).

Syntax

Input varName [ , varName, varName,... ]

Parameters

varName Variable name. Multiple variables can be used with the Input command as long as they are separated by commas.

Description

Input receives data from the display device and assigns the data to the variable(s) used with the Input instruction.

When executing the Input instruction, a “?” prompt appears at the display device. After inputting data press the return key (Enter) on the keyboard.

Notes

Rules for Numeric Input

When inputting numeric values and non-numeric data is found in the input other than the delimiter (comma), the Input instruction discards the non-numeric data and all data following that non-numeric data.

Rules for String Input

When inputting strings, numeric and alpha characters are permitted as data.

Other Rules for the Input Instruction

- When more than one variable is specified in the instruction, the numeric data input intended for each variable has to be separated by a comma (",") character.
- Numeric variable names and string variable names are allowed. However, the input data type must match the variable type.

Potential Error

Number of variables and input data differ

For multiple variables, the number of input data must match the number of Input variable names. When the number of the variables specified in the instruction is different from the number of numeric data received from the keyboard, an Error 2505 will occur.

See Also

Input #, Line Input, Line Input #, Print, String
Input Statement Example

This is a simple program example using Input statement.

Function InputNumbers
    Integer A, B, C

    Print "Please enter 1 number"
    Input A
    Print "Please enter 2 numbers separated by a comma"
    Input B, C
    Print "A = ", A
    Print "B = ", B, "C = ", C

Fend

A sample session of the above program running is shown below:
(Use the Run menu or F5 key to start the program)

Please enter 1 number
?-10000
Please enter 2 numbers separated by a comma
?25.1, -99
25.1 -99
B = 25.1 C = -99
>

>
Input # Statement

Allows string or numeric data to be received from a file, communications port, or database and stored in one or more variables.

Syntax

\[ \text{Input} \ # \text{portNumber}, \text{varName} \ [\ , \text{varName}, \text{varName}, \ldots] \]

Parameters

- **#portNumber**: The ID number that specifies a file, communication port, database, or device. The File number can be specified in ROpen, WOpen, and AOpen statements. Communication port number can be specified in OpenCom (RS-232C) and OpenNet (TCP/IP) statements. The database number can be specified in OpenDB statement.
  
  Device ID is:
  
  21 RC+
  24 TP (TP1 only)
  20 TP3

- **varName**: Variable name to receive the data.

Description

The Input # instruction receives numeric or string data from the device specified by handle, and assigns the data to the variable(s).

Notes

**Rules for Numeric Input**

When inputting numeric values and non-numeric data is found in the input other than the delimiter (comma), the Input instruction discards the non-numeric data and all data following that non-numeric data.

**Rules for String Input**

When inputting strings, numeric and alpha characters are permitted as data.

**Maximum data length**

This command can handle up to 256 bytes. However, the target is the database, it can handle up to 4096 bytes. If the target is the communication port (TCP/IP), it can handle up to 1024 bytes.

**Other Rules for the Input Instruction**

- When more than one variable is specified in the instruction, the numeric data input intended for each variable has to be separated by a comma ("," character or blank (" ").
- When more than one string variable or both of numeric variable and string variable is specified, the numeric data has to be separated by a comma ("," character or blank (" ").
- The input data type must match the variable type.

The following programs are examples to exchange the string variable and numeric variable between the controllers using a communication port.

Sending end (Either pattern is OK.)

\[
\text{Print} \ #\text{PortNum}, \"\$Status\", \\text{InData}, \text{OutData} \\
\text{Print} \ #\text{PortNum}, \"\$Status\", \",\", \text{InData}, \text{OutData}
\]

Receiving end

\[ \text{Input} \ #\text{PortNum}, \text{Response}\$, \text{InData}, \text{OutData} \]
Potential Error

**Number of variables and input data differ**

When the number of the variables specified in the instruction is different from the number of numeric data received from the device, an Error 2505 will occur.

---

See Also

Input, Line Input, Line Input #, Print #

---

**Input # Statement Example**

This function shows some simple Input # statement examples.

```plaintext
Function GetData
    Integer A
    String B$

    OpenCom #1
    Print #1, "Send"
    Input #1, A   'Get a numeric value from Port#1
    Input #1, B$  'Get a string from Port#1
    CloseCom #1
Fend
```
InputBox Statement

Displays a prompt in a dialog box, waits for the operator to input text or choose a button, and returns the contents of the box.

Syntax

InputBox prompt, title, default, data$

Parameters

prompt String expression displayed as a message in the dialog box.
title String expression displayed in the title bar of the dialog box.
default String expression displayed in the text box as the default response. If no default is desired, use an empty string ("").
data$ A string variable which will contain what the operator entered. If the operator clicks Cancel, this string will be “@”.

Description

InputBox displays the dialog and waits for the operator to click OK or Cancel. data is a string that contains what the operator typed in.

See Also

MsgBox

InputBox Statement Example

This function shows an InputBox example.

Function GetPartName$ As String
    String prompt$, title$, data$
    prompt$ = "Enter " + Chr$(34) + "part name" + Chr$(34) + ":"
    title$ = "Sample Application"
    InputBox prompt$, title$, "", data$
    If data$ <> "@" Then
        GetPartName$ = data$
    EndIf
Fend

The following picture shows the example output from the InputBox example code shown above.
InReal Function

Returns the input data of 2 words (32 bits) as the floating-point data (IEEE754 compliant) of 32 bits.

**Syntax**

\[
\text{InReal}(\text{WordPortNumber})
\]

**Parameters**

- **WordPortNumber** Integer expression representing the I/O Input Word.

**Return Values**

Returns the input port status in Real type number.

**Description**

From the input word port specified by the word port number, retrieve the 2 input word values as IEEE754 Real type value. Input word label can be used for the word port number parameter.

InReal Function cannot be used for the Wait command, or the condition of Till, Find, Sense.

**See Also**

In, InW, InBCD, Out, OutW, OpBCD, OutReal

**InW Function Example**

```cpp
Real realVal

realVal = InReal(32)
```
InsideBox Function

Returns the check status of the approach check area.

Syntax
InsideBox(AreaNum [, robotNumber | All])

Parameters
AreaNum  Integer expression from 1 to 15 representing which approach check area to return status for.
robotNumber  Integer value that contains the robot number you want to search. If omitted, the current robot will be specified. If you specify All, True is returned if one robot is in the check area.

Return Values
True if the robot end effector approaches the specified approach check area, otherwise False.

See Also
Box, BoxClr, BoxDef, GetRobotInsideBox, InsidePlane

Note
You can use the Wait statement with InsideBox to wait for the result of the InsideBox function in EPSON RC+ 5.0. However, you cannot use it in EPSON RC+ 6.0 and 7.0. In this case, use the GetRobotInsideBox function instead of the InsideBox function.

Correspondence table

<table>
<thead>
<tr>
<th>RC+ version</th>
<th>Robot Controller</th>
<th>Wait</th>
<th>Till, Find, Sense, Trap</th>
<th>Other commands (such as Print) / branch decision processing</th>
<th>Use of GetRobotInsideBox Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC+ 7.0</td>
<td>RC700</td>
<td>Not available</td>
<td>Not available</td>
<td>Available</td>
<td>All available</td>
</tr>
<tr>
<td>RC+ 7.0</td>
<td>RC90</td>
<td>Not available</td>
<td>Not available</td>
<td>Available</td>
<td>All available</td>
</tr>
<tr>
<td>RC+ 6.0</td>
<td>RC620</td>
<td>Not available</td>
<td>Not available</td>
<td>Available</td>
<td>All available</td>
</tr>
<tr>
<td>RC+ 5.0</td>
<td>RC90</td>
<td>Available</td>
<td>Not available</td>
<td>Available</td>
<td>Not available</td>
</tr>
</tbody>
</table>

Not available: Unavailable combination
Available: Available combination
All available: Available for Wait, Till, Find, Sense, Trap, Print, and branch decision processing.

InsideBox Function Example
The following program checks Robot 1 is in the check area (Box 3) or not.

```plaintext
Function PrintInsideBox
    If InsideBox(3,1) = True Then
        Print “Inside Box3”
    Else
        Print “Outside Box3”
    EndIf
EndFunction
```
InsidePlane Function

Returns the check status of the approach check plane.

Syntax

\[
\text{InsidePlane}(\text{PlaneNum [, robotNumber | All]})
\]

Parameters

- \text{PlaneNum} Integer expression from 1 to 15 representing which approach check plane to return status for.
- \text{robotNumber} Integer value that contains the robot number you want to search. If omitted, the current robot will be specified. If you specify All, True is returned if one robot is in the check area.

Return Values

True if the robot end effector approaches the specified approach check plane, otherwise False.

See Also

InsideBox, GetRobotInsidePlane, Plane, PlaneClr, PlaneDef

Note

You can use the Wait statement with InsidePlane to wait for the result of the InsidePlane function in EPSON RC+ 5.0. However, you cannot use it in EPSON RC+ 6.0 and 7.0. In this case, use the GetRobotInsidePlane function instead of the InsidePlane function.

<table>
<thead>
<tr>
<th>Correspondence table</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC+ version</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>RC+ 7.0</td>
</tr>
<tr>
<td>RC+ 7.0</td>
</tr>
<tr>
<td>RC+ 6.0</td>
</tr>
<tr>
<td>RC+ 5.0</td>
</tr>
</tbody>
</table>

Not available: Unavailable combination
Available: Available combination
All available: Available for Wait, Till, Find, Sense, Trap, Print, and branch decision processing.

InsidePlane Function Example

This is an example to check Robot 1 is in the check plane (Plane 3).

```plaintext
Function PrintInsidePlane
    If InsidePlane(3,1) = True Then
        Print “Inside Plane3”
    Else
        Print “Outside Plane3”
    EndIf
EndFend
```
InStr Function

Returns position of one string within another.

Syntax

\[ \text{InStr}(string, searchString) \]

Parameters

- \( string \) String expression to be searched.
- \( searchString \) String expression to be searched for within \( string \).

Return Values

Returns the position of the search string if the location is found, otherwise -1.

See Also

Mid$

Instr Function Example

```plaintext
Integer pos

pos = InStr("abc", "b")
```
Int Function

Converts a Real number to Integer. Returns the largest integer that is less than or equal to the specified value.

Syntax

\[ \text{Int}(\text{number}) \]

Parameters

\( \text{number} \) A real number expression.

Return Values

Returns an Integer value of the real number used in \( \text{number} \).

Description

\( \text{Int}(\text{number}) \) takes the value of \( \text{number} \) and returns the largest integer that is less than or equal to \( \text{number} \).

Note

For Values Less than 1 (Negative Numbers)

If the parameter \( \text{number} \) has a value of less than 1 then the return value have a larger absolute value than \( \text{number} \). (For example, if \( \text{number} = -1.35 \) then -2 will be returned.)

See Also

Abs, Atan, Atan2, Cos, Mod, Not, Sgn, Sin, Sqr, Str$, Tan, Val

Int Function Example

Some simple examples from the Command window are as follows:

```plaintext
> Print Int(5.1)
5
> Print Int(0.2)
0
> Print Int(-5.1)
-6
> 
```
Int32 Statement

 Declares variables of Int32 type. (4 byte integer type variable).

Syntax

```
Int32  varName [(subscripts)] [, varName [(subscripts)]]...
```

Parameters

- **varName**: Variable name which the user wants to declare.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows:

  (ubound1, [ubound2], [ubound3])

  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

<table>
<thead>
<tr>
<th>Type</th>
<th>Number of Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local variable</td>
<td>2,000</td>
</tr>
<tr>
<td>Global Preserve variable</td>
<td>4,000</td>
</tr>
<tr>
<td>Global variable and module variable</td>
<td>100,000</td>
</tr>
</tbody>
</table>

Description

Int32 is used to declare variables as type integer. Integer variables can contain values from -2147483648 to 2147483647. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

Boolean, Byte, Double, Global, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UInt64, UShort

Int32 Statement Example

The following example shows a simple program that declares some variables using Int32.

```plaintext
Function int32test
  Int32 A(10)         ' Single dimension array of Int32
  Int32 B(10, 10)     ' Two dimension array of Int32
  Int32 C(5, 5, 5)    ' Three dimension array of Int32
  Int32 var1, arrayvar(10)
  Integer i
  Print "Please enter an Integer Number"
  Input var1
  Print "The Integer variable var1 = ", var1
  For i = 1 To 5
    Print "Please enter an Integer Number"
    Input arrayvar(i)
    Print "Value Entered was ", arrayvar(i)
  Next i
End
```

Int64 Statement

Declares variables of Int64 type. (8 byte integer type variable).

Syntax

\[\text{Int64 } \text{varName} \{[(\text{subscripts})]\} [, \text{varName} \{[(\text{subscripts})]\}]...\]

Parameters

- \textit{varName} Variable name which the user wants to declare.
- \textit{subscripts} Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows

\[(\text{ubound1}, [\text{ubound2}], [\text{ubound3}])\]

\text{ubound1}, \text{ubound2}, \text{ubound3} each specify the maximum upper bound for the associated dimension.

The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

When specifying the upper bound value, make sure the number of total elements is within the range shown below:

- Local variable 2,000
- Global Preserve variable 4,000
- Global variable and module variable 100,000

Description

Int64 is used to declare variables as type integer. Integer variables can contain values from -922372036854775808 to 922372036854775807. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

- Boolean, Byte, Double, Global, Int32, Integer, Long, Real, Short, String, UByte, UInt32, UShort, UInt64

Int64 Statement Example

The following example shows a simple program that declares some variables using Int64.

```plaintext
Function int64test
    Int64 A(10)   'Single dimension array of Int64
    Int64 B(10, 10)   'Two dimension array of Int64
    Int64 C(5, 5, 5)   'Three dimension array of Int64
    Int64 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
End
```


Integer Statement

Declares variables of Integer type. (2 byte integer type variable).

Syntax

```
Integer varName [(subscripts)] [, varName [(subscripts)]]...
```

Parameters

- **varName**: Variable name which the user wants to declare as type integer.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows
  
  `ubound1, [ubound2], [ubound3]`

  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

  - Local variable: 2,000
  - Global Preserve variable: 4,000
  - Global variable and module variable: 100,000

Description

Integer is used to declare variables as type integer. Variables of type integer can contain whole numbers with values from -32768 to 32767. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

Boolean, Byte, Double, Global, Int32, Int64, Long, Real, Short, String, UByte, UInt32, UInt64, UShort

Integer Statement Example

The following example shows a simple program that declares some variables using Integer.

```plaintext
Function inttest
  Integer A(10)         'Single dimension array of integer
  Integer B(10, 10)     'Two dimension array of integer
  Integer C(5, 5, 5)    'Three dimension array of integer
  Integer var1, arrayvar(10)
  Integer i
  Print "Please enter an Integer Number"
  Input var1
  Print "The Integer variable var1 = ", var1
  For i = 1 To 5
    Print "Please enter an Integer Number"
    Input arrayvar(i)
    Print "Value Entered was ", arrayvar(i)
  Next i
End
```

366 EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
InW Function

Returns the status of the specified input word port. Each word port contains 16 input bits.

Syntax

\[
\text{InW(WordPortNum)}
\]

Parameters

WordPortNum Integer expression representing the I/O Input Word.

Return Values

Returns the current status of inputs (long integers from 0 to 65535).

Note

Rule of word port which contains the input bit of Real Time I/O

Word ports =1, 3, 17, 19 return the state of the input port with an integer from 0 to 255. The input bit of the Real Time I/O is not reflected.

See Also

In, Out, OutW

InW Function Example

\[
\begin{align*}
\text{Long word0} \\
\text{word0} &= \text{InW}(0)
\end{align*}
\]
IODef Function

Returns whether the specified input or output bit, byte, word, or I/O label are defined.

Syntax

- IODEf (IOType, IOWidth, portNumber)
- IODEf (IOlabel)

Parameters

- **IOType**: Integer expression representing the type of I/O.
  - 0 - Input
  - 1 - Output
  - 2 - Memory
- **IOWidth**: Integer expression representing the width of the port: 1(bit), 8 (byte), or 16 (word).
- **portNumber**: Integer expression representing the bit, byte, or word port number to return the label for.
- **IOlabel**: String expression that specifies the standard I/O or memory I/O label.

Return Values

True if the specified input or output bit, byte, word or the I/O label are defined, otherwise False.

See Also

- IOLabel$, IONumber

IODef Function Example

```plaintext
Integer i

For i = 0 To 15
    If IODEf( 0, i, i) = TRUE Then
        Print “Port " , i, “ is defined"
    Else
        Print “Port " , i, “ is undefined"
    EndIf
Next i
```
**IOLabel$ Function**

Returns the I/O label for a specified input or output bit, byte, or word.

**Syntax**

\[ \text{IOLabel$}((\text{IOType}, \text{IOWidth}, \text{portNumber})) \]

**Parameters**

- **IOType**: Integer expression representing the type of I/O.
  - 0 - Input
  - 1 - Output
  - 2 - Memory

- **IOWidth**: Integer expression representing the width of the port: 1 (bit), 8 (byte), or 16 (word).

- **portNumber**: Integer expression representing the bit, byte, or word port number to return the label for.

**Return Values**

String containing the label.

**See Also**

- PLabel$, IONumber, IODef

**IOLabel$ Function Example**

```plaintext
Integer i

For i = 0 To 15
    Print "Input ", i, ": ", IOLabel$(0, 1, i)
Next i
```
IONumber Function

Returns the I/O number of the specified I/O label.

Syntax

```
IONumber(IOlabel)
```

Parameters

- `IOlabel` String expression that specifies the standard I/O or memory I/O label.

Return Values

Returns the I/O port number (bit, byte, word) of the specified I/O label. If there is no such I/O label, an error will be generated.

See Also

- `IOLabel$`, `IODef`

IONumber Function Example

```plaintext
Integer IObit

IObit = IONumber("myIO")

IObit = IONumber("Station" + Str$(station) + "InCycle")
```
J1Angle Statement

Sets the J1Angle attribute of a point.

Syntax

(1) J1Angle point [, Step ]
(2) J1Angle

Parameters

point P number or P(expr) or point label.
Step Optional. Real value that specifies the set value.

Result

The J1Angle attribute can be used for the RS and N robot series.
If Step is omitted, the J1Angle value for the specified point will be displayed.
If both parameters are omitted, the J1Angle value of the current robot position will be displayed.

RS series: Specify the angle of the Joint #1 when both X and Y coordinate values of a point are “0” (singularity). For other robot series points, J1Angle has no meaning.

N series: Specify the angle of the Joint #1 when the axis centers of “Joint #1, #4, and #6”, “Joint #1 and #6”, or “Joint #1 and #4” are on the straight line. For other robot series points (not singularity), J1Angle has no meaning.

See Also

Hand, J1Angle Function, J1Flag, J2Flag, J4Angle, J4Angle Function

J1Angle Statement Example

J1Angle P0, 10.0
J1Angle P(mypoint), 0.0
J1Angle Function

Returns the J1Angle attribute of a point.

Syntax

\[
\text{J1Angle } ([ \text{ point }] )
\]

Parameters

point \hspace{1cm} \text{Point expression}

Optional. If omitted, returns the J1Angle setting of the current robot position.

Return Values

The J1Angle attribute can be used for the RS and N robot series.

Returns the angle of Joint 1 when both X and Y coordinate values of a point are “0” (singularity) in a real value. The J1Angle attribute can be used for the RS series.

RS series: Returns an integer value representing the angle of the Joint #1 when both X and Y coordinate values of a point are “0” (singularity).

N series: Returns an integer value representing the angle of the Joint #1 when the axis centers of “Joint #1, #4, and #6”, “Joint #1 and #6”, or “Joint #1 and #4” are on the straight line.

See Also

Hand, J1Angle, J1Flag, J2Flag, J4Angle, J4Angle Function

J1Angle Function Example

```
Print J1Angle(pick)
Print J1Angle(P1)
Print J1Angle
```
J1Flag Statement

Specifies the J1Flag attribute of a point.

Syntax
(1) J1Flag point [ , value]
(2) J1Flag

Parameters
point    Pnumber or P(expr) or point label.
value   Optional. Integer expression.
         For RS series Manipulator:
         0 (/J1F0) J1 range is -90 to +270 degrees
         1 (/J1F1) J1 range is from -270 to -90 or +270 to +450 degrees
         For C8, C12 series Manipulator:
         0 (/J1F0) J1 range is 0 to -180 or 0 to +180 degrees
         1 (/J1F1) J1 range is -180 to -240 or +180 to -240 degrees

Return Values
The J1Flag attribute specifies the range of values for joint 1 for one point. If value is omitted, the J1Flag value for the specified point is displayed. When both parameters are omitted, the J1Flag value is displayed for the current robot position.

See Also
Hand, J1Flag Function, J2Flag

J1Flag Statement Example

J1Flag P0, 1
J1Flag P(mypoint), 0
J1Flag Function

Returns the J1Flag attribute of a point.

Syntax

\[
\text{J1Flag ([point])}
\]

Parameters

\(\text{point}\)  Optional. Point expression. If \(\text{point}\) is omitted, then the J1Flag setting of the current robot position is returned.

Return Values

\[
\begin{align*}
0 & /J1F0 \\
1 & /J1F1 \\
\end{align*}
\]

See Also

Hand, J1Flag Statement, J2Flag

J1Flag Function Example

\[
\begin{align*}
\text{Print J1Flag(pick)} \\
\text{Print J1Flag(P1)} \\
\text{Print J1Flag} \\
\text{Print J1Flag(Pallet(1, 1))}
\end{align*}
\]
J2Flag Statement

Sets the J2Flag attribute of a point.

Syntax

(1) J2Flag point [, value ]
(2) J2Flag

Parameters

point Pnumber or P(expr) or point label.
value Optional. Integer expression.
0 (/J2F0) J2 range is -180 to +180 degrees
1 (/J2F1) J2 range is from -360 to -180 or +180 to +360 degrees

Return Values

The J2Flag attribute specifies the range of values for joint 2 for one point. If value is omitted, the J2Flag value for the specified point is displayed. When both parameters are omitted, the J2Flag value is displayed for the current robot position.

See Also

Hand, J1Flag, J2Flag Function

J2Flag Statement Example

J2Flag P0, 1
J2Flag P(mypoint), 0
J2Flag Function

Returns the J2Flag attribute of a point.

Syntax

```
J2Flag [(point)]
```

Parameters

- **point**: Optional. Point expression. If `point` is omitted, then the J2Flag setting of the current robot position is returned.

Return Values

- 0 /J2F0
- 1 /J2F1

See Also

- Hand, J1Flag, J2Flag Statement

J2Flag Function Example

```
Print J2Flag(pick)
Print J2Flag(P1)
Print J2Flag
Print J2Flag(P1 + P2)
```
J4Angle Statement

Sets the J4Angle attribute of a point.

Syntax

(1) J4Angle point [, value ]
(2) J4Angle

Parameters

point                  Pnumber or P(expr) or point label.
value                 Optional. Integer expression.

Result

The J4Angle attribute is used only for N robot series.
It specifies the angle of the Joint #4 when the axis centers of the Joint #4 and #6 are on the straight line.
If the point is not singularity, J4Angle has no meaning.
If value is omitted, the J4Angle value for the specified point is displayed. When both parameters are omitted,
the J4Angle value is displayed for the current robot position.

See Also

Hand, J1Angle, J1Angle Function, J4Angle Function

Note

When both J4Flag and J4Angle are used, J4Angle is prioritized as follows:

J4Angle P0,0
J4Flag P0,1

J4Angle Example

    J4Angle P0, 10.0
    J4Angle P(mypoint), 0.0
J4Angle Function

Returns the J4Angle attribute of a point.

Syntax

\[
\text{J4Angle } ([\text{point}])
\]

Parameters

\(\text{point}\) Optional. Point expression. If \text{point} is omitted, then the J4Angle setting of the current robot position is returned.

Return Values

Returns an integer value representing the angle of the Joint #4 when the axis centers of the Joint #4 and #6 are on the straight line.

The J4Angle attribute is used only for N robot series.

See Also

Hand, J1Angle, J1Angle Function, J4Angle

J4Angle Function Example

Print \texttt{J4Angle(pick)}
Print \texttt{J4Angle(P1)}
Print \texttt{J4Angle}
J4Flag Statement

Sets the J4Flag attribute of a point.

Syntax

(1) J4Flag point [, value ]
(2) J4Flag

Parameters

point

Pnumber or P(expr) or point label.

value

Optional. Integer expression.

0 (/J4F0) J4 range is -180 to +180 degrees

1 (/J4F1) J4 range is from -360 to -180 or +180 to +360 degrees

Return Values

The J4Flag attribute specifies the range of values for joint 4 for one point. If value is omitted, the J4Flag value for the specified point is displayed. When both parameters are omitted, the J4Flag value is displayed for the current robot position.

See Also

Elbow, Hand, J4Flag Function, J6Flag, Wrist

J4Flag Statement Example

J4Flag P0, 1
J4Flag P(mypoint), 0
J4Flag Function

Returns the J4Flag attribute of a point.

Syntax

\[
\text{J4Flag } [\text{point}]
\]

Parameters

point  Optional. Point expression. If point is omitted, then the J4Flag setting of the current robot position is returned.

Return Values

0  /J4F0
1  /J4F1

See Also

Elbow, Hand, Wrist, J4Flag Statement, J6Flag

J4Flag Function Example

Print \texttt{J4Flag}(pick)
Print \texttt{J4Flag}(P1)
Print \texttt{J4Flag}
Print \texttt{J4Flag}(Pallet(1, 1))
J6Flag Statement

Sets the J6Flag attribute of a point.

Syntax
(1) J6Flag point [, value]
(2) J6Flag

Parameters
point Pnumber or P(expr) or point label.
value Integer expression. Range is 0 - 127 (/J6F0 to /J6F127). J6 range for the specified point is as follows:
( -180 * (value+1) < J6 <= 180 * value ) and ( 180 * value < J6 <= 180 * (value+1) )

Return Values
The J6Flag attribute specifies the range of values for joint 6 for one point. If value is omitted, the J6Flag value for the specified point is displayed. When both parameters are omitted, the J6Flag value is displayed for the current robot position.

See Also
Elbow, Hand, J4Flag, J6Flag Function, Wrist

Note
Range of J6Flag differs depending on manipulator models
C4 : 0 - 127 (/J6F0 to /J6F127)
C8, C12 : 0 - 81 (/J6F0 to /J6F81)
N2 : 0 - 40 (/J6F0 to /J6F40)
N6 : 0 - 61 (/J6F0 to /J6F61)

J6Flag Statement Example

J6Flag P0, 1
J6Flag P(mypoint), 0
J6Flag Function

Returns the J6Flag attribute of a point.

Syntax

\[
\text{J6Flag}\left[\text{point}\right]
\]

Parameters

\text{point} \quad \text{Optional. Point expression. If } \text{point} \text{ is omitted, then the J6Flag setting of the current robot position is returned.}

Return Values

0 to 127 /J6F0 to /J6F127

See Also

Elbow, Hand, Wrist, J4Flag, J6Flag

J6Flag Function Example

\begin{verbatim}
Print J6Flag(pick)
Print J6Flag(P1)
Print J6Flag
Print J6Flag(P1 + P2)
\end{verbatim}
Returns a robot point specified in joint angles.

**Syntax**

\[ \text{JA} (j1, j2, j3, j4, j5, j6, j7, j8, j9) \]

**Parameters**

- **j1 – j9**: Real expressions representing joint angles.
  - For linear joints, specifies in units of mm.
  - j5 and j6 are for the 6-axis robot (including N series) and Joint type 6-axis robot.
  - j7 is for the Joint type 7-axis robot.
  - j8 and j9 are for the additional ST axis.

**Note**

If the angle exceeding the motion range is specified, an out of range error occurs.

**Return Values**

A robot point whose location is determined by the specified joint angles.

**Description**

Use JA to specify a robot point using joint angles.

When the points returned from JA function specify a singularity of the robot, the joint angles of the robot do not always agree with the joint angles supplied to the JA function as arguments during the execution of a motion command for the points. To operate the robot using the joint angles specified for the JA function, avoid a singularity of the robot.

For example:

```plaintext
> go ja(0,0,0,90,0,-90)
> where
WORLD:  X:  0.000 mm  Y: 655.000 mm  Z: 675.000 mm  U: 0.000 deg
V:  -90.000 deg  W:  -90.000 deg
JOINT:  1:  0.000 deg  2:  0.000 deg  3:  0.000 deg  4: 0.000 deg
  5:  0.000 deg  6:  0.000 deg
PULSE:  1:  0 pls  2: 0 pls  3: 0 pls  4:  0 pls
  5:  0 pls  6: 0 pls
> go ja(0,0,0,90,0.001,-90)
> where
WORLD:  X:  -0.004 mm  Y: 655.000 mm  Z: 675.000 mm  U: 0.000 deg
V:  -90.000 deg  W:  -89.999 deg
JOINT:  1:  0.000 deg  2:  0.000 deg  3:  0.000 deg  4:  90.000 deg
  5:  0.001 deg  6:  -90.000 deg
PULSE:  1:  0 pls  2: 0 pls  3: 0 pls  4: 2621440 pls
  5:  29 pls  6: -1638400 pls
```

**See Also**

AglToPls, XY

**JA Function Example**

```plaintext
P10 = JA(60, 30, -50, 45)
Go JA(135, 90, -50, 90)
P3 = JA(0, 0, 0, 0, 0, 0)
```
Joint Statement

Displays the current position for the robot in joint coordinates.

Syntax
Joint

See Also
Pulse, Where

Joint Statement Example

```
> joint
  JOINT:  1: -6.905 deg 2:  23.437 deg 3:  -1.999 mm  4: -16.529 deg
>
```
**JRange Statement**

Defines the permissible working range of the specified joint in pulses.

**Syntax**

\[ \text{JRange \ jointNumber, lowerLimit, upperLimit} \]

**Parameters**

- **jointNumber**: Integer expression from 1 to 9 representing the joint for which JRange will be specified. The additional S axis is 8 and T axis is 9.
- **lowerLimit**: Long integer expression representing the encoder pulse count position for the lower limit range of the specified joint.
- **upperLimit**: Long integer expression representing the encoder pulse count position for the upper limit range of the specified joint.

**Description**

Defines the permissible working range for the specified joint with upper and lower limits in encoder pulse counts. JRange is similar to the Range command. However, the Range command requires that all joint range limits be set while the JRange command can be used to set each joint working limits individually thus reducing the number of parameters required. To confirm the defined working range, use the Range command.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

**Notes**

- **Lower Limits Must Not Exceed Upper Limits**: The Lower limit defined in the JRange command must not exceed the Upper limit. A lower limit in excess of the Upper limit will cause an error, making it impossible to execute a motion command.

- **Factors Which can Change JRange**: Once JRange values are set they remain in place until the user modifies the values either by the Range or JRange commands. Turning controller power off will not change the JRange joint limit values.

- **Maximum and Minimum Working Ranges**: Refer to the specifications in the Robot manual for maximum working ranges for each robot model since these vary from model to model.

**See Also**

- Range, JRange Function

**JRange Statement Example**

The following examples are done from the Command window:

- `> JRange 2, -6000, 7000'` *Define the 2nd joint range*
- `> JRange 1, 0, 7000` *Define the 1st joint range*
JRange Function

Returns the permissible working range of the specified joint in pulses.

Syntax

\[ \text{JRange}(\text{jointNumber}, \text{paramNumber}) \]

Parameters

- **jointNumber**: Specifies reference joint number (integer from 1 to 9) by an expression or numeric value.
  - The additional S axis is 8 and T axis is 9.

- **paramNumber**: Integer expression containing one of two values:
  - 1: Specifies lower limit value.
  - 2: Specifies upper limit value.

Return Values

Range configuration (integer value, pulses) of the specified joint.

See Also

Range, JRange Statement

JRange Function Example

```plaintext
Long i, oldRanges(3, 1)

For i = 0 To 3
    oldRanges(i, 0) = JRange(i + 1, 1)
    oldRanges(i, 1) = JRange(i + 1, 2)
Next i
```
JS Function

Jump Sense detects whether the arm stopped prior to completing a Jump, Jump3, JumpTLZ, or Jump3CP instruction which used a Sense input or if the arm completed the move.

Syntax
JS

Return Values
Returns a True or a False.
- True: When the arm was stopped prior to reaching its target destination because a Sense Input condition was met JS returns a True.
- False: When the arm completes the normal move and reaches the target destination as defined in the Jump instruction JS returns a False.

Description
JS is used in conjunction with the Jump and Sense instructions. The purpose of the JS instruction is to provide a status result as to whether an input condition (as defined by the Sense instruction) is met during motion caused by the Jump instruction or not. When the input condition is met, JS returns a True. When the input condition is not met and the arm reaches the target position, JS returns a False.

JS is simply a status check instruction and does not cause motion or specify which Input to check during motion. The Jump instruction is used to initiate motion and the Sense instruction is used to specify which Input (if any) to check during Jump initiated motion.

Note
JS Works only with the Most Recent Jump, Jump3, JumpTLZ, Jump3CP Instruction:
JS can only be used to check the most recent Jump instruction's input check (which is initiated by the Sense instruction.) Once a 2nd Jump instruction is initiated, the JS instruction can only return the status for the 2nd Jump instruction. The JS status for the first Jump is gone forever. So be sure to always do any JS status check for Jump instructions immediately following the Jump instruction to be checked.

See Also
JT, Jump, Jump3, Jump3CP, JumpTLZ, Sense

JS Function Example

Function SearchSensor As Boolean
    Sense Sw(5) = On
    Jump P0
    Jump P1 Sense
    If JS = TRUE Then
        Print "Sensor was found"
        SearchSensor = TRUE
    EndIf
EndFunction
JT Function

Returns the status of the most recent Jump, Jump3, JumpTLZ, or Jump3CP instruction for the current robot.

Syntax

JT

Return Values

JT returns a long with the following bits set or clear:

| Bit 0 | Set to 1 when rising motion has started or rising distance is 0. |
| Bit 1 | Set to 1 when horizontal motion has started or horizontal distance is 0. |
| Bit 2 | Set to 1 when descent motion has started or descent distance is 0. |
| Bit 16 | Set to 1 when rising motion has completed or rising distance is 0. |
| Bit 17 | Set to 1 when horizontal motion has completed or horizontal distance is 0. |
| Bit 18 | Set to 1 when descent motion has completed or descent distance is 0. |

Description

Use JT to determine the status of the most recent Jump command that was stopped before completion by Sense, Till, abort, etc.

See Also

JS, Jump, Jump3, Jump3CP, JumpTLZ, Sense, Till

JT Function Example

Function SearchTill As Boolean

    Till Sw(5) = On

    Jump P0
    Jump P1 Till
    If JT And 4 Then
        Print "Motion stopped during descent"
        SearchTill = TRUE
    EndIf
EndFunction
Perform a relative move of one joint.

Syntax

\texttt{JTran jointNumber, distance}

Parameters

- \textit{jointNumber} \hspace{1cm} Integer expression representing which joint to move. The additional S axis is 8 and T axis is 9.
- \textit{distance} \hspace{1cm} Real expression representing the distance to move in degrees for rotational joints or millimeters for linear joints.

Description

Use JTran to move one joint a specified distance from the current position.

See Also

Go, Jump, Move, Ptran

JTran Statement Example

\texttt{JTran 1, 20}
Jump Statement

Moves the arm from the current position to the specified destination point using point to point motion by first moving in a vertical direction up, then horizontally and then finally vertically downward to arrive on the final destination point.

Syntax


Parameters

destination The target destination of the motion using a point expression.
archNumber Optional. The arch number (archNumber) specifies which Arch Table entry to use for the Arch type motion caused by the Jump instruction. archNumber must always be proceeded by the letter C. (Valid entries are from C0 to C7.)
zLimit Optional. This is a Z limit value which represents the maximum position the Z joint will travel to during the Jump motion. This can be thought of as the Z Height Ceiling for the Jump instruction. Any valid Z joint Coordinate value is acceptable.
PerformMode Optional. Specify the robot performance mode.
modeNumber Specify the operation mode assigned to PerformMode with an integer value (1 to 3) or with the following constant. If PerformMode is specified, this parameter cannot be omitted.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode_Standard</td>
<td>1</td>
<td>Sets the Standard mode</td>
</tr>
<tr>
<td>Mode_High_Speed</td>
<td>2</td>
<td>Sets the High-speed mode</td>
</tr>
<tr>
<td>Mode_Low_Oscillation</td>
<td>3</td>
<td>Sets the Low-oscillation mode</td>
</tr>
</tbody>
</table>

CP Optional. Specifies continuous path motion.

searchExpr Optional. A Sense, Till or Find expression.

Sense | Till | Find

Sense Sw(expr) = {On | Off}

Till Sw(expr) = {On | Off}

Find Sw(expr) = {On | Off}

!...! Optional. Parallel Processing statements can be added to the Jump instruction to cause I/O and other commands to execute during motion.

SYNC Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Jump moves the arm from the current position to destination using what is called Arch Motion. Jump can be thought of as 3 motions in 1. For example, when the Arch table entry defined by archNumber is 7, the following 3 motions will occur.

1) The move begins with only Z-joint motion until it reaches the Z joint height calculated by the Arch number used for the Jump command.

2) Next the arm moves horizontally (while still moving upward in Z) towards the target point position until the upper Z Limit (defined by LimZ ) is reached. Then the arm begins to move downward in the Z direction (while continuing X, Y and U joint motion) until the final X, and Y and U joint positions are reached.

3) The Jump instruction is then completed by moving the arm down with only Z-joint motion until the target Z-joint position is reached.
The coordinates of destination (the target position for the move) must be taught previously before executing the Jump instruction. The coordinates cannot be specified in the Jump instruction itself. Acceleration and deceleration for the Jump is controlled by the Accel instruction. Speed for the move is controlled by the Speed instruction.

The Jump instruction cannot be executed for the vertical 6-axis robots (including N series). Use the Jump3 instruction.

**CP Details**
The CP parameter causes acceleration of the next motion command to start when the deceleration starts for the current motion command. In this case the robot will not stop at the destination coordinate and will continue to move to the next point.

**archNumber Details**
The Arch for the Jump instruction can be modified based on the archNumber value optionally specified with the Jump instruction. This allows the user to define how much Z to move before beginning the X, Y, and U joint motion. (This allows the user to move the arm up and out of the way of parts, feeders and other objects before beginning horizontal motion.) Valid archNumber entries for the Jump instruction are between C0 and C7. The Arch table entries for C0 to C6 are user definable with the Arch instruction. However, C7 is a special Arch entry which always defines what is called Gate Motion. Gate Motion means that the robot first moves Z all the way to the coordinate defined by LimZ before beginning any X, Y, or U joint motion. Once the LimZ Z limit is reached, X, Y and U joint motion begins. After the X, Y, and U joints each reaches its final destination position, then the Z joint can begin moving downward towards the final Z joint coordinate position as defined by destination (the target point). Gate Motion looks as follows:

![Gate Motion Diagram](image)

**LimZ Details**
LimZ zLimit specifies the upper Z coordinate value for the horizontal movement plane in the current local coordinate system. The specified arch settings can cause the X, Y, and U joints to begin movement before reaching LimZ, but LimZ is always the maximum Z height for the move. When the LimZ optional parameter is omitted, the previous value specified by the LimZ instruction is used for the horizontal movement plane definition.

It is important to note that the LimZ zLimit height limit specification is the Z value for the local robot coordinate system. It is not the Z value for Arm or Tool. Therefore take the necessary precautions when using tools or hands with different operating heights.

**Sense Details**
The Sense optional parameter allows the user to check for an input condition or memory I/O condition before beginning the final Z motion downward. If satisfied, this command completes with the robot stopped above the target position where only Z motion is required to reach the target position. It is important to note that the robot arm does not stop immediately upon sensing the Sense input modifier.
Jump Statement

The JS or Stat commands can then be used to verify whether the Sense condition was satisfied and the robot stopped prior to its target position or that the Sense condition was not satisfied and the robot continued until stopping at its target position.

Till Details
The optional Till qualifier allows the user to specify a condition to cause the robot to decelerate to a stop prior to completing the Jump. The condition specified is simply a check against one of the I/O inputs or one of the memory I/O. This is accomplished through using either the Sw or MemSw function. The user can check if the input is On or Off and cause the arm to decelerate and stop based on the condition specified.

The Stat function can be used to verify whether the Till condition has been satisfied and this command has been completed, or the Till condition has not been satisfied and the robot stopped at the target position.

Deceleration motion and acceleration motion of different modes can be combined when PerformMode is set while the path motion is enabled. Some combinations are not available depending on operation modes. For details, refer to PerformMode Statement.

Notes

Jump cannot be executed for 6-axis robots (including N series)
Use Jump3 or Jump3CP for 6-axis robots.

Omitting archNumber Parameters
If the archNumber optional parameter is omitted, the default Arch entry for use with the Jump instruction is C7. This will cause Gate Motion, as described above.

Difference between Jump and Jump3, Jump3CP
The Jump3 and Jump3CP instructions can be used for 6-axis robots (including N series). On the other hand the Jump instruction cannot be used for 6-axis robots. For SCARA robots (including RS series), using the Jump instruction shortens the joint motion time for depart and approach motion. The depart and approach motions in Jump3 can be executed along the Z axis and in other directions.

Difference between Jump and Go
The Go instruction is similar to Jump in that they both cause Point to Point type motion, however there are many differences. The most important difference is that the Go instruction simply causes Point to Point motion where all joints start and stop at the same time (they are synchronized). Jump is different since it causes vertical Z movement at the beginning and end of the move. Jump is ideal for pick and place type applications.

Decelerating to a Stop With the Jump Instruction
The Jump instruction always causes the arm to decelerate to a stop prior to reaching the destination point.

Proper Speed and Acceleration Instructions with Jump:
The Speed and Accel instructions are used to specify the speed and acceleration of the robot during Jump motion. Pay close attention to the fact that Speed and Accel apply to point to point type motion (Go, Jump, Etc.). while linear and circular interpolated motion instructions such as Move or Arc use the SpeedS and AccelS instructions. For the Jump instruction, it is possible to separately specify speeds and accelerations for Z joint upward motion, horizontal travel including U joint rotation, and Z joint downward motion.
Pass function of Jump
When the CP parameter is specified for Jump with 0 downward motion, the Jump horizontal travel does not decelerate to a stop but goes on smoothly to the next PTP motion.
When the CP parameter is specified for a PTP motion command right before a Jump with 0 upward motion, the PTP motion does not decelerate to a stop but connects smoothly with the Jump horizontal travel.
This is useful when you want to replace the horizontal travel of Jump (a PTP motion) with several PTP motions.
(Example)

Go P1
Jump P2 :Z(-50) C0 LimZ -50 CP
Go P3 :Z(0) CP
Jump P4 C0 LimZ 0

Caution for Arch motion
Jump motion trajectory is comprised of vertical motion and horizontal motion. It is not a continuous path trajectory. The actual Jump trajectory of arch motion is not determined by Arch parameters alone. It also depends on motion and speed.

Always use care when optimizing Jump trajectory in your applications. Execute Jump with the desired motion and speed to verify the actual trajectory.

When speed is lower, the trajectory will be lower. If Jump is executed with high speed to verify an arch motion trajectory, the end effector may crash into an obstacle with lower speed.

In a Jump trajectory, the depart distance increases and the approach distance decreases when the motion speed is set high. When the fall distance of the trajectory is shorter than the expected, lower the speed and/or the deceleration, or change the fall distance to be larger.

Even if Jump commands with the same distance and speed are executed, the trajectory is affected by motion of the robot arms. As a general example, for a SCARA robot the vertical upward distance increases and the vertical downward distance decreases when the movement of the first arm is large. When the vertical fall distance decreases and the trajectory is shorter than the expected, lower the speed and/or the deceleration, or change the fall distance to be larger.

Potential Errors
LimZ Value Not High Enough
When the current arm position of the Z joint is higher than the value set for LimZ and a Jump instruction is attempted, an Error 4005 will occur.

See Also
Accel, Arc, Arch, Go, JS, JT, LimZ, P#= (Point Assignment), PerformMode, Pulse, Sense, Speed, Stat, Till
Jump Statement Example

The example shown below shows a simple point to point move between points P0 and P1 and then moves back to P0 using the Jump instruction. Later in the program the arm moves using the Jump instruction. If input #4 never goes high then the arm starts the approach motion and moves to P1. If input #4 goes high then the arm does not execute the approach motion.

Function jumptest
Home
Go P0
Go P1
Sense Sw(4) = On
Jump P0 LimZ -10
Jump P1 LimZ -10 Sense 'Check input #4
If Js(0) = 1 Then
  Print "Input #4 came on during the move and"
  Print "the robot stopped prior to arriving on"
  Print "point P1."
Else
  Print "The move to P1 completed successfully."
  Print "Input #4 never came on during the move."
EndIf
Fend

> Jump P10+X50 C0 LimZ-20 Sense !D50;On 0;D80;On 1!
Jump3, Jump3CP Statements

3D gate motion.
Jump3 is a combination of two CP motions and one PTP motion.
Jump3CP is a combination of three CP motions.

Syntax

(1) \textbf{Jump3} \hspace{0.5cm} \textit{depart, approach, destination [, CarchNumber] [, CP] [, LJM [, orientationFlag]] [, searchExpr] [, !...!] [, SYNC]}
(2) \textbf{Jump3CP} \hspace{0.5cm} \textit{depart, approach, destination [, ROT] [, CarchNumber] [, CP] [, LJM [, orientationFlag]] [, searchExpr] [, !...!] [, SYNC]}

Parameters

\textit{depart} \hspace{0.5cm} The departure point above the current position using a point expression.
\textit{approach} \hspace{0.5cm} The approach point above the destination position a point expression.
\textit{destination} \hspace{0.5cm} The target destination of the motion using a point expression.
\textit{ROT} \hspace{0.5cm} Optional. :Decides the speed/acceleration/deceleration in favor of tool rotation.
\textit{archNumber} \hspace{0.5cm} Optional. The arch number (\textit{archNumber}) specifies which Arch Table entry to use for the Arch type motion caused by the Jump instruction. \textit{archNumber} must always be proceeded by the letter C. (Valid entries are C0 to C7.)
\textit{CP} \hspace{0.5cm} Optional. Specifies continuous path motion.
\textit{LJM} \hspace{0.5cm} Optional. Convert the target destination using LJM function.
\textit{orientationFlag} \hspace{0.5cm} Optional. Specifies a parameter that selects an orientation flag for LJM function.
\textit{searchExpr} \hspace{0.5cm} Optional. A Sense, Till or Find expression.
\textit{Sense} | \textit{Till} | \textit{Find}
\textit{Sense Sw(expr) = (On | Off)}
\textit{Till Sw(expr) = (On | Off)}
\textit{Find Sw(expr) = (On | Off)}
\textit{!...!} \hspace{0.5cm} Optional. Parallel Processing statements can be added to the Jump instruction to cause I/O and other commands to execute during motion.
\textit{SYNC} \hspace{0.5cm} Reserves a motion command. The robot will not move until \textit{SyncRobots} is executed.

Description

Moves the arm from the current position to the destination point with 3D gate motion. 3D gate motion consists of depart motion, span motion, and approach motion. The depart motion form the current position to the depart point is always CP motion. The span motion from the depart point to the start approach point is PTP motion in Jump3, and the CP motion in Jump3CP.
The approach motion from the starting approach point to the target point is always CP motion.
Arch motion is achieved by specifying the arch number. The arch motion for Jump3, Jump3CP is as shown in the figure below. For arch motion to occur, the Depart distance must be greater than the arch upward distance and the Approach distance must be greater than the arch downward distance.

Jump3CP uses the SpeedS speed value and AccelS acceleration and deceleration values. Refer to **Using Jump3CP with CP** below on the relation between the speed/acceleration and the acceleration/deceleration. If, however, the ROT modifier parameter is used, Jump3CP uses the SpeedR speed value and AccelR acceleration and deceleration values. In this case SpeedS speed value and AccelS acceleration and deceleration value have no effect.

Usually, when the move distance is 0 and only the tool orientation is changed, an error will occur. However, by using the ROT parameter and giving priority to the acceleration and the deceleration of the tool rotation, it is possible to move without an error. When there is not an orientational change with the ROT modifier parameter and movement distance is not 0, an error will occur.

Also, when the tool rotation is large as compared to move distance, and when the rotation speed exceeds the specified speed of the manipulator, an error will occur. In this case, please reduce the speed or append the ROT modifier parameter to give priority to the rotational speed/acceleration/deceleration.

**Notes**

**LimZ does not affect Jump3 and Jump3CP**

LimZ has no effect on Jump3 or Jump3CP since the span motion is not necessarily perpendicular to the Z axis of the coordinate system.

**Jump3 span motion is PTP (point to point)**

It is difficult to predict Jump3 span motion trajectory. Therefore, be careful that the robot doesn't collide with peripheral equipment and that robot arms don’t collide with the robot.

**Using Jump3, Jump3CP with CP**

The CP parameter causes the arm to move to destination without decelerating or stopping at the point defined by destination. This is done to allow the user to string a series of motion instructions together to cause the arm to move along a continuous path while maintaining a specified speed throughout all the motion. The Jump3 and Jump3CP instructions without CP always cause the arm to decelerate to a stop prior to reaching the point destination.

**Pass function of Jump3**

When the CP parameter is specified for Jump3 with 0 approach motion, the Jump3 span motion does not decelerate to a stop but goes on smoothly to the next PTP motion. When the CP parameter is specified for a PTP motion command right before Jump3 with 0 depart motion, the PTP motion does not decelerate to a stop but connects smoothly with the Jump3 span motion. This is useful when you want to replace the span motion of Jump3 (a PTP motion) with several PTP motions.

**Pass function of Jump3CP**
When the CP parameter is specified for Jump3CP with 0 approach motion, the Jump3CP span motion does not decelerate to a stop but goes on smoothly to the next CP motion. When the CP parameter is specified for a CP motion command right before Jump3CP with 0 depart motion, the CP motion does not decelerate to a stop but connects smoothly with the Jump3CP span motion. This is useful when you want to replace the span motion of Jump3CP (a CP motion) with several CP motions.

(Example 1)

Jump3 P1, P2, P2 CP
Go P3, P4 CP
Jump3 P4, P5, P5+tlz(50)

(Example 2)

Jump3CP P1, P2, P2 CP
Move P3, P4 CP
Jump3CP P4, P5, P5+tlz(50)

Using Jump3, Jump3CP with LJM

With LJM parameter, the program using LJM function can be more simple. For example, the following four-line program

\[
\begin{align*}
P11 &= \text{LJM}(P1, \text{Here}, 2) \\
P12 &= \text{LJM}(P2, P11, 2) \\
P13 &= \text{LJM}(P3, P12, 2) \\
\text{Jump3} &= P11, P12, P13
\end{align*}
\]

can be... the one-line program.

\[
\text{Jump3} P1, P2, P3 \text{ LJM 2}
\]

LJM parameter is available for 6-axis (including N series) and RS series robots. Jump3CP span motion is straight line (CP) motion and it cannot switch the wrist orientation along the way. Therefore, do not use the orientationFlag (LJM 1) of LJM function which is able to switch the wrist orientation.

Caution for Arch motion

Jump3 Motion trajectory changes depending on motion and speed

Jump3 motion trajectory is comprised of depart, span, and approach motions. It is not a continuous path trajectory. The actual Jump3 trajectory of arch motion is not determined by Arch parameters alone. It also depends on motion and speed.

Always use care when optimizing Jump3 trajectory in your applications. Execute Jump3 with the desired motion and speed to verify the actual trajectory.

When speed is lower, the trajectory will be lower. If Jump3 is executed with high speed to verify an arch motion trajectory, the end effector may crash into an obstacle with lower speed.

In a Jump3 trajectory, the depart distance increases and the approach distance decreases when the motion speed is set high. When the approach distance of the trajectory is shorter than the expected, lower the speed and/or the deceleration, or change the approach distance to be larger.

Even if Jump commands with the same distance and speed are executed, the trajectory is affected by motion of the robot arms.
Jump3, Jump3CP Statements

Potential acceleration errors

When the majority of depart (approach) motion uses the same joint as the span motion
An acceleration error may occur during an arch motion execution by the Jump3 and Jump3CP commands. This error is issued frequently when the majority of the motion during depart or approach uses the same joint as the span motion. To avoid this error, reduce the acceleration/deceleration speed of the span motion using Accel command for Jump3 or using AccelS command for Jump3CP. Depending on the motion and orientation of the robot, it may also help to reduce the acceleration and deceleration of the depart motion (approach motion) using the AccelS command.

See Also
Accel, Arc, Arch, Go, JS, JT, Point Expression, Pulse, Sense, Speed, Stat, Till

Jump3 Statement Example

` 6 axis robot (including N series) motion which works like Jump of SCARA robot
  Jump3  Here :Z(100), P3 :Z(100), P3

  Depart and approach use Z tool coordinates
  Jump3  Here -TLZ(100), P3 -TLZ(100), P3

  Depart uses base Z and approach uses tool Z
  Jump3  Here +Z(100), P3 -TLZ(100), P3

Example for the depart motion from P1 in Tool 1 and the approach motion to P3 in Tool 2

  Arch 0,20,20
  Tool 1
  Go P1

  P2 = P1 -TLZ(100)
  Tool 2
  Jump3 P2, P3-TLZ(100), P3 C0`
JumpTLZ Statement

3D gate motion. 
JumpTLZ is a combination of two CP motions and one PTP motion.

Syntax

JumpTLZ destination, TLZ movement [, CarchNumber] [, CP] [, LJM [, orientationFlag]] [, searchExpr] [, !...!] [, SYNC]

Parameters

destination The target destination of the motion using a point expression.

TLZ movement The amount of movement in Z direction in Tool coordinate system. The unit is [mm].

The Tool coordinate system for the currently used Tool number is used.

archNumber Optional. The arch number (archNumber) specifies which Arch Table entry to use for the Arch type motion caused by the JumpTLZ instruction. archNumber must always be proceeded by the letter C. (Valid entries are C0 to C7.)

CP Optional. Specifies continuous path motion.

LJM Optional. Convert the target destination using LJM function.

orientationFlag Optional. Specifies a parameter that selects an orientation flag for LJM function.

searchExpr Optional. A Sense, Till or Find expression.

Sense | Till | Find
Sense Sw(expr) = {On | Off}
Till Sw(expr) = {On | Off}
Find Sw(expr) = {On | Off}

!...! Optional. Parallel Processing statements can be added to the Jump3 and Jump3CP instruction to cause I/O and other commands to execute during motion.

SYNC Reserves a motion command. The robot will not move until SyncRobots is executed.
JumpTLZ Statement

Description
Moves the arm from the current position to the destination point with 3D gate motion. 3D gate motion consists of depart motion, span motion, and approach motion. The depart motion form the current position to the depart point is always CP motion. The span motion from the depart point to the start approach point is PTP motion.
The depart point is a point moved from the current position with $TLZ$ amount in the Z direction.
The robot posture at the depart point is same as the current position. (Posture may change if the robot passes the singularity or singularity neighborhood.)
The approach point is a point moved from the depart point in X and Y direction of the Tool coordinate system with move amount to approach to the destination point. The U, V, and W coordinates and the robot posture at the depart point are the same as the current position. (Posture may change if the robot passes the singularity or singularity neighborhood)

Arch motion is achieved by specifying the arch number. For arch motion to occur, the Depart distance must be greater than the arch upward distance and the Approach distance must be greater than the arch downward distance.

Notes
LimZ does not affect JumpTLZ
LimZ has no effect on JumpTLZ since the span motion is not necessarily perpendicular to the Z axis of the coordinate system.

JumpTLZ span motion is PTP (point to point)
It is difficult to predict JumpTLZ span motion trajectory. Therefore, be careful that the robot doesn't collide with peripheral equipment and that robot arms don't collide with the robot.

Difference between JumpTLZ and Jump3
JumpTLZ and Jump3 are different in the following points.
JumpTLZ:
The depart point must be in the Z direction from the current position.
The approach point must be in the Z direction from the destination point. Also, the approach distance cannot be specified.
Different Tool coordinate systems cannot be selected for the depart, approach, and destination points.
(It is not possible to execute the depart motion in Tool1, and execute the approach motion in Tool2.)

Jump3:
The depart point can be anywhere.
The approach point can be anywhere.
Different Tool coordinate systems can be selected for the depart, approach, and destination points.
(It is possible to execute the depart motion in Tool1, and execute the approach motion in Tool2.)

**Applicable manipulators**

JumpTLZ is only available for N series.

---

**Caution for Arch motion**

JumpTLZ motion trajectory is comprised of depart, span, and approach motions. It is not a continuous path trajectory. The actual JumpTLZ trajectory of arch motion is not determined by Arch parameters alone. It also depends on motion and speed.

Always use care when optimizing JumpTLZ trajectory in your applications. Execute JumpTLZ with the desired motion and speed to verify the actual trajectory.

When speed is lower, the trajectory will be lower. If JumpTLZ is executed with high speed to verify an arch motion trajectory, the end effector may crash into an obstacle with lower speed.

In a JumpTLZ trajectory, the depart distance increases and the approach distance decreases when the motion speed is set high. When the approach distance of the trajectory is shorter than the expected, lower the speed and/or the deceleration, or change the approach distance to be larger.

Even if JumpTLZ commands with the same distance and speed are executed, the trajectory is affected by motion of the robot arms.

---

**Potential acceleration errors**

**When the majority of depart (approach) motion uses the same joint as the span motion**

An acceleration error may occur during an arch motion execution by the JumpTLZ command. This error is issued frequently when the majority of the motion during depart or approach uses the same joint as the span motion. To avoid this error, reduce the acceleration/deceleration speed of the span motion using Accel command for JumpTLZ. Depending on the motion and orientation of the robot, it may also help to reduce the acceleration and deceleration of the depart motion (approach motion) using the AccelS command.

---

**See Also**

Accel, Arc, Arch, Go, JS, JT, Point Expression, Pulse, Sense, Speed, Stat, Till

---

**JumpTLZ Example**

Move 100 mm upward from the current point in Z direction of the Tool coordinate system. Then, move to the target point (P0):

```
JumpTLZ P0, -100
```
LatchEnable Statement

Enables / Disables the latch function for the robot position by the R-I/O input.

Syntax

```
LatchEnable { On | Off }
```

Parameters

<table>
<thead>
<tr>
<th>On</th>
<th>Off</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>Off</td>
<td>Enables the latch function of the robot position.</td>
</tr>
<tr>
<td>Off</td>
<td>On</td>
<td>Disables the latch function of the robot position.</td>
</tr>
</tbody>
</table>

Result

When the parameter is omitted, displays that the current latch function is ON or OFF.

Description

Enables / Disables the latch function for the robot position using the trigger input signals connected to the R-I/O. It latches the robot position with the first trigger input after you enable the latch function.

To repeatedly latch the robot position, execute LatchEnable Off and then execute LatchEnable On again. To use the command repeatedly, it needs at least 60 ms interval for each command processing time but you do not need to consider the command executing time.

Note

Before enabling the latch function, set the trigger input port and trigger signal logic using SetLatch.

See Also

LatchPos Function, LatchState Function, SetLatch

LatchEnable Statement Example

```
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
    LatchEnable On  ' Enables the latch function
    Go P1
    Wait LatchState = True  ' Waits a trigger
    Print LatchPos  ' Displays the latched position
    LatchEnable Off  ' Disables the latch function
    Fend
```
Returns the latch state of robot position using the R-I/O.

**Syntax**

```
LatchState
```

**Return Values**

Returns True when the robot position has been latched, False when the latch is not finished.

When confirmed the latch completion, acquires the latched position information by LatchPos Function.

**See Also**

LatchEnable, LatchPos Function, SetLatch, Wait

**LatchState Function Example**

```plaintext
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
    LatchEnable On          ' Enables the latch function
    Go P1
    Wait LatchState = True  ' Wait a trigger
    Print LatchPos          ' Display the latched position
    LatchEnable Off         ' Disable the latch function
Fend
```
LatchPos Function

Returns the robot position latched using the R-I/O input signal.

Syntax

```
LatchPos ([WithToolArm | WithoutToolArm])
```

Parameters

Optional. If omitted, WithToolArm is used.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WithToolArm</td>
<td>0</td>
</tr>
<tr>
<td>WithoutToolArm</td>
<td>1</td>
</tr>
</tbody>
</table>

WithToolArm  Constant value of 0. Returns the position according to the Tool and Arm settings at function call.

WithoutToolArm Constant value of 0.1 Returns the position of Tool 0 and Arm 0, regardless of the Tool and Arm settings.

Return Values

Returns the robot position latched by the R-I/O input signal in point data. Executing this function needs approx. 15 msec for processing.

When WithToolArm is specified, returns the position according to the Tool and Arm settings at function call. When WithoutToolArm is specified, returns the position of Tool 0 and Arm 0, regardless of the Tool and Arm settings.

See Also

LatchEnable, LatchState Function, SetLatch

LatchPos Function Example

```
Function main
    SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE
    LatchEnable On  ' Enables the latch function
    Go P1
    Wait LatchState = True  ' Wait a trigger
    Print LatchPos  ' Display the latched position
    LatchEnable Off  ' Disable the latch function
Fend
```

To assign the return value of LatchPos to the point data:

```
P2 = LatchPos
```
LCase$ Function

Returns a string that has been converted to lowercase.

Syntax

LCase$(string)

Parameters

string A valid string expression.

Return Values

The converted lower case string.

See Also

LTrim$, Trim$, RTrim$, UCase$

LCase$ Function Example

str$ = "Data"
str$ = LCase$(str$) ' str$ = "data"
Left$ Function

Returns a substring from the left side of a string expression.

Syntax

\[ \text{Left$}(\text{string, count}) \]

Parameters

- **string**: String expression from which the leftmost characters are copied.
- **count**: The number of characters to copy from *string* starting with the leftmost character.

Return Values

Returns a string of the leftmost *number* characters from the character string specified by the user.

Description

Left$ returns the leftmost *number* characters of a string specified by the user. Left$ can return up to as many characters as are in the character string.

See Also

Asc, Chr$, InStr, Len, Mid$, Right$, Space$, Str$, Val

Left$ Function Example

The example shown below shows a program which takes a part data string as its input and parses out the part number, part name, and part count.

```vbscript
Function ParsePartData(DataIn$ As String, ByRef PartNum$ As String, ByRef PartName$ As String, ByRef PartCount As Integer)
    Integer pos
    String temp$
    pos = Instr(DataIn$, ",")
    PartNum$ = Left$(DataIn$, pos - 1)
    DataIn$ = Right$(DataIn$, Len(DataIn$) - pos)
    pos = Instr(DataIn$, ",")
    PartName$ = Left$(DataIn$, pos - 1)
    PartCount = Val(Right$(DataIn$, Len(DataIn$) - pos))
Fend
```

Some other example results from the Left$ instruction from the Command window.

```vbscript
> Print Left$("ABCDEFG", 2)
AB
> Print Left$("ABC", 3)
ABC
```
Len Function

Returns the number of characters in a character string.

Syntax
Len(string)

Parameters
string String expression.

Return Values
Returns an integer number representing the number of characters in the string which was given as an argument to the Len instruction.

Description
Len returns an integer number representing the number of characters in a string specified by the user. Len will return values between 0 and 255 (since a string can contain between 0 and 255 characters).

See Also
Asc, Chr$, InStr, Left$, Mid$, Right$, Space$, Str$, Val

Len Function Example
The example shown below shows a program which takes a part data string as its input and parses out the part number, part name, and part count.

Function ParsePartData(DataIn$ As String, ByRef PartNum$ As String, ByRef PartName$ As String, ByRef PartCount As Integer)
    Integer pos
    String temp$
    pos = Instr(DataIn$, ","
    PartNum$ = Left$(DataIn$, pos - 1)
    DataIn$ = Right$(DataIn$, Len(DataIn$) - pos)
    pos = Instr(DataIn$, ","
    PartName$ = Left$(DataIn$, pos - 1)
    PartCount = Val(Right$(DataIn$, Len(DataIn$) - pos))
Fend

Some other example results from the Len instruction from the command window.

> ? len("ABCDEFG")
7

> ? len("ABC")
3

> ? len(""
0
>
LimitTorque Statement

Sets / returns the upper limit torque value in High power mode.

Syntax
(1) LimitTorque AllMax
(2) LimitTorque j1Max, j2Max, j3Max, j4Max
(3) LimitTorque j1Max, j2Max, j3Max, j4Max, j5Max, j6Max
(4) LimitTorque

Parameters
AllMax Specify the upper limit torque value for all axes in high power mode by an integer number representing the percentage of the maximum momentary torque of each axis
j #n Max Specify the upper limit torque value for axis #n in high power mode by an integer number representing the percentage of the maximum momentary torque of axis #n

Result
Returns the current LimitTorque value when the parameters are omitted.

Description
Sets the upper limit value of torque in high power mode. Normally, the maximum torque is set and there is no need to change this setting value. This statement is useful to restrict the torque not to exceed which is necessary for the specific motion in order to reduce damage to the manipulator and equipment caused by collision with peripherals.
The upper limit value is a peak torque in specific motion measured by PTRQ with allowance considering the variation added (approximately 10%).

The torque lower than the upper limit value in Low power mode cannot be set for this command. The minimum values vary for models and joints. Obtain the setting value and confirm the actual upper limit value after setting the value.

In any of the following cases, LimitTorque becomes the default value.

Controller startup
Motor On
SFree, SLock, or Brake is executed
Reset or Reset Error is executed
Task end by STOP switch or Quit All

Note

Too low LimitTorque setting
LimitTorque restricts the torque for the specific motion with the set torque restriction value as the upper limit value, regardless of the size of torque necessary for the motion to be executed with the set acceleration/deceleration. Therefore, if the motion requires larger torque than the set upper limit value, the robot may not move properly and cause vibrational motion, noise, or position deviation error and overrun. Make sure to measure PTRQ before using the torque restriction function. If the above problems occur, set the upper limit value larger and adjust the value so that the manipulator can operate properly.
See Also
LimitTorque Function, Power, PTrq, RealTorque

LimitTorque Statement Example
Following is the example which operates the manipulator with the maximum torque of the Joint #1 at 80%.

Function main
  Motor On
  Power high
  Speed 100; Accel 100,100
  LimitTorque 80,100,100,100  \* Restricts the maximum torque of Joint #1 to 80%
  Jump P1  \* Executes the Jump motion
  Fend
LimitTorque Function

Returns the setting value of LimitTorque command.

Syntax

LimitTorque(jointNumber)

Parameters

jointNumber  Integer expression ranging from 1 to 9.
             Additional S axis is 8, and T axis is 9.

Return Values

Returns an integer number representing the setting value of LimitTorque command.

See Also

LimitTorque

LimitTorque Function Example

Print LimitTorque(1)  'Displays the LimitTorque value of the Joint #1.
LimitTorqueLP Statement

Sets / returns the upper limit torque value in Low power mode.

Syntax
(1) LimitTorqueLP AllMax
(2) LimitTorqueLP j1Max, j2Max, j3Max, j4Max
(3) LimitTorqueLP j1Max, j2Max, j3Max, j4Max, j5Max, j6Max
(4) LimitTorqueLP

Parameters
AllMax Specify the upper limit torque value for all axes in low power mode by an integer number representing the percentage of the maximum momentary torque of each axis

j #n Max Specify the upper limit torque value for axis #n in low power mode by an integer number representing the percentage of the maximum momentary torque of axis #n

Result
Returns the current LimitTorqueLP value when the parameters are omitted.
Displays the default value when the values are not changed by this command.

Description
Sets the upper limit value of torque in low power mode. Normally, the maximum torque is set and there is no need to change this setting value (the values vary depending on the robot models and axes. Approx. 15-60%). This command is useful to restrict the torque not to exceed which is necessary for the normal motion in order to reduce damage to the manipulator and equipment caused by collision with peripherals.

The upper limit value is a peak torque in the motion measured by PTRQ with allowance considering the variation added (40% is recommended). To apply the same value to a different robot, add a further 10-20% allowance.

The PTRQ value considers the default maximum torque in low power mode as 1.0. For example, when the default value before change is 27% and the value measured by PTRQ is 0.43, the upper limit value is as follows: 27% × 0.43 × 1.4 = 16.25. Then, round up the value and set 17.

The value lower than 5% or larger than the default value cannot be set for this command. If these values are set, the setting values lower than 5 will be rounded up to 5, and the values exceeding the default value will be rounded down to the default. For instance, when “LimitTorqueLP 100”, the values are returned to the default for all joints because the default value is always less than 100.

Obtain the setting value and confirm the actual upper limit value after setting the value. The LimitTorqueLP setting value is effective until the Controller is restarted.

Note
Too low LimitTorqueLP setting
LimitTorqueLP restricts the torque for the specific motion with the set torque restriction value as the upper limit value, regardless of the size of torque necessary for the motion to be executed with the set acceleration/deceleration. Therefore, if the motion requires larger torque than the set upper limit value, the robot may not move properly and cause position deviation error. Make sure to measure PTRQ before using the torque restriction function. If the above problem occurs, set the upper limit value larger and adjust the value so that the manipulator can operate properly.
LimitTorqueLP Statement

See Also
LimitTorqueLP Function, PTRQ

LimitTorqueLP Example
Following is the example which operates the manipulator with the maximum torque of the Joint #1 at 10%.

```plaintext
Function main
Motor On
Power low
LimitTorqueLP 10,27,31,42 ' Restricts the maximum torque of the Joint #1 to 10%
' Set the default value for other axes
Go P1 ' Executes the Go motion
Fend
```
LimitTorqueLP Function

Returns the setting value of LimitTorqueLP command.

Syntax

```
LimitTorqueLP(jointNumber)
```

Parameters

- `jointNumber` Integer expression ranging from 1 to 9.
  Additional S axis is 8, and T axis is 9.

Return Values

Returns an integer number representing the setting value of LimitTorqueLP command.

See Also

LimitTorqueLP

Len Function Example

```
Print LimitTorqueLP(1)  'Displays the LimitTorqueLP value of the Joint #1.
```
LimitTorqueStop Statement

Enables or disables the function to stop the robot when the upper limit torque is reached in High power mode.

Syntax

1. LimitTorqueStop status
2. LimitTorqueStop status, jointNumber
3. LimitTorqueStop

Parameters

status
On: Enables the function to stop the robot at the upper limit torque
Off: Disables the function to stop the robot at the upper limit torque

jointNumber
The joint number from 1 to 6.
(For SCARA robots, joint numbers are from 1 to 4)

Result

Returns the current LimitTorqueStop status if the parameter is omitted.

Description

LimitTorqueStop enables the function to stop the robot at the upper limit torque value in High power mode. The robot immediately stops when it reaches the upper limit torque (default is 100%). Using this command together with the torque restriction function of LimitTorque provides the effect to reduce damage on the robot and peripherals at a collision in High power mode.

This function can be enabled or disabled for each joint or all joints. The default is “all joints off”. The setting returns to the default at the Controller startup. In other cases, the setting does not change unless otherwise configured by this command explicitly.

When the upper limit torque is reached, Error 5040 “Motor torque output failure in high power state.” will be output and the robot will stop.

See Also

LimitTorque, LimitTorque Function

LimitTorqueStop Example

Following is the example which restricts the maximum torque of the Joint #1 at 30% and stops the robot immediately.

```
Function main
Motor On
Power high
Speed 20
Accel 20,20
LimitTorque 30,100,100,100
LimitTorqueStop On, 1
Go P1
Fend
```

' Restrictions the maximum torque of the Joint #1 to 30%
' Joint #1 immediately stops at the maximum torque
' Executes the Go motion
LimitTorqueStop Function

Returns the setting value of LimitTorqueStop command.

**Syntax**

```
LimitTorqueStop(jointNumber)
```

**Parameters**

- `jointNumber` Integer expression ranging from 1 to 6.

**Return Values**

Returns an integer number representing the setting value of LimitTorqueStop command.

- 0 = Off
- 1 = On

**See Also**

LimitTorqueStop

**LimitTorqueStop Function Example**

```
Print LimitTorqueStop(1) ' Displays the LimitTorqueStop value of the Joint #1.
```
LimitTorqueStopLP Statement

Enables or disables the function to stop the robot when the upper limit torque is reached in Low power mode.

Syntax

(1) LimitTorqueStopLP status
(2) LimitTorqueStopLP status, jointNumber
(3) LimitTorqueStopLP

Parameters

status
- On: Enables the function to stop the robot at the upper limit torque
- Off: Disables the function to stop the robot at the upper limit torque

jointNumber
- The joint number from 1 to 6.
  (For SCARA robots, joint numbers are from 1 to 4)

Result

Returns the current LimitTorqueStopLP status if the parameter is omitted.

Description

LimitTorqueStopLP enables the function to stop the robot at the upper limit torque value in Low power mode. The robot immediately stops when it reaches the upper limit torque. Using this command together with the torque restriction function of LimitTorqueLP provides the effect to reduce damage on the robot and peripherals at a collision in Low power mode.

This function can be enabled or disabled for each joint or all joints. The default is “all joints off”. The setting returns to the default at the Controller startup. In other cases, the setting does not change unless otherwise configured by this command explicitly.

When the upper limit torque is reached, Error 5041 “Motor torque output failure in low power state.” will be output and the robot will stop.

See Also

LimitTorqueLP, LimitTorqueLP Function

LimitTorqueStopLP Example

Following is the example which restricts the maximum torque of the Joint #3 at 15% and stops the robot immediately.

```plaintext
Function main
Motor On
Power low
LimitTorqueLP 20,27,15,42
LimitTorqueStopLP On, 3
Go P1
Fend
```

' Restricts the maximum torque of the Joint #3 to 15%
' Set the default value for other axes
' Joint #3 immediately stops at the maximum torque
' Executes the Go motion
LimitTorqueStopLP Function

Returns the setting value of LimitTorqueStopLP command.

Syntax

LimitTorqueStopLP(jointNumber)

Parameters

jointNumber  Integer expression ranging from 1 to 6.

Return Values

Returns an integer number representing the setting value of LimitTorqueStopLP command.
0 = Off
1 = On

See Also

LimitTorqueStopLP

LimitTorqueStopLP Function Example

Print LimitTorqueStopLP(3)  ' Displays the LimitTorqueStopLP value of the Joint #3.
LimZ Statement

Determines the default value of the Z joint height for Jump commands.

Syntax
(1) LimZ zLimit
(2) LimZ

Parameters
zLimit A coordinate value within the movable range of the Z joint.

Return Values
Displays the current LimZ value when parameter is omitted.

Description
LimZ determines the maximum Z joint height which the arm move to when using the Jump instruction, wherein the robot arm raises on the Z joint, moves in the X-Y plane, then lowers on the Z joint. LimZ is simply a default Z joint value used to define the Z joint ceiling position for use during motion caused by the Jump instruction. When a specific LimZ value is not specified in the Jump instruction, the last LimZ setting is used for the Jump instruction.

Notes

Resetting LimZ to 0
Restarting the controller, or executing the SFree, SLock, Motor On commands will initialize LimZ to 0.

LimZ Value is Not Valid for Arm, Tool, or Local Coordinates:
LimZ Z joint height limit specification is the Z joint value for the robot coordinate system. It is not the Z joint value for Arm, Tool, or Local coordinates. Therefore take the necessary precautions when using tools or end effectors with different operating heights.

LimZ does not affect Jump3 and Jump3CP
LimZ has no effect on Jump3 or Jump3CP since the span motion is not necessarily perpendicular to the Z axis of the coordinate system.

See Also
Jump

LimZ Statement Example
The example below shows the use of LimZ in Jump operations.

Function main
LimZ -10 ' Set the default LimZ value
Jump P1 ' Move up to Z=-10 position for Jump
Jump P2 LimZ -20 ' Move up to Z=-20 position for Jump
Jump P3 ' Move up to Z=-10 position for Jump
Fend
**LimZ Function**

Returns the current LimZ setting.

**Syntax**

LimZ

**Return Values**

Real number containing the current LimZ setting.

**See Also**

LimZ Statement

**LimZ Function Example**

```plaintext
Real savLimz

savLimz = LimZ
LimZ -25
Go pick
LimZ savLimZ
```
LimZMargin Statement

Sets and returns the setting value for error detection when operation starts at higher than the LimZ value.

Syntax

(1) **LimZMargin**  *LimZmargin*
(2) **LimZMargin**

Parameters

*LimZmargin*  a margin value for LimZ error detection

Return Values

If the parameter is omitted, current LimZMargin value will be returned.

Description

When Jump command is executed, Joint #3 lifts up to the position set by LimZ. However, if the start position of the joint is above the LimZ position, an error will occur. LimZMargin sets a margin value for the error detection. Default is 0.02 mm.

Note

Resetting LimZ to default

Restarting the controller, or executing the SFree, SLock, Motor On commands will initialize LimZ to the default value.

See Also

LimZMargin Function, LimZ

LimZ Statement Example

Following is a usage example of LimZMargin in Jump operation.

```plaintext
Function main
    LimZ -10                   \ sets LimZ default value
    LimZMargin 0.03            \ sets 0.03 mm for a margin of LimZ error detection
    Jump P1
    Jump P2 LimZ -20           \ horizontal movement with -10 at Jump execution
    Jump P3                   \ horizontal movement with -20 at Jump execution
Fend
```
LimZMargin Function

Returns the current LimZMargin setting.

Syntax
LimZMargin

Return Values
Real number containing the current LimZMargin setting.

See Also
LimZMargin Statement, LimZ Statement

LimZ Function Example

Real savLimzMargin

savLimzMargin = LimZMargin
LimZMargin 0.03
Jump pick
LimZ savLimzMargin
Line Input Statement

Reads input data of one line and assigns the data to a string variable.

Syntax

Line Input  stringVar$

Parameters

stringVar$  A string variable name. (the string variable must end with the $ character.)

Description

Line Input reads input data of one line from the display device and assigns the data to the string variable used in the Line Input instruction. When the Line Input instruction is ready to receive data from the user, it causes a “?” prompt to be displayed on the display device. The input data line after the prompt is then received as the value for the string variable. After inputting the line of data press the [ENTER] key.

See Also

Input, Input #, Line Input#, ParseStr

Line Input Statement Example

The example below shows the use of Line Input.

```plaintext
Function Main
  String A$
  Line Input A$  'Read one line input data into A$
  Print A$
Fend
```

Run the program above using the F5 key or Run menu from EPSON RC+ main screen. A resulting run session may be as follows:

```
?A, B, C
A, B, C
```
Line Input # Statement

Reads data of one line from a file, communication port, database, or the device.

Syntax

Line Input #portNumber, stringVar$

Parameters

- **portNumber**: ID number representing a file, communications port, database, or device. File number can be specified in ROpen, WOpen, and AOpen statements. Communications port number can be specified in OpenCom (RS232) and OpenNet (TCP/IP) statements. Database number can be specified in OpenDB statement. Device ID integers are as follows.
  - 21 RC+
  - 23 OP
  - 24 TP (TP1 only)
  - 20 TP3

- **stringVar$**: A string variable. (string variables must end with a $ character.)

Description

Line Input # reads string data of one line from the device specified with the portNumber parameter, and assigns the data to the string variable stringVar$.

See Also

Input, Input #, Line Input

Line Input # Statement Example

This example receives the string data from the communication port number 1, and assigns the data to the string variable A$.

```epsonrc+
Function lintest
  String a$
  Print #1, "Please input string to be sent to robot"
  Line Input #1, a$
  Print "Value entered = ", a$
Fend
```
LJM Function

Returns the point data with the orientation flags converted to enable least joint motion when moving to a specified point based on the reference point.

Syntax

\[
\text{LJM (Point[, refPoint[, orientationFlag]]})
\]

Parameters

- **Point**: Specifies point data.
- **refPoint**: Specifies the reference point data. When this is omitted, the reference point is the current position (Here).
- **orientationFlag**: 6-axis robot

1: Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #4 will be the shortest movement. This is the default setting when “orientationFlag” is omitted.
2: Converts the J4Flag or J6Flag.
3: Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #5 will be the shortest movement.
4: Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #6 will be the shortest movement.

<table>
<thead>
<tr>
<th>“orientation Flag”</th>
<th>Hand orientation</th>
<th>Elbow orientation</th>
<th>Wrist orientation</th>
<th>J1Flag</th>
<th>J4Flag</th>
<th>J6Flag</th>
<th>Priority order of axis with the shortest movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J4</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J5</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J6</td>
</tr>
</tbody>
</table>

Note: Orientation of “-” is the same as the orientation specified by “refPoint”.

**RS series**

1: Converts the hand orientation (Hand Flag), J1Flag or J2Flag. This is the default setting when “orientationFlag” is omitted.
2: Converts the hand orientation (Hand Flag), J1Flag or J2Flag. Prevents the U axis from moving out of motion range at “orientationFlag” convert.

**N2 series**

1: Converts to the posture with minimum joint movement in priority order of Joint #1 and Joint #5. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag. The elbow orientation (Elbow Flag) is always above elbow orientation. This is the default setting when “orientationFlag” is omitted.
2: Converts to the posture with minimum joint movement in priority order of Joint #1 and Joint #4. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag. The elbow orientation (Elbow Flag) is always above elbow orientation.
3: Converts the wrist orientation (Wrist Flag), J4Flag, and J6Flag so that Joint #4 will be the shortest movement.
4: Converts the J4Flag and J6Flag.
5: Change the hand orientation specified by “refPoint” to different hand orientation (Hand Flag). Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #5 will be the shortest movement. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag. The elbow orientation (Elbow Flag) is always above elbow orientation.

6: Change the hand orientation specified by “refPoint” to different hand orientation (Hand Flag). Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #4 will be the shortest movement. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag. The elbow orientation (Elbow Flag) is always above elbow orientation.

7: Change the elbow orientation to the below elbow orientation (Elbow Flag). To be the shortest movement, converts the wrist orientation (Wrist Flag), J4Flag, and J6Flag in priority order of Joint #1 and Joint #5. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag.

8: Change the elbow orientation to the below elbow orientation (Elbow Flag). To be the shortest movement, converts the wrist orientation (Wrist Flag), J4Flag, and J6Flag in priority order of Joint #1 and Joint #4. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag.

<table>
<thead>
<tr>
<th>“orientation Flag”</th>
<th>Hand orientation</th>
<th>Elbow orientation</th>
<th>Wrist orientation</th>
<th>J4Flag</th>
<th>J6Flag</th>
<th>Priority order of axis with the shortest movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J1&gt;J5</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J1&gt;J4</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J4</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J4</td>
</tr>
<tr>
<td>6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J5</td>
</tr>
<tr>
<td>7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J1&gt;J5</td>
</tr>
<tr>
<td>8</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J1&gt;J4</td>
</tr>
</tbody>
</table>

Note: Orientation of “-” is the same as the orientation specified by “refPoint”.

*1: Above elbow orientation
*2: Hand orientation is different from the orientation specified by “refPoint”.
*3: Below elbow orientation

N6 series

1: Converts the wrist orientation (Wrist Flag), J4Flag, and J6Flag so that Joint #4 will be the shortest movement. This is the default setting when “orientationFlag” is omitted.

2: Converts the J4Flag and J6Flag.

3: Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #5 will be the shortest movement.

4: Converts the wrist orientation (Wrist Flag), J4Flag, J6Flag or J1Flag so that Joint #6 will be the shortest movement.

<table>
<thead>
<tr>
<th>“orientation Flag”</th>
<th>Hand orientation</th>
<th>Elbow orientation</th>
<th>Wrist orientation</th>
<th>J1Flag</th>
<th>J4Flag</th>
<th>J6Flag</th>
<th>Priority order of axis with the shortest movement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J4</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J5</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>J6</td>
</tr>
</tbody>
</table>

Note: Orientation of “-” is the same as the orientation specified by “refPoint”.

115x621: Change the elbow orientation to the below elbow orientation (Elbow Flag). To be the shortest movement, converts the wrist orientation (Wrist Flag), J4Flag, and J6Flag in priority order of Joint #1 and Joint #5. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag.

8: Change the elbow orientation to the below elbow orientation (Elbow Flag). To be the shortest movement, converts the wrist orientation (Wrist Flag), J4Flag, and J6Flag in priority order of Joint #1 and Joint #4. The target postures are hand orientation (Hand Flag), elbow orientation (Elbow Flag), wrist orientation (Wrist Flag), J4Flag, and J6Flag.

Note: Orientation of “-” is the same as the orientation specified by “refPoint”.

*1: Above elbow orientation
*2: Hand orientation is different from the orientation specified by “refPoint”.
*3: Below elbow orientation
LJM Function

Description
When the 6-axis or N series robot moves to a point calculated by such as pallet or relative offsets, the wrist part may rotate to an unintended direction. The point calculation above does not depend on robot models and results in motion without converting the required point flag.
LJM function can be used to convert the point flag to prevent the unintended wrist rotation.

For the N series robots, it is also possible to reduce the cycle time and omit teaching of the avoidance point, which is necessary for the 6-axis robots, by changing the Hand Flag and Elbow Flag.

When the RS series robot moves to a point calculated by such as pallet or relative offsets, Arm #1 may rotate to an unintended direction. LJM function can be used to convert the point flag to prevent the unintended rotation of Arm #1.

In addition, the U axis of an RS series robot may go out of motion range when the orientation flag is converted, which will cause an error.
To prevent this error, the LJM function adjusts the U axis target angle so that it is inside the motion range.
This is available when “2” is selected for orientationFlag.

Returns the specified point for all robots except the 6-axis, N series, and RS series robot.

Note

The reference point omission and Parallel Processing
You cannot use both of the parallel point omission and parallel processing in one motion command like this:
Go LJM(P10) ! D10; MemOn 1 !

Be sure to change the program like this:
P999 = Here
Go LJM(P10,P999) ! D10; MemOn 1 !

orientationFlag for N2 series
- orientationFlag 1, 2:
  To shorten the cycle time, select orientationFlag 1 or 2.
  Since the posture has minimum Joint #1 movement, the cycle time can be shortest in most motion.
  To reduce the Joint #5 movement, select orientationFlag 1.
  To reduce the Joint #4 movement, select orientationFlag 2.
- orientationFlag 3, 4:
  Use these flags if you do not want to change the reference orientation, hand orientation, and elbow orientation.
  Use these flages if you want to use them in a same manner as the flags for vertical 6-axis robots.
  orientationFlag 3 is same as orientationFlag 1 of the vertical 6-axis robots.
  orientationFlag 4 is same as orientationFlag 2 of the vertical 6-axis robots.
- orientationFlag 5, 6:
  If the hand collides with peripheral walls during the operation, select orientationFlag 5 or 6.
  Since the hand passes the neighborhood of the robot’s origin point, the robot can move with less possibility to collide with the obstacles.
  To reduce the Joint #5 movement, select orientationFlag 5.
  To reduce the Joint #4 movement, select orientationFlag 6.
- orientationFlag 7, 8:
  To have a below elbow orientation, select orientationFlag 7 or 8.
  Depending on motion, the robot passes the neighborhood of the origin like orientationFlag 5 and orientationFlag 6. Therefore, the robot can move with less possibility to collide with the obstacles, if these are located around the robot.
  To reduce the Joint #5 movement, select orientationFlag 7.
  To reduce the Joint #4 movement, select orientationFlag 8.

localNumber
Local numbers of the points returned by LJM function are the same as that of “Point Expression”.

426  EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
LJM Function Example

Function main

```lua
Integer i, j
P0 = XY(300, 300, 300, 90, 0, 180)
P1 = XY(200, 280, 150, 90, 0, 180)
P2 = XY(200, 330, 150, 90, 0, 180)
P3 = XY(-200, 280, 150, 90, 0, 180)
Pallet 1, P1, P2, P3, 10, 10
Motor On
Power High
Speed 50; Accel 50, 50
SpeedS 1000; AccelS 5000
Go P0
P11 = P0 -TLZ(50)
For i = 1 To 10
    For j = 1 To 10
        Specify points
            P10 = P11
            P12 = Pallet(1, i, j)
            P11 = P12 -TLZ(50)
        Converting each point to LJM
            P10 = LJM(P10)
            P11 = LJM(P11, P10)
            P12 = LJM(P12, P11)
        Execute motion
            Jump3 P10, P11, P12 C0
    Next
Next
```

Function main2

```lua
P0 = XY(300, 300, 300, 90, 0, 180)
P1 = XY(400, 0, 150, 90, 0, 180)
P2 = XY(400, 500, 150, 90, 0, 180)
P3 = XY(-400, 0, 150, 90, 0, 180)
Pallet 1, P1, P2, P3, 10, 10
Motor On
Power High
Speed 50; Accel 50, 50
SpeedS 1000; AccelS 5000
Go P0
Do
    Specify points
        P10 = Here -TLZ(50)
        P12 = Pallet(1, Int(Rnd(9)) + 1, Int(Rnd(9)) + 1)
        P11 = P12 -TLZ(50)
    Converting each point to LJM
        P10 = LJM(P10)
        P11 = LJM(P11, P10)
        P12 = LJM(P12, P11)
    Execute motion
        Jump3 P10, P11, P12 C0
    Loop
```

See Also

Pallet
LoadPoints Statement

Loads a point file into the point memory area for the current robot.

Syntax

\[
\text{LoadPoints } \text{fileName} \ [\text{, Merge}]
\]

Parameters

- **fileName**: String expression containing the specific file to load into the current robot's point memory area. The extension must be .PTS. The file must exist in the current project for the current robot. You cannot specify a file path and fileName doesn’t have any effect from ChDisk. See ChDisk for the details.

- **Merge**: Optional. If supplied, then the current points are not cleared before loading the new points. Points in the file are added to the current points. If a point exists in the file, it will overwrite the point in memory.

Description

LoadPoints loads point files from disk into the main memory area of the controller for the current robot. Use Merge to combine point files. For example, you could have one main point file that includes common points for locals, parking, etc. in the range from 0 to 100. Then use Merge to load other point files for each part being run without clearing the common points. The range could be from 101 to 999.

Potential Errors

- **A Path Cannot be Specified**
  
  If `fileName` contains a path, an error will occur. Only a file name in the current project can be specified.

- **File Does Not Exist**
  
  If `fileName` does not exist, an error will occur.

- **Point file not for the current robot**
  
  If `fileName` is not a point file for the current robot, the following error will be issued: Point file not found for current robot. To correct this, add the Point file to the robot in the Project editor, or execute SavePoints or ImportPoints.

See Also

ImportPoints, Robot, SavePoints

LoadPoints Statement Example

```plaintext
Function main
  ' Load common points for the current robot
  LoadPoints "R1Common.pts"

  ' Merge points for part model 1
  LoadPoints "R1Model1.pts", Merge

Robot 2
  ' Load point file for the robot 2
  LoadPoints "R2Model1.pts"

Fend
```
Local Statement

Defines and displays local coordinate systems.

Syntax
(1) Local localNumber, (pLocal1 : pBase1), (pLocal2 : pBase2) ![L | R] [BaseU]
(2) Local localNumber, pCoordinateData
(3) Local localNumber, pOrigin, [pXaxis], [pYaxis], ![X | Y]
(4) Local localNumber

Parameters
localNumber The local coordinate system number. A total of 15 local coordinate systems (of the integer value from 1 to 15) may be defined.
pLocal1, pLocal2 Point variables with point data in the local coordinate system.
pBase1, pBase2 Point variables with point data in the base coordinate system.
L | R Optional. Align local origin to left (first) or right (second) base points.
BaseU Optional. When supplied, U axis coordinates are in the base coordinate system. When omitted, U axis coordinates are in the local coordinate system.
pCoordinateData Point data representing the coordinate data of the origin and direction.
pOrigin Integer expression representing the origin point using robot coordinate system.
pXaxis Optional. Integer expression representing a point along the X axis using robot coordinate system if X alignment is specified.
pYaxis Optional. Integer expression representing a point along the Y axis using robot coordinate system if Y alignment is specified.
X | Y If X alignment is specified, then pXaxis lies on the X axis of the local. The Y axis and Z axis are calculated to be orthogonal to X in the plane that is created by the 3 local points. If Y alignment is specified, then pYaxis lies on the Y axis of the local. The X axis and Z axis are calculated to be orthogonal to Y in the plane that is created by the 3 local points.

Description
(1) Local defines a local coordinate system by specifying 2 points, pLocal1 and pLocal2, contained in it that coincide with two points, pBase1 and pBase2, contained in the base coordinate system.

Example:
Local 1, (P1:P11), (P2:P12)
P1 and P2 are local coordinate system points. P11 and P12 are base coordinate system points.

If the distance between the two specified points in the local coordinate system is not equal to that between the two specified points in the base coordinate system, the XY plane of the local coordinate system is defined in the position where the midpoint between the two specified points in the local coordinate system coincides with that between the two specified points in the base coordinate system.

Similarly, the Z axis of the local coordinate system is defined in the position where the midpoints coincide with each other.
(2) Defines a local coordinate system by specifying the origin and axis rotation angles with respect to the base coordinate system.

Example:
- `Local 1, XY(x, y, z, u)`
- `Local 1, XY(x, y, z, u, v, w)`
- `Local 1, P1`

(3) Defines a 3D local coordinate system by specifying the origin point, x axis point, and y axis point. Only the X, Y, and Z coordinates of each point are used. The U, V, and W coordinates are ignored. When the X alignment parameter is used, then `pXaxis` is on the X axis of the local and only the Z coordinate of `pYaxis` is used. When the Y alignment parameter is used, then `pYaxis` is on the Y axis of the local and only the Z coordinate of `pXaxis` is used.

Example:
- `Local 1, P1, P2, P3`
- `Local 1, P1, P2, P3, X`
- `Local 1, P1, P2, P3, Y`

(4) Displays the specified local settings.

Using L and R parameters

While Local basically uses midpoints for positioning the axes of your local coordinate system as described above, you can optionally specify left or right local by using the L and R parameters.

**Left Local**
Left local defines a local coordinate system by specifying point `pLocal1` corresponding to point `pBase1` in the base coordinate system (Z axis direction is included.)

**Right Local**
Right local defines a local coordinate system by specifying point `pLocal2` corresponding to point `pBase2` in the base coordinate system. (Z axis direction is included.)

Using the BaseU parameter

If the BaseU parameter is omitted, then the U axis of the local coordinate system is automatically corrected in accordance with the X and Y coordinate values of the specified 4 points. Therefore, the 2 points in the base coordinate system may initially have any U coordinate values.

It may be desired to correct the U axis of the local coordinate system based on the U coordinate values of the two points in the base coordinate system, rather than having it automatically corrected (e.g. correct the rotation axis through teaching). To do so, supply the BaseU parameter.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also
- ArmSet, Base, ECPSet, LocalClr, TLSet, Where
**Local Statement Examples**

Here are some examples from the command window:

**Left aligned local:**

```
> p1 = 0, 0, 0, 0/1
> p2 = 100, 0, 0, 0/1
> p11 = 150, 150, 0, 0
> p12 = 300, 150, 0, 0
> local 1, (P1:P11), (P2:P12), L
> p21 = 50, 0, 0, 0/1
> go p21
```

**Local defined with only the origin point:**

```
> local 1, 100, 200, -20, 0
```

**Local defined with only the origin point rotated 45 degrees about the X axis:**

```
> local 2, 50, 200, 0, 0, 45, 0
```

**3D Local with p2 aligned with the X axis of the local:**

```
> local 3, p1, p2, p3, x
```

**3D Local with p3 aligned with the Y axis of the local:**

```
> local 4, p1, p2, p3, y
```
Local Function

Returns the specified local coordinate system data as a point.

Syntax

Local(localNumber)

Parameters

localNumber    local coordinate system number (integer from 1 to 15) using an expression or numeric value.

Return Values

Specified local coordinate system data as point data.

See Also

Local Statement

Local Function Example

P1 = Local(1)
LocalClr Statement

Clears (undefines) a local coordinate system.

Syntax

```
LocalClr localNumber
```

Parameters

```
localNumber
```

Integer expression representing which of 15 locals (integer from 1 to 15) to clear (undefine).

Description

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also

Arm, ArmSet, ECPSet, Local, Tool, TLClr, TLSet

LocalClr Statement Example

```
LocalClr 1
```
LocalDef Function

Returns local definition status.

Syntax

LocalDef (localCoordinateNumber)

Parameters

localCoordinateNumber  Integer expression representing which local coordinate to return status for.

Return Values

True if the specified local has been defined, otherwise False.

See Also

Arm, ArmClr, ArmSet, ECPSet, Local, LocalClr, Tool, TLClr, TLSet

LocalDef Function Example

Function DisplayLocalDef(localNum As Integer)
    
    If LocalDef(localNum) = False Then
        Print "Local ", localNum, "is not defined"
    Else
        Print "Local 1: ",
        Print Local(localNum)
    EndIf
EndFunction
Lof Function

Checks whether the specified RS-232 or TCP/IP port has any lines of data in its buffer.

Syntax

Lof( fileNumber As Integer )

Parameters

fileNumber A number specified with OpenCom (RS-232C) or OpenNet (TCP/IP) statement.

Return Values

The number of lines of data in the buffer. If there is no data in the buffer, Lof returns “0”.

Description

Lof checks whether or not the specified port has received data lines. The data received is stored in the buffer irrespective of the Input# instruction. You can wait for the return value of Lof function by executing Wait.

Note

When using PC COM port (1001 to 1008), you cannot use Lof function with Wait command.

See Also

ChkCom, ChkNet, Input#, Wait

Lof Function Example

This Command window example prints out the number of lines of data received through the communication port number 1.

>print lof(1)
5
>


LogIn Statement

Log into EPSON RC+ 6.0 as another user.

Syntax

LogIn logID, password

Parameters

logID
String expression that contains user login id.

password
String expression that contains user password.

Description

You can utilize EPSON RC+ security in your application. For example, you can display a menu that allows different users to log into the system. Each type of user can have its own security rights. For more details on security, see the EPSON RC+ User's Guide.

When you are running programs in the development environment, the user before programs are started will be restored after programs stop running.

When running the Operator Window in Auto Mode, the application is logged in as a guest user, unless Auto LogIn is enabled, in which case the application is logged in as the current Windows user if such user has been configured in the EPSON RC+ system.

Note

This command will only work if the Security option is active.

See Also

GetCurrentUser$ Function

LogIn Statement Example

Integer errCode
errCode = LogIn("operator", "oprpass")
Long Statement

Declares variables of type long integer. (4 byte whole number).

**Syntax**

Long varName [([subscripts])] [, varName [([subscripts])]...]

**Parameters**

| varName | Variable name which the user wants to declare as type Long. |
| subscripts | Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows (ubound1, [ubound2], [ubound3]) ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension. The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1. When specifying the upper bound value, make sure the number of total elements is within the range shown below: |
| Local variable | 2,000 |
| Global Preserve variable | 4,000 |
| Global variable and module variable | 100,000 |

**Description**

Long is used to declare variables as type Long. Variables of type Long can contain whole numbers with values between -2,147,483,648 to 2,147,483,647. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

**See Also**

Boolean, Byte, Double, Global, Int32, Int64, Integer, Real, Short, String, UByte, UInt32, UInt64, UShort

**Long Statement Example**

The following example shows a simple program which declares some variables as Longs using Long.

```
Function longtest
    Long A(10) 'Single dimension array of long
    Long B(10, 10) 'Two dimension array of long
    Long C(5, 5, 5) 'Three dimension array of long
    Long var1, arrayVar(10)
    Long i
    Print "Please enter a Long Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter a Long Number"
        Input arrayVar(i)
        Print "Value Entered was ", arrayVar(i)
    Next I
End
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 437
LSet$ Function

Returns the specified string with trailing spaces appended up to the specified length.

Syntax

LSet$ (string, length)

Parameters

string    String expression.
length    Integer expression for the total length of the string returned.

Return Values

Specified string with trailing spaces appended.

See Also

RSet$, Space$

LSet$ Function Example

```
temp$ = "123"
temp$ = LSet$(temp$, 10) ' temp$ = "123     "
```
LShift Function

Shifts numeric data to the left by a user specified number of bits.

Syntax

\[
\text{LShift}(\text{number, shiftBits})
\]

Parameters

- **number**: Integer expression to be shifted.
- **shiftBits**: The number of bits (integer from 0 to 31) to shift **number** to the left.

Return Values

Returns a numeric result which is equal to the value of **number** after shifting left **shiftBits** number of bits.

Description

LShift shifts the specified numeric data (**number**) to the left (toward a higher order digit) by the specified number of bits (**shiftBits**). The low order bits shifted are replaced by 0.

The simplest explanation for LShift is that it simply returns the result of \( \text{number} \times 2^{\text{shiftBits}} \).

Note

**Numeric Data Type:**

The numeric data **number** may be any valid numeric data type. LShift works with data types: Byte, Double, Int32, Integer, Long, Real, Short, UByte, UInt32, and UShort.

See Also

And, LShift64, Not, Or, RShift, RShift64, Xor

LShift Function Example

```plaintext
Function lshiftst
    Integer i
    Integer num, snum
    num = 1
    For i = 1 to 10
        Print "i =", i
        snum = LShift(num, i)
        Print "The shifted num is ", snum
    Next i
End
```

Some other example results from the LShift instruction from the command window.

```
> Print LShift(2,2)
8
> Print LShift(5,1)
10
> Print LShift(3,2)
12
> 
```
LShift64 Function

Shifts numeric data to the left by a user specified number of bits.

Syntax

```
LShift64(number, shiftBits)
```

Parameters

- **number**: Integer expression to be shifted.
- **shiftBits**: The number of bits (integer from 0 to 63) to shift number to the left.

Return Values

Returns a numeric result which is equal to the value of number after shifting left shiftBits number of bits.

Description

LShift64 shifts the specified numeric data (number) to the left (toward a higher order digit) by the specified number of bits (shiftBits). The low order bits shifted are replaced by 0.

The simplest explanation for LShift64 is that it simply returns the result of number * 2^{shiftBits}.

Note

Numeric Data Type:

The numeric data number may be any valid numeric data type. LShift64 works with data types: Int64 and UInt64.

See Also

And, LShift, Not, Or, RShift, RShift64, Xor

LShift64 Function Example

```plaintext
Function lshiftst
    Int64 i
    Int64 num, snum
    num = 1
    For i = 1 to 10
        Print "i =", i
        snum = LShift64(num, i)
        Print "The shifted num is ", snum
    Next i
End
```

Some other example results from the LShift64 instruction from the command window.

```
> Print LShift64(2,2)
8
> Print LShift64(5,1)
10
> Print LShift64(3,2)
12
> 
```
LTrim$ Function

Returns a string equal to specified string without leading spaces.

Syntax

LTrim$ (string)

Parameters

string String expression.

Return Values

Specified string with leading spaces removed.

See Also

RTrim$, Trim$

LTrim$ Function Example

```plaintext
str$ = "  data  
str$ = LTrim$(str$)  ' str$ = "data  
```
Mask Operator

Bitwise mask for Wait statement condition expression.

Syntax
Wait expr1 Mask expr2

Parameters
expr1 Any valid expression input condition for Wait.
expr2 Any valid expression which returns a numeric result.

Description
The Mask operator is a bitwise And for Wait statement input condition expressions.

See Also
Wait

Mask Operator Example

' Wait for the lower 3 bits of input port 0 to equal 1
Wait In(0) Mask 7 = 1
MCal Statement

Executes machine calibration for robots with incremental encoders.

**Syntax**

```
MCal
```

**Description**

It is necessary to calibrate robots which have incremental encoders. This calibration must be executed after turning on the main power. If you attempt motion command execution, or any command which requires the current position data without first executing machine calibration, an error will occur.

Machine calibration is executed according to the moving joint order which is specified with the MCordr command. The default value of MCordr at the time of shipment differs from model to model, so please refer to the proper manipulator manual for details.

**Potential Errors**

**Attempt to Execution a Motion command without Executing Mcal First**

If you attempt motion command execution, or any command which requires the current position data (e.g. Plist* instruction) without first executing machine calibration, an error will occur.

**Absolute encoder robots**

Absolute encoder robots do not need MCAL.

**Robot Installation Note**

**Z Joint Space Required for Homing**

When the Z joint homes it first moves up and then moves down and settles into the home position. This means it is very important to properly install the robot so that enough space is provided for the arm to home the Z joint. It is recommended that a space of 6 mm be provided above the upper limit. (Do not install tooling or fixtures within a 6 mm space above the robot so enough room is left for proper Z joint homing.)

**See Also**

Hofs, Home, Hordr, Mcorg, MCordr

**Mcal Statement Example**

The following example is done from the monitor window:

```
> Motor On
> Mcal
> 
```
MCalComplete Function

Returns status of MCal.

Syntax

MCalComplete

Return Values
True if MCal has been completed, otherwise False.

See Also
MCal

MCalComplete Function Example

    If Not MCalComplete Then
    MCal
    EndIf
MCordr Statement

Specifies and displays the moving joint order for machine calibration Mcal. Required only for robots with incremental encoders.

Syntax

(2) MCordr

Parameters

Step1  Bit pattern that tells which axes should be calibrated during the 1st step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 1st step. (see below for bit pattern definitions)

Step2  Bit pattern that tells which axes should be calibrated during the 2nd step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 2nd step. (see below for bit pattern definitions)

Step3  Bit pattern that tells which axes should be calibrated during the 3rd step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 3rd step. (see below for bit pattern definitions)

Step4  Bit pattern that tells which axes should be calibrated during the 4th step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 4th step. (see below for bit pattern definitions)

Step5  Bit pattern that tells which axes should be calibrated during the 5th step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 5th step. (see below for bit pattern definitions)

Step6  Bit pattern that tells which axes should be calibrated during the 6th step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 6th step. (see below for bit pattern definitions)

Step7  Bit pattern that tells which axes should be calibrated during the 7th step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 7th step. (see below for bit pattern definitions)

Step8  Bit pattern that tells which axes should be calibrated during the 8th step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 8th step. (see below for bit pattern definitions)

Step9  Bit pattern that tells which axes should be calibrated during the 9th step of the Mcal process. Any number of axes between 0 to all 4 axes may calibrate during the 9th step. (see below for bit pattern definitions)

Return Values

Displays current Machine Calibration Order when parameters are omitted.

Description

After the system is powered on, Mcal instruction must be issued prior to any robot arm operation. When the Mcal instruction is issued each of the 4 axes of the robot will move to their respective calibration positions.

Specifies joint motion order for the Mcal command. (i.e. Defines which joint will home 1st, which joint will Mcal 2nd, 3rd, etc.)

The purpose of the MCordr instruction is to allow the user to change the homing order. The homing order is broken into 9 separate steps. The user then uses MCordr to define the specific axes which will move to the calibration position (done with the Mcal command) during each step. It is important to realize that more than 1 joint can be defined to move to the calibration position during a single step. This means that all four axes can potentially be calibrated at the same time. However, it is recommended that the Z joint normally be
MCorrdr Statement

defined to move to the calibration position first (in Step 1) and then allow the other Axes to follow in
subsequent steps. (See notes below)

The MCorrdr instruction expects that a bit pattern be defined for each of the 9 steps. Since there are 4 axes,
each joint is assigned a specific bit. When the bit is high (1) (for a specific step), then the corresponding
joint will calibrate. When the bit is low (0), then the corresponding joint will not calibrate during that step.
The joint bit patterns are assigned as follows:

<table>
<thead>
<tr>
<th>Joint</th>
<th>Bit Number</th>
<th>Binary Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>bit 0</td>
<td>&amp;B000001</td>
</tr>
<tr>
<td>2</td>
<td>bit 1</td>
<td>&amp;B000010</td>
</tr>
<tr>
<td>3</td>
<td>bit 2</td>
<td>&amp;B001000</td>
</tr>
<tr>
<td>4</td>
<td>bit 3</td>
<td>&amp;B001000</td>
</tr>
<tr>
<td>5</td>
<td>bit 4</td>
<td>&amp;B010000</td>
</tr>
<tr>
<td>6</td>
<td>bit 5</td>
<td>&amp;B100000</td>
</tr>
<tr>
<td>7</td>
<td>bit 6</td>
<td>&amp;B100000</td>
</tr>
<tr>
<td>8</td>
<td>bit 7</td>
<td>&amp;B1000000</td>
</tr>
<tr>
<td>9</td>
<td>bit 8</td>
<td>&amp;B10000000</td>
</tr>
</tbody>
</table>

Notes

Difference Between MCorrdr and Hordr
While at first glance the Hordr and MCorrdr commands may appear very similar there is one major difference
which is important to understand. MCorrdr is used to define the Robot Calibration joint order (used with
Mcal ) while Hordr is used to define the Homing joint order (used with the Home command).

Default MCal Order (Factory Setting)
The default joint calibration order from the factory is that joint 3 will home in Step 1. Then joints 1, 2, and
4 joints will all home at the same time in step 2. (Steps 3 and 4 are not used in the default configuration.)
The default MCorrdr values are as follows:
MCorrdr &B0100, &B1011, 0, 0

Z Joint should normally be calibrated first
The reason for moving the Z joint first (and by itself) is to allow the tooling to be moved above the work
surface before beginning any horizontal movement. This will help prevent the tooling from hitting something
in the work envelope during the homing process.

MCorrdr values are maintained
The MCorrdr Table values are permanently saved and are not changed until either the user changes them or
the robot is redefined.

See Also
Mcal

MCorrdr Statement Example
Following are some monitor window examples:

This example defines the calibration order as J3 in the first step, J1 in second step, J2 in third step, and J4
in the fourth step. The order is specified with binary values.

```
>mcordr &B0100, &B0001, &B0010, &B1000
```

This example defines the calibration order as J3 in the first step, then J1, J2 and J4 joints simultaneously in
the second step. The order is specified with decimal values.

```
>mcordr 4, 11, 0, 0
```

This example displays the current calibration order in decimal numbers.

```
>mcordr 4, 11, 0, 0
```
MCordr Function

Returns an MCordr parameter setting.

Syntax
MCordr (paramNumber)

Parameters
paramNumber Specifies reference setting numbers (integers from 1 to 9) by an expression or numeric value.

Return Values
Returns binary values (integers) representing the joint of the specified setting number to execute machine calibration.

Description
Returns the joint motion order to execute machine calibration by Mcal.

See Also
Mcal

MCordr Function Example
This example uses the MCordr function in a program:

```
Integer a
a = MCordr(1)
```
MemIn Function

Returns the status of the specified memory I/O port. Each port contains 8 memory bits.

Syntax

\[
\text{MemIn}(\text{portNumber})
\]

Parameters

\[
\text{portNumber} \quad \text{Integer expression representing memory I/O bytes.}
\]

Return Values

Returns an integer value between 0 and 255. The return value is 8 bits, with each bit corresponding to 1 memory I/O bit.

Description

MemIn provides the ability to look at the value of 8 memory I/O bits at the same time. The MemIn instruction can be used to store the 8 memory I/O bit status into a variable or it can be used with the Wait instruction to Wait until a specific condition which involves more than 1 memory I/O bit is met.

Since 8 bits are retrieved at a time, the return value ranges from 0 and 255. Please review the chart below to see how the integer return values correspond to individual memory I/O bits.

### Memory I/O Bit Result (Using Port #0)

<table>
<thead>
<tr>
<th>Return Values</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>5</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>15</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

### Memory I/O Bit Result (Using Port #31)

<table>
<thead>
<tr>
<th>Return Values</th>
<th>255</th>
<th>254</th>
<th>253</th>
<th>252</th>
<th>251</th>
<th>250</th>
<th>249</th>
<th>248</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>7</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>32</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

Note

**Difference Between MemIn and MemSw**

The MemSw instruction allows the user to read the value of 1 memory I/O bit. The return value from MemSw is either a 1 or a 0 which indicates that the memory I/O bit is either On or Off. MemSw can check each of the memory I/O bits individually. The MemIn instruction is very similar to the MemSw instruction in that it also is used to check the status of the memory I/O bits. However there is 1 distinct difference. The MemIn instruction checks 8 memory I/O bits at a time vs. the single bit checking functionality of the MemSw instruction. MemIn returns a value between 0 and 255 which tells the user which of the 8 I/O bits are On and which are Off.

**See Also**

In, InBCD, Off, MemOff, On, MemOn, OpBCD, Oport, Out, MemOut, Sw, MemSw, Wait
**MemIn Function Example**

The program example below gets the current value of the first 8 memory I/O bits and then makes sure that all 8 I/O are currently set to “0” before proceeding. If they are not “0” an error message is given to the operator and the task is stopped.

```plaintext
Function main
    Integer var1
    var1 = MemIn(0)  'Get the 1st 8 memory I/O bit value
    If var1 = 0 Then
        Go P1
        Go P2
    Else
        Print "Error in initialization!"
        Print "First 8 memory I/O bits were not all set to 0"
    EndIf
EndFend
```

Other simple examples from the Command window are as follows:

```plaintext
> memout 0, 1
> print MemIn(0)
1
> memon 1
> print MemIn(0)
3
> memout 31, 3
> print MemIn(31)
3
> memoff 249
> print MemIn(31)
1
> 
```
MemInW Function

Returns the status of the specified memory I/O word port. Each word port contains 16 memory I/O bits.

Syntax

```
MemInW(WordPortNum)
```

Parameters

- **WordPortNum**: Integer expression representing the I/O word port.

Return Values

- Returns the current status of the memory I/O (long integers from 0 to 65535).

See Also

- MemIn, MemOut, MemOutW

MemInW Function Example

```
Long word0

word0 = MemInW(0)
```
MemOff Statement

Turns Off the specified bit of the memory I/O.

Syntax

MemOff \{ bitNumber | memIOLabel \}

Parameters

- **bitNumber**: Integer expression representing memory I/O bits.
- **memIOLabel**: Memory I/O label.

Description

MemOff turns Off the specified bit of memory I/O. The 256 memory I/O bits are typically excellent choices for use as status bits for uses such as On/Off, True/False, Done/Not Done, etc. The MemOn instruction turns the memory bit On, the MemOff instruction turns it Off, and the MemSw instruction is used to check the current state of the specified memory bit. The Wait instruction can also be used with the memory I/O bit to cause the system to wait until a specified memory I/O status is set.

Note

**Memory outputs off**

All memory I/O bits are turned off when the controller are restarted. They are not turned off by Emergency stop, safeguard open, program end, Reset command, or EPSON RC+ restart.

See Also

In, MemIn, InBCD, Off, On, MemOn, OpBCD, Oport, Out, MemOut, Sw, MemSw, Wait

MemOff Statement Example

The example shown below shows 2 tasks each with the ability to initiate motion instructions. However, a locking mechanism is used between the 2 tasks to ensure that each task gains control of the robot motion instructions only after the other task is finished using them. This allows 2 tasks to each execute motion statements as required and in an orderly predictable fashion. MemSw is used in combination with the Wait instruction to wait until the memory I/O #1 is the proper value before it is safe to move again. MemOn and MemOff are used to turn on and turn off the memory I/O for proper synchronization.

```plaintext
Function main
    Integer I
    MemOff 1
    Xqt 2, task2
    For i = 1 to 100
        Wait MemSw(1) = Off
        Go P(i)
        MemOn 1
    Next I
    Fend

Function task2
    Integer I
    For i = 101 to 200
        Wait MemSw(1) = On
        Go P(i)
        MemOff 1
    Next I
    Fend
```
Other simple examples from the command window are as follows:

> MemOn 1     'Switch memory I/O bit #1 on
> Print MemSw(1)
1
> MemOff 1     'Switch memory I/O bit #1 off
> Print MemSw(1)
0
MemOn Statement

Turns On the specified bit of the memory I/O.

Syntax

MemOn \{ bitNumber | memIOLabel \}

Parameters

bitNumber
   Integer expression representing memory I/O bits.

memIOLabel
   Memory I/O label.

Description

MemOn turns on the specified bit of the robot memory I/O. The 256 memory I/O bits are typically used as task communication status bits. The MemOn instruction turns the memory bit On, the MemOff instruction turns it Off, and the MemSw instruction is used to check the current state of the specified memory bit. The Wait instruction can also be used with the memory bit to cause the system to wait until a specified status is set.

Note

Memory outputs off

All memory I/O bits are turned off when the controller are restarted. They are not turned off by Emergency stop, safeguard open, program end, Reset command, or EPSON RC+ restart.

See Also

In, MemIn, InBCD, Off, MemOff, On, OpBCD, Oport, Out, MemOut, Sw, MemSw, Wait

MemOn Statement Example

The example shown below shows 2 tasks each with the ability to initiate motion instructions. However, a locking mechanism is used between the 2 tasks to ensure that each task gains control of the robot motion instructions only after the other task is finished using them. This allows 2 tasks to each execute motion statements as required and in an orderly predictable fashion. MemSw is used in combination with the Wait instruction to wait until the memory I/O #1 is the proper value before it is safe to move again. MemOn and MemOff are used to turn on and turn off the memory I/O for proper synchronization.

Function main
   Integer I
   MemOff 1
   Xqt 2, task2
   For i = 1 to 100
      Wait MemSw(1) = Off
      Go P(i)
      MemOn 1
   Next I
Fend

Function task2
   Integer I
   For i = 101 to 200
      Wait MemSw(1) = On
      Go P(i)
      MemOff 1
   Next I
Fend
Other simple examples from the command window are as follows:

```plaintext
> memon 1
> print memsw(1)
 1
> memoff 1
> print memsw(1)
 0
```
MemOut Statement

Simultaneously sets 8 memory I/O bits.

Syntax

MemOut portNumber, outData

Parameters

portNumber Integer expression representing memory I/O bit port number. The portNumber selection corresponds to the following:

<table>
<thead>
<tr>
<th>Portnum</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0-7</td>
</tr>
<tr>
<td>1</td>
<td>8-15</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

outData Integer expression between 0 and 255 representing the output pattern for the output group selected by portNumber. If represented in hexadecimal form the range is from &H0 to &HFF. The lower digit represents the least significant digits (or the 1st 4 outputs) and the upper digit represents the most significant digits (or the 2nd 4 outputs).

Description

MemOut simultaneously sets 8 memory I/O bits using the combination of the portNumber and outData values specified by the user to determine which outputs will be set. The portNumber parameter specifies which group of 8 outputs to use where portNumber = 0 means outputs 0 to 7, portNumber = 1 means outputs 8 to 15, etc.

Once a portNumber is selected, a specific output pattern must be defined. This is done using the outData parameter. The outData parameter may have a value between 0 and 255 and may be represented in hexadecimal or integer format. (i.e. &H0 to &HFF or 0 to 255)

The table below shows some of the possible I/O combinations and their associated outData values assuming that portNumber is “0”, and “1” accordingly.

<table>
<thead>
<tr>
<th>Output Settings When portNumber=0 (Output number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutData Value</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>01</td>
</tr>
<tr>
<td>02</td>
</tr>
<tr>
<td>03</td>
</tr>
<tr>
<td>08</td>
</tr>
<tr>
<td>09</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>99</td>
</tr>
<tr>
<td>255</td>
</tr>
</tbody>
</table>
MemOut Statement

Output Settings When portNumber=1 (Output number)

<table>
<thead>
<tr>
<th>OutData Value</th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>02</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>03</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>08</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>09</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>10</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>11</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>99</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

See Also
In, MemIn, InBCD, MemOff, MemOn, MemSw, Off, On, OpBCD, Oport, Out, Sw, Wait

MemOut Statement Example
The example below shows main task starting a background task called “iotask”. The “iotask” is a simple task to toggle memory I/O bits from 0 to 3 On and Off. The MemOut instruction makes this possible using only 1 command rather than turning each memory I/O bit on and off individually.

Function main
    Xqt 2, iotask
    Go P1
    .
    .
Fend

Function iotask
    Do
        MemOut 0, &H
        Wait 1
        MemOut 0, &H0
        Wait 1
    Loop
Fend

Other simple examples from the command window are as follows:

> MemOut 1,6  'Turns on memory I/O bits 9 & 10
> MemOut 2,1  'Turns on memory I/O bit 8
> MemOut 3,91  'Turns on memory I/O bits 24, 25, 27, 28, and 30
MemOutW Statement

Simultaneously sets 16 memory I/O bits.

Syntax

MemOutW wordPortNum, outputData

Parameters

wordPortNum  Integer expression representing memory I/O words.
outputData   Specifies output data (integers from 0 to 65535) using an expression or numeric value.

Description

Changes the current status of memory I/O port group specified by the word port number to the specified output data.

See Also

MemIn, MemInW, MemOut

MemOutW Statement Example

   MemOutW 0, 25
MemSw Function

Returns the status of the specified memory I/O bit.

Syntax

MemSw(bitNumber)

Parameters

bitNumber Integer expression representing the memory I/O bit number.

Return Values

Returns “1” when the specified bit is On and “0” when the specified bit is Off.

Description

MemSw returns the status of one memory I/O bit. Valid entries for MemSw range from bit 0 to bit 511. MemOn turns the specified bit on and MemOff turns the specified bit Off.

See Also

In, MemIn, InBCD, MemOff, MemOn, MemOut, Off, On, OpBCD, Oport, Out, Sw, Wait

MemSw Function Example

The example shown below shows 2 tasks each with the ability to initiate motion instructions. However, a locking mechanism is used between the 2 tasks to ensure that each task gains control of the robot motion instructions only after the other task is finished using them. This allows 2 tasks to each execute motion statements as required and in an orderly predictable fashion. MemSw is used in combination with the Wait instruction to wait until the memory I/O bit 1 is the proper value before it is safe to move again.

Function main
    Integer I
    MemOff 1
    Xqt 2, task2
    For i = 1 to 100
        Wait MemSw(1) = Off
        Go P(i)
        MemOn 1
    Next I
    Fend

Function task2
    Integer I
    For i = 101 to 200
        Wait MemSw(1) = On
        Go P(i)
        MemOff 1
    Next I
    Fend

Other simple examples from the Command window are as follows:

> memon 1
> print memsw(1)
1
> memoff 1
> print memsw(1)
0
MHour Function

Returns the accumulated MOTOR ON time of the robot motors.

Syntax

\[ \text{MHour ([robotNumber])} \]

Parameters

- **robotNumber**: Specify the robot number to check the MOTOR ON time by an integer value. If omitted, currently selected robot will be used.

Return Values

Returns the accumulated MOTOR ON time of the motors by an integer value.

See Also

- Time, Hour

MHour Function Example

```
Robot 2
Print MHour
Print MHour(1)
```
Mid$ Function

Returns a substring of a string starting from a specified position.

Syntax

Mid$(string, position [, count])

Parameters

- `string`  
  Source string expression.

- `position`  
  The starting position in the character string for copying `count` characters.

- `count`  
  Optional. The number of characters to copy from `string` starting with the character defined by `position`. If omitted, then all characters from `position` to the end of the string are returned.

Return Values

Returns a substring of characters from `string`.

Description

Mid$ returns a substring of as many as `count` characters starting with the `position` character in `string`.

See Also

Asc, Chr$, InStr, Left$, Len, Right$, Space$, Str$, Val

Mid$ Function Example

The example shown below shows a program that extracts the middle 2 characters from the string “ABCDEFGHIJ” and the remainder of the string starting at position 5.

```plaintext
Function midtest
  String basestr$, m1$, m2$
  basestr$ = "ABCDEFGHIJ"
  m1$ = Mid$(basestr$, (Len(basestr$) / 2), 2)
  Print "The middle 2 characters are: ", m1$
  m2$ = Mid$(basestr$, 5)
  Print "The string starting at 5 is: ", m2$
Fend
```
MkDir Statement

Creates a subdirectory on a controller disk drive.

Syntax

MkDir dirName

Parameters

dirName  String expression that defines the path and name of the directory to create.
See ChDisk for the details.

Description

Creates a subdirectory in the specified path. If omitted, a subdirectory is created in the current directory.

Note

- This statement is executable only with PC disk.

See Also

ChDir, ChDrive, RenDir, RmDir

MkDir Statement Example

The following examples are done from the command window:

  > MkDir \Data
  > MkDir \Data\PTS
  > MkDir \TEST1 \TEST2
Mod Operator

Returns the remainder obtained by dividing a numeric expression by another numeric expression.

Syntax

\[ \text{number} \ MOD \ divisor \]

Parameters

- **number**: The number being divided (the dividend).
- **divisor**: The number which **number** is divided by.

Return Values

Returns the remainder after dividing **number** by **divisor**.

Description

Mod is used to get the remainder after dividing 2 numbers. The remainder is a whole number. One clever use of the Mod instruction is to determine if a number is odd or even. The method in which the Mod instruction works is as follows: **number** is divided by **divisor**. The remainder left over after this division is then the return value for the Mod instruction.

See Also

Abs, Atan, Atan2, Cos, Int, Not, Sgn, Sin, Sqr, Str$, Tan, Val

Mod Operator Example

The example shown below determines if a number (var1) is even or odd. When the number is even the result of the Mod instruction will return “0”. When the number is odd, the result of the Mod instruction will return “1”.

```epson
Function modtest
  ....Integer var1, result
  ....Print "Enter an integer number:"
  ....Input var1
  ....result = var1 \ MOD \ 2
  ....Print "Result = ", result
  ....If result = 0 Then
  ........Print "The number is EVEN"
  ....Else
  ........Print "The number is ODD"
  ....EndIf
End
```

Some other example results from the Mod instruction from the Command window.

```epson
> Print 36 \ MOD \ 6
> 0

> Print 25 \ MOD \ 10
> 5
> ```
Motor Statement

Turns motor power for all axes on or off for the current robot.

Syntax

Motor ON | OFF

Parameters

ON | OFF  The keyword ON is used to turn the Motor Power on. The keyword OFF is used to turn Motor Power Off.

Description

The Motor On command is used to turn Motor Power On and release the brakes for all axes. Motor Off is used to turn Motor Power Off and set the brakes.

In order to move the robot, motor power must be turned on.

After an emergency stop, or after an error has occurred that requires resetting with the Reset command, execute Reset, and then execute Motor On.

Motor On sets the robot control parameter as below:

- Speed, SpeedR, SpeedS Default values
- Accel, AccelR, AccelS Default values
- QPDecelR, QPDecelS Default values
- LimZ 0
- CP Off
- SoftCP Off
- Fine Default values
- Power Low Low
- PTPBoost Default values
- TCLim, TCSpeed Default values
- PgLSpeed Default values

See Also

Brake, Power, Reset, SFree, SLock

Motor Statement Example

The following examples are done from the command window:

  > Motor On

  > Motor Off
Motor Function

Returns status of motor power for the specified robot.

Syntax

\[ \text{Motor } [(\text{robotNumber})] \]

Parameters

\[ \text{robotNumber} \]

Specify the robot number to check the status by an integer value. If omitted, currently selected robot will be used.

Return Values

0 = Motors off, 1 = Motors on.

See Also

Motor

Motor Function Example

\[
\text{If Motor = Off Then} \\
\quad \text{Motor On} \\
\text{EndIf}
\]
Move Statement

Moves the arm from the current position to the specified point using linear interpolation (i.e. moving in a straight line) at a constant tool center point velocity.

Syntax

```
Move destination [ROT] [ECP] [CP] [searchExpr] [!]...!] [SYNC]
```

Parameters

- **destination**: The target destination of the motion using a point expression.
- **ROT**: Optional. Decides the speed/acceleration/deceleration in favor of tool rotation.
- **ECP**: Optional. External control point motion. This parameter is valid when the ECP option is enabled.
- **CP**: Optional. Specifies continuous path motion.
- **searchExpr**: Optional. A Till or Find expression.
  - **Till**: Till Sw(expr) = {On | Off}
  - **Find**: Find Sw(expr) = {On | Off}
- **!...!**: Optional. Parallel Processing statements can be added to execute I/O and other commands during motion.
- **SYNC**: Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Move moves the arm from the current position to `destination` in a straight line. Move coordinates all axes to start and stop at the same time. The coordinates of `destination` must be taught previously before executing the Move instruction. Acceleration and deceleration for the Move is controlled by the AccelS instruction. Speed for the move is controlled by the SpeedS instruction. If the SpeedS speed value exceeds the allowable speed for any joint, power to all four joint motors will be turned off, and the robot will stop.

Move uses the SpeedS speed value and AccelS acceleration and deceleration values. Refer to Using Move with CP below on the relation between the speed/acceleration and the acceleration/deceleration. If, however, the ROT modifier parameter is used, Move uses the SpeedR speed value and AccelR acceleration and deceleration values. In this case SpeedS speed value and AccelS acceleration and deceleration value have no effect.

Usually, when the move distance is “0” and only the tool orientation is changed, an error will occur. However, by using the ROT parameter and giving priority to the acceleration and the deceleration of the tool rotation, it is possible to move without an error. When there is not an orientational change with the ROT modifier parameter and movement distance is not “0”, an error will occur.

Also, when the tool rotation is large as compared to move distance, and when the rotation speed exceeds the specified speed of the manipulator, an error will occur. In this case, please reduce the speed or append the ROT modifier parameter to give priority to the rotational speed / acceleration / deceleration.

When ECP is used, the trajectory of the external control point corresponding to the ECP number specified by ECP instruction moves straight with respect to the tool coordinate system. In this case, the trajectory of tool center point does not follow a straight line.
The optional Till qualifier allows the user to specify a condition to cause the robot to decelerate to a stop prior to completing the Move. The condition specified is simply a check against one of the inputs. This is accomplished through using the Sw instruction. The user can check if the input is On or Off and cause the arm to stop based on the condition specified. This feature works almost like an interrupt where the Move is interrupted (stopped) once the Input condition is met. If the input condition is never met during the Move then the arm successfully arrives on the point specified by destination. For more information about the Till qualifier see the Till command.

Notes

Move Cannot

Move cannot execute range verificat ion of the trajectory prior to starting the move itself. Therefore, even for target positions that are within an allowable range, it is possible for the system to find a prohibited position along the way to a target point. In this case, the arm may abruptly stop which may cause shock and a servo out condition of the arm. To prevent this, be sure to perform range verifications at low speed prior to using Move at high speeds. In summary, even though the target position is within the range of the arm, there are some Moves which will not work because the arm cannot physically make it to some of the intermediate positions required during the Move.

Using Move with CP

The CP parameter causes the arm to move to destination without decelerating or stopping at the point defined by destination. This is done to allow the user to string a series of motion instructions together to cause the arm to move along a continuous path while maintaining a specific speed throughout all the motion. The Move instruction without CP always causes the arm to decelerate to a stop prior to reaching the point destination.

Proper Speed and Acceleration Instructions with Move

The SpeedS and AccelS instructions are used to specify the speed and acceleration of the manipulator during Move motion. Pay close attention to the fact that SpeedS and AccelS apply to linear and circular interpolated motion while point to point motion uses the Speed and Accel instructions.

Potential Errors

Attempt to Change Only Tool Orientation

Changing only tool orientation during the move is impossible. If this is attempted, an error will occur. In this case, use the ROT parameter.

Joint Overspeed Errors

When the motion requested results in the speed of one of the axes to exceed its maximum allowable speed an overspeed error occurs. In the case of a motor overspeed error, the robot arm is brought to a stop and servo power is turned off.
Attempt to Pass the Original Point (RS series)

It is impossible to operate the arm of RS series to pass near an original point. If attempted this, an overspeed error will occur. For the operation near an original point, take the following actions.
- Lower the speed of SpeedS
- Find a different path to prevent an original point
- Use PTP motion such as Go command instead of Move command.

See Also
AccelS, Arc, CP, Go, Jump, Jump3, Jump3CP, SpeedS, Sw, Till

Move Statement Example

The example shown below shows a simple point to point move between points P0 and P1 and then moves back to P0 in a straight line. Later in the program the arm moves in a straight line toward point P2 until input #2 turns on. If input #2 turns On during the Move, then the arm decelerates to a stop prior to arriving on point P2 and the next program instruction is executed.

```
Function movetest
  Home
  Go P0
  Go P1
  Move P0
  Move P2 Till Sw(2) = On
  If Sw(2) = On Then
    Print "Input #2 came on during the move and"
    Print "the robot stopped prior to arriving on"
    Print "point P2."
  Else
    Print "The move to P2 completed successfully."
    Print "Input #2 never came on during the move."
  EndIf
EndFunction
```

This example uses Move with CP. The diagram below shows arc motion which originated at the point P100 and then moves in a straight line through P101, at which time the arm begins to form an arc. The arc is then continued through P102 and on to P103. Next the arm moves in a straight line to P104 where it finally decelerates to a stop. Note that the arm doesn't decelerate between each point until its final destination of P104. The following function would generate such a motion.

```
Function CornerArc
  Go P100
  Move P101 CP  ' Do not stop at P101
  Arc P102, P103 CP  ' Do not stop at P103
  Move P104  ' Decelerate to stop at P104
EndFunction
```
_MsgBox Statement_

Displays a message in a dialog box and waits for the operator to choose a button.

**Syntax**

```plaintext
_MsgBox message$ [ , type ] [ , title$ ] [ , answer ]
```

**Parameters**

- `message$`  
  The message that will be displayed.

- `type`  
  Optional. A numeric expression that is the sum of values specifying the number and type of buttons to display, the icon style to use, the identity of the default button. EPSON RC+ 7.0 includes predefined constants that can be used for this parameter. The following table shows the values that can be used.

<table>
<thead>
<tr>
<th>Symbolic constant</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB_OK</td>
<td>0</td>
<td>Display OK button only.</td>
</tr>
<tr>
<td>MB_OKCANCEL</td>
<td>1</td>
<td>Display OK and cancel buttons.</td>
</tr>
<tr>
<td>MB_ABORTRETRYIGNORE</td>
<td>2</td>
<td>Display Abort, Retry, and Ignore buttons.</td>
</tr>
<tr>
<td>MB_YESNOCANCEL</td>
<td>3</td>
<td>Display Yes, No, and Cancel buttons.</td>
</tr>
<tr>
<td>MB_YESNO</td>
<td>4</td>
<td>Display Yes and No buttons.</td>
</tr>
<tr>
<td>MB_RETRYCANCEL</td>
<td>5</td>
<td>Display Retry and Cancel buttons.</td>
</tr>
<tr>
<td>MB_ICONSTOP</td>
<td>16</td>
<td>Stop sign.</td>
</tr>
<tr>
<td>MB_ICONQUESTION</td>
<td>32</td>
<td>Question mark.</td>
</tr>
<tr>
<td>MB_ICONEXCLAMATION</td>
<td>64</td>
<td>Exclamation mark.</td>
</tr>
<tr>
<td>MB_DEFBUTTON1</td>
<td>0</td>
<td>First button is default.</td>
</tr>
<tr>
<td>MB_DEFBUTTON2</td>
<td>256</td>
<td>Second button is default.</td>
</tr>
</tbody>
</table>

- `title$`  
  Optional. String expression that is displayed in the title bar of the message box.

- `answer`  
  Optional. An integer variable that receives a value indicating the action taken by the operator. EPSON RC+ 6.0 includes predefined constants that can be used for this parameter. The table below shows the values returned in `answer`.

<table>
<thead>
<tr>
<th>Symbolic constant</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDOK</td>
<td>1</td>
<td>OK button selected.</td>
</tr>
<tr>
<td>IDCANCEL</td>
<td>2</td>
<td>Cancel button selected.</td>
</tr>
<tr>
<td>IDABORT</td>
<td>3</td>
<td>Abort button selected.</td>
</tr>
<tr>
<td>IDRETRY</td>
<td>4</td>
<td>Retry button selected.</td>
</tr>
<tr>
<td>IDYES</td>
<td>6</td>
<td>Yes button selected.</td>
</tr>
<tr>
<td>IDNO</td>
<td>7</td>
<td>No button selected.</td>
</tr>
</tbody>
</table>

**Description**

_MsgBox automatically formats the message. If you want blank lines, use CRLF in the message. See the example._

**See Also**

- InputBox
**MsgBox Statement Example**

This example displays a message box that asks the operator if he/she wants to continue or not. The message box will display two buttons: Yes and No. A question mark icon will also be displayed. After MsgBox returns (after the operator clicks a button), then the answer is examined. If it's no, then all tasks are stopped with the Quit command.

```plaintext
Function msgtest
    String msg$, title$
    Integer mFlags, answer

    msg$ = Chr$(34) + "Operation complete" + Chr$(34) + CRLF
    msg$ = msg$ + "Ready to continue?"
    title$ = "Sample Application"
    mFlags = MB_YESNO + MB_ICONQUESTION
    MsgBox msg$, mFlags, title$, answer
    If answer = IDNO then
        Quit All
    EndIf
EndFunction
```

A picture of the message box that this code will create is shown below.
MyTask Function

Returns the task number of the current program.

Syntax

MyTask

Return Values

The task number of the current task. Valid entries are as below:

<table>
<thead>
<tr>
<th>Type</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal task</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Background tasks</td>
<td>65 to 80</td>
</tr>
<tr>
<td>Trap tasks</td>
<td>257 to 267</td>
</tr>
</tbody>
</table>

Description

MyTask returns the task number of the current program with a numeral. The MyTask instruction is inserted inside a specific program and when that program runs the MyTask function will return the task number that the program is running in.

See Also

Xqt

MyTask Function Example

The following program switches On and Off the I/O ports from 1 to 8.

```
Function main
  Xqt 2, task  'Execute task 2.
  Xqt 3, task  'Execute task 3.
  Xqt 4, task  'Execute task 4.
  Xqt 5, task  'Execute task 5.
  Xqt 6, task  'Execute task 6.
  Xqt 7, task  'Execute task 7.
  Xqt 8, task  'Execute task 8.
Call task
Fend

Function task
  Do
    On MyTask  'Switch On I/O port which has the same number as current task number
    Off MyTask 'Switch Off I/O port which has the same number as current task number
  Loop
Fend
```
The For/Next instructions are used together to create a loop where instructions located between the For and Next instructions are executed multiple times as specified by the user.

**Syntax**

For \[ var1 = initval \text{ To } finalval \ [\text{Step Increment}] \]

Next \[ var1 \]

**Parameters**

- \[ var1 \]
  - The counting variable used with the For/Next loop. This variable is normally defined as an integer but may also be defined as a Real variable.
- \[ initval \]
  - The initial value for the counter \[ var1 \].
- \[ finalval \]
  - The final value of the counter \[ var1 \]. Once this value is met, the For/Next loop is complete and execution continues starting with the statement following the Next instruction.
- \[ Increment \]
  - An optional parameter which defines the counting increment for each time the Next statement is executed within the For/Next loop. This variable may be positive or negative. However, if the value is negative, the initial value of the variable must be larger than the final value of the variable. If the increment value is left out the system automatically increments by 1.
- \[ statements \]
  - Any valid SPEL+ statements can be inserted inside the For/Next loop.

**Return Values**

None

**Description**

For/Next executes a set of statements within a loop a specified number of times. The beginning of the loop is the For statement. The end of the loop is the Next statement. A variable is used to count the number of times the statements inside the loop are executed.

The first numeric expression (\[ initval \]) is the initial value of the counter. This value may be positive or negative as long as the \[ finalval \] variable and Step increment correspond correctly.

The second numeric expression (\[ finalval \]) is the final value of the counter. This is the value which once reached causes the For/Next loop to terminate and control of the program is passed on to the next instruction following the Next instruction.

Program statements after the For statement are executed until a Next instruction is reached. The counter variable (\[ var1 \]) is then incremented by the Step value defined by the \[ increment \] parameter. If the Step option is not used, the counter is incremented by one.

The counter variable (\[ var1 \]) is then compared with the final value (\[ finalval \]). If the counter is less than or equal to the final value (\[ finalval \]), the statements following the For instruction are executed again. If the counter variable is greater than the final value (\[ finalval \]), execution branches outside of the For/Next loop and continues with the instruction immediately following the Next instruction.

Nesting of For/Next statements is supported up to 10 levels deep. This means that a For/Next Loop can be put inside of another For/Next loop and so on and so on until there are 10 "nests" of For/Next loops.
Note

**Negative Step Values**

If the value of the Step increment (*increment*) is negative, the counter variable (var1) is decremented (decreased) each time through the loop and the initial value (*initval*) must be greater than the final value (*finalval*) for the loop to work.

See Also

For

**Next Statement Example**

```plaintext
Function fornext
    Integer ctr
    For ctr = 1 to 10
        Go Pctr
    Next ctr
    ,
    For ctr = 10 to 1 Step -1
        Go Pctr
    Next ctr
Fend
```
Not Operator

Performs the bitwise complement on the value of the operand.

Syntax
   Not operand

Parameters
   operand  Integer expression.

Return Values
   1’s complement of the value of the operand.

Description
   The Not function performs the bitwise complement on the value of the operand. Each bit of the result is the complement of the corresponding bit in the operand, effectively changing 0 bits to 1, and 1 bits to 0.

See Also
   Abs, And, Atan, Atan2, Cos, Int, LShift, Mod, Or, RShift, Sgn, Sin, Sqr, Str$, Tan, Val, Xor

Not Operator Example
   This is a simple Command window example on the usage of the Not instruction.

   >print not(1)
   -2
   >
Off Statement

Turns Off the specified output and after a specified time can turn it back on.

Syntax

\[
\text{Off} \{ \text{bitNumber} \mid \text{outputLabel} \}, [\text{time}], [\text{parallel}][\text{Forced}]
\]

Parameters

- \text{bitNumber}: Integer expression representing which Output to turn Off.
- \text{outputLabel}: Output label.
- \text{time}: Optional. Specifies a time interval in seconds for the output to remain Off. After the time interval expires, the Output is turned back on. The minimum time interval is 0.01 seconds and maximum time interval is 10 seconds.
- \text{parallel}: Optional. When a timer is set, the parallel parameter may be used to specify when the next command executes:
  - 0 - immediately after the output is turned off
  - 1 - after the specified time interval elapses. (default value)
- \text{Forced}: Optional. Usually omitted.

Description

Off turns off (sets to 0) the specified output.

If the \text{time} interval parameter is specified, the output bit specified by \text{bitNumber} is switched off, and then switched back on after the \text{time} interval elapses. If prior to executing Off, the Output bit was already off, then it is switched On after the time interval elapses.

The \text{parallel} parameter settings are applicable when the time interval is specified as follows:

- 1: Switches the output off, switches it back on after specified interval elapses, then executes the next command. (This is also the default value for the parallel parameter. If this parameter is omitted, this is the same as setting the parameter to “1”.)
- 0: Switches the output off, and simultaneously executes the next command.

Notes

Output bits Configured as Remote Control output

If an output bit which was set up as a system output is specified, an error will occur. Remote control output bits are turned on or off automatically according to system status.

Outputs and When an Emergency Stop Occurs:

EPSON RC+ has a feature which causes all outputs to go off when an E-Stop occurs. If you want to keep the settings even in case of the emergency stop, this feature can be reconfigured from the [Outputs Off during emergency stop] checkbox in the [Setup]-[System Configuration]-[Controller]-[Preferences].

Forced Flag

This flag is used to turn Off the I/O output at Emergency Stop and Safety Door Open from NoPause task or NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt).

Be sure that the I/O outputs change by Emergency Stop and Safety Door Open when designing the system.
See Also
In, InBCD, MemOn, MemOff, MemOut, MemSw, OpBCD, Oport, Out, Wait

Off Statement Example
The example shown below shows main task start a background task called “iotask”. The “iotask” is a simple task to turn discrete output bits 1 and 2 on and then off, Wait 10 seconds and then do it again.

Function main
  Xqt 2, iotask
  Go P1
  .
  .
  .
Fend

Function iotask
  Do
  On 1
  On 2
  Off 1
  Off 2
  Wait 10
  Loop
Fend

Other simple examples from the Command window are as follows:

> on 1
> off 1, 10 ' Turn Output 1 off, wait 10 seconds, turn on again
> on 2
> off 2
OLAccel Statement

Sets up the automatic adjustment of acceleration/deceleration that is adjusted according to the overload rating.

Syntax
OLAccel {On | Off}

Parameters
On | Off
On: Enables the automatic adjustment of acceleration/deceleration that is adjusted according to the overload rating.
Off: Disables the automatic adjustment of acceleration/deceleration that is adjusted according to the overload rating.

Description
OLAccel can be used to enable the automatic adjustment function of acceleration and deceleration that is adjusted according to the robot loading rate (OLRate). When OLAncel is On, the acceleration and deceleration are automatically adjusted in accordance with the robot loading rate at PTP motion commands. This is done to prevent the over load error by reducing the acceleration/deceleration automatically when the loading rate is exceeding a certain value at PTP motion. Heretofore, when users were executing motion with heavy duty that may cause over load error, users had to stop the robot by the program or adjust the speed and acceleration to prevent the error. OLAncel statement lessens these measures. However, this statement do not prevent over load error at all types of cycles. When the cycle has very heavy duty and load, the over load error may occur. In this case, users need to stop the robot or adjust the speed and acceleration. In some operation environment, the motor temperature may rise by operating the robot without over load error and result in over heat error.

This statement is unnecessary at proper load operation.
Use OLRate in the test cycle to check whether the over load error may occur or not.

The OLAncel value initializes to the default values (low acceleration) when any one of the following conditions occurs:

```
Controller Startup
Motor On
SFree, SLock, Brake
Reset, Reset Error
Stop button or QuitAll stops tasks
```

Note
If OLAncel On is executed to a robot that does not support the automatic adjustment function of acceleration and deceleration, an error occurs.

See Also
OLAccel Function, OLRate
OLAccel Statement Example

>olaccel on
>olaccel
OLACCEL is ON

Function main
Motor On
Power High
Speed 100
Accel 100, 100
OLAccel On
Xqt 2, MonitorOLRate
Do
    Jump P0
    Jump P1
Loop
Fend

Function MonitorOLRate
Do
    'Displays OLRate
    OLRate
    Wait 1
Loop
Fend
OLAccel Function

Returns the automatic adjustment setting.

Syntax

```
OLAccel
```

Return Values

- **Off** = Automatic adjustment of acceleration/deceleration that is adjusted according to the overload rating is disabled.
- **On** = Automatic adjustment of acceleration/deceleration that is adjusted according to the overload rating is enabled.

See Also

- OLAccel, OLRate

OLAccel Function Example

```
If OLAccel = Off Then
  Print "OLAccel is off"
EndIf
```
OLRate Statement

Display overload rating for one or all joints for the current robot.

**Syntax**

\[ \text{OLRate}[\text{jointNumber}] \]

**Parameters**

- **jointNumber**
  - Integer expression from 1 to 9.
  - The additional S axis is 8 and T axis is 9.

**Description**

OLRate can be used to check whether a cycle is causing stress on the servo system. Factors such as temperature and current can cause servo errors during applications with high duty cycles. OLRate can help to check if the robot system is close to having a servo error.

During a cycle, run another task to command OLRate. If OLRate exceeds 1.0 for any joint, then a servo error will occur.

Servo errors are more likely to occur with heavy payloads. By using OLRate during a test cycle, you can help insure that the speed and acceleration settings will not cause a servo error during production cycling.

To get valid readings, you must execute OLRate while the robot is moving.

**See Also**

- OLRate Function

**OLRate Statement Example**

```plaintext
> olrate
0.10000 0.20000
0.30000 0.40000
0.50000 0.60000

Function main
    Power High
    Speed 50
    Accel 50, 50
    Xqt 2, MonitorOLRate
    Do
        Jump P0
        Jump P1
    Loop
    Fend

Function MonitorOLRate
    Do
        OLRate ' Display OLRate
        Wait 1
    Loop
    Fend
```
OLRate Function

Returns overload rating for one joint for the current robot.

Syntax

\[
\text{OLRate}(\text{jointNumber})
\]

Parameters

\(\text{jointNumber} \quad \text{Integer expression from 1 to 9.}
\)
\(\quad \text{The additional S axis is 8 and T axis is 9.}
\)

Return Values

Returns the OLRate for the specified joint. Values are between 0.0 and 2.0.

Description

OLRate can be used to check whether a cycle is causing stress on the servo system. Factors such as temperature and current can cause servo errors during applications with high duty cycles. OLRate can help to check if the robot system is close to having a servo error.

During a cycle, run another task to command OLRate. If OLRate exceeds 1.0 for any joint, then a servo error will occur.

Servo errors are more likely to occur with heavy payloads. By using OLRate during a test cycle, you can help insure that the speed and acceleration settings will not cause a servo error during production cycling.

To get valid readings, you must execute OLRate while the robot is moving.

See Also

OLRate

OLRate Function Example

Function main

\[
\begin{align*}
\text{Power High} \\
\text{Speed 50} \\
\text{Accel 50, 50} \\
\text{Xqt 2, MonitorOLRate} \\
\text{Do} \\
\text{\quad Jump P0} \\
\text{\quad Jump P1} \\
\text{\quad Loop} \\
\text{\quad Fend}
\end{align*}
\]

Function MonitorOLRate

\[
\begin{align*}
\text{Integer i} \\
\text{Real olRates(4)} \\
\text{Do} \\
\text{\quad For i = 1 to 4} \\
\text{\quad \quad olRates(i) = OLRate(i)} \\
\text{\quad \quad If olRate(i) > .5 Then} \\
\text{\quad \quad \quad Print "Warning: OLRate(", i, ") is over .5"} \\
\text{\quad \quad EndIf} \\
\text{\quad \quad Next i} \\
\text{\quad \quad Loop} \\
\text{\quad \quad Fend}
\end{align*}
\]
On Statement

Turns on the specified output and after a specified time can turn it back off.

**Syntax**

```plaintext
On { bitNumber | outputLabel }, [ time ], [ parallel ] [ Forced ]
```

**Parameters**

- `bitNumber` Integer expression representing which Output to turn On.
- `outputLabel` Output label.
- `time` Optional. Specifies a time interval in seconds for the output to remain On. After the time interval expires, the Output is turned back off. (Minimum time interval is 0.01 seconds)
- `parallel` Optional. When a timer is set, the parallel parameter may be used to specify when the next command executes:
  - 0 - immediately after the output is turned on
  - 1 - after the specified time interval elapses. (default value)
- `Forced` Optional. Usually omitted.

**Description**

On turns On (sets to 1) the specified output. If the `time` interval parameter is specified, the output bit specified by `outnum` is switched On, and then switched back Off after the `time` interval elapses.

The `parallel` parameter settings are applicable when the time interval is specified as follows:

- 1: Switches the output On, switches it back Off after specified interval elapses, then executes the next command. (This is also the default value for the parallel parameter. If this parameter is omitted, this is the same as setting the parameter to “1”.)
- 0: Switches the output On, and simultaneously executes the next command.

**Notes**

**Output bits Configured as remote**

If an output bit which was set up as remote is specified, an error will occur. Remote output bits are turned ON or OFF automatically according to system status. For more information regarding remote, refer to *EPSON RC+ User’s Guide*. The individual bits for the remote connector can be set as remote or I/O from [Setup]-[System Configuration]-[Controller]-[Remote Control] panel.

**Outputs and When an Emergency Stop Occurs**

The Controller has a feature which causes all outputs to go off when an E-Stop occurs. If you want to keep the settings even in case of the emergency stop, this feature can be reconfigured from the [Outputs Off during emergency stop] checkbox in [Setup]-[System Configuration]-[Controller]-[Preferences].

**Forced Flag**

This flag is used to turn On the I/O output at Emergency Stop and Safety Door Open from NoPause task, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), or background tasks. Be sure that the I/O outputs change by Emergency Stop and Safety Door Open when designing the system.
On Statement

See Also
In, InBCD, MemOff, MemOn, Off, OpBCD, Oport, Out, Wait

On Statement Example
The example shown below shows main task start a background task called “iotask”. The “iotask” is a simple task to turn discrete output bits 1 and 2 on and then off, Wait 10 seconds and then do it again.

```plaintext
Function main
   Xqt iotask
   Go P1
   .
   .
   .
Fend

Function iotask
   Do
      On 1
      On 2
      Off 1
      Off 2
      Wait 10
   Loop
Fend

Other simple examples from the command window are as follows:

> on 1
> off 1, 10  ' Turn Output 1 off, wait 10 seconds, turn on again
> on 2
> off 2
```
OnErr Statement

Sets up interrupt branching to cause control to transfer to an error handing subroutine when an error occurs. Allows users to perform error handling.

Syntax

-OnErr GoTo {label | 0}

Parameters

<table>
<thead>
<tr>
<th>label</th>
<th>Statement label to jump to when an error occurs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Parameters used to clear OnErr setting.</td>
</tr>
</tbody>
</table>

Description

OnErr enables user error handling. When an error occurs without OnErr being used, the task is terminated and the error is displayed. However, when OnErr is used it allows the user to "catch" the error and go to an error handler to automatically recover from the error. Upon receiving an error, OnErr branches control to the designated label specified in the EResume instruction. In this way the task is not terminated and the user is given the capability to automatically handle the error. This makes work cells run much smoother since potential problems are always handled and recovered from in the same fashion.

When the OnErr command is specified with the “0” parameter, the current OnErr setting is cleared. (i.e. After executing OnErr 0, if an error occurs program execution will stop)

See Also

Err, EResume

OnErr Statement Example

The following example shows a simple utility program which checks whether points P0-P399 exist. If the point does not exist, then a message is printed on the screen to let the user know this point does not exist. The program uses the CX instruction to test each point for whether or not it has been defined. When a point is not defined control is transferred to the error handler and a message is printed on the screen to tell the user which point was undefined.

```
Function errDemo
    Integer i, errNum

    OnErr GoTo errHandler

    For i = 0 To 399
        temp = CX(P(i))
        Next i
    Exit Function

    '*********************************************
    '* Error Handler                             *
    '*********************************************
    errHandler:
        errNum = Err
        ' Check if using undefined point
        If errNum = 7007 Then
            Print "Point number P", i, " is undefined!"
        Else
            Print "ERROR: Error number ", errNum, " occurred while trying to process point P", i, " !"
        EndIf
    EResume Next
EndFunction
```
OpBCD Statement

Simultaneously sets 8 output lines using BCD format. (Binary Coded Decimal)

Syntax

OpBCD  \textit{portNumber}, \textit{outData} [, \textit{Forced}]

Parameters

\textit{portNumber}  Integer expression representing I/O output bytes. Where the \textit{portNumber} selection corresponds to the following outputs:

<table>
<thead>
<tr>
<th>PortNumber</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0-7</td>
</tr>
<tr>
<td>1</td>
<td>8-15</td>
</tr>
<tr>
<td>2</td>
<td>16-23</td>
</tr>
<tr>
<td>3</td>
<td>24-31</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\textit{outData}  Integer expression between 0 and 99 representing the output pattern for the output group selected by \textit{portNumber}. The 2nd digit (called the 1's digit) represents the lower 4 outputs in the selected group and the 1st digit (called the 10's digit) represents the upper 4 outputs in the selected group.

\textit{Forced}  Optional. Usually omitted.

Description

OpBCD simultaneously sets 8 output lines using the BCD format. The standard and expansion user outputs are broken into groups of 8. The \textit{portNumber} parameter for the OpBCD instruction defines which group of 8 outputs to use where \textit{portNumber} = 0 means outputs 0 to 7, \textit{portNumber} = 1 means outputs 8 to 15, etc.

Once a port number is selected (i.e. a group of 8 outputs has been selected), a specific output pattern must be defined. This is done in Binary Coded Decimal format using the \textit{outData} parameter. The outdata parameter may have 1 or 2 digits. (Valid entries range from 0 to 99.) The 1st digit (or 10's digit) corresponds to the upper 4 outputs of the group of 8 outputs selected by \textit{portNumber}. The 2nd digit (or 1's digit) corresponds to the lower 4 outputs of the group of 8 outputs selected by \textit{portNumber}.

Since valid entries in BCD format range from 0 to 9 for each digit, every I/O combination cannot be met. The table below shows some of the possible I/O combinations and their associated \textit{outnum} values assuming that \textit{portNumber} is 0.

<table>
<thead>
<tr>
<th>Output Settings (Output number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textbf{Outnum Value}</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>01</td>
</tr>
<tr>
<td>02</td>
</tr>
<tr>
<td>03</td>
</tr>
<tr>
<td>08</td>
</tr>
<tr>
<td>09</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>99</td>
</tr>
</tbody>
</table>

Note that the Binary Coded Decimal format only allows decimal values to be specified. This means that through using Binary Coded Decimal format it is impossible to turn on all outputs with the OpBCD instruction. Please note that the maximum value for either digit for outnum is “9”. This means that the largest value possible to use with OpBCD is “99”. In the table above it is easy to see that “99” does not turn all Outputs on. Instead it turns outputs 0, 3, 4, and 7 On and all the others off.
Notes

Difference between OpBCD and Out

The OpBCD and Out instructions are very similar in the SPEL+ language. However, there is one major difference between the two. This difference is shown below:

- The OpBCD instruction uses the Binary Coded Decimal format for specifying an 8 bit value to use for turning the outputs on or off. Since Binary Coded Decimal format precludes the values of &HA, &HB, &HC, &HD, &HE or &HF from being used, all combinations for setting the 8 output group cannot be satisfied.
- The Out instruction works very similarly to the OpBCD instruction except that Out allows the range for the 8 bit value to use for turning outputs on or off to be between 0 and 255 (0 to 99 for OpBCD). This allows all possible combinations for the 8 bit output groups to be initiated according to the users specifications.

Output bits Configured as Remote:

If an output bit which was set up as remote is specified to be turned on by OpBCD, an error will occur. Remote output bits are turned On or Off automatically according to system status. For more information regarding remote, refer to EPSON RC+ User’s Guide. The individual bits for the remote connector can be set as remote or I/O from [Setup]-[System Configuration]-[Controller]-[Remote Control] panel.

Outputs and When an Emergency Stop Occurs:

The Controller has a feature which causes all outputs to go off when an E-Stop occurs. This feature is set or disabled from the [Outputs Off during emergency stop] checkbox in the [Setup]-[System Configuration]-[Controller]-[Preferences].

Forced Flag

This flag is used to turn On the I/O output at Emergency Stop and Safety Door Open from NoPause task, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), or background tasks. Be sure that the I/O outputs change by Emergency Stop and Safety Door Open when designing the system.

See Also

In, InBCD, MemOff, MemOn, MemSw, Off, On, Oport, Out, Sw, Wait

OpBCD Function Example

The example shown below shows main task start a background task called “iotask”. The “iotask” is a simple task to flip flop between turning outputs 1 & 2 on and then outputs 0 and 3 on. When 1 & 2 are turned on, then 0 & 3 are also turned off and vice versa.

Function main
  Xqt 2, iotask
  Go P1
  .
  .
Fend

Function iotask
  Do
    OpBCD 0, 6
    OpBCD 0, 9
    Wait 10
  Loop
Fend

Other simple examples from the command window are as follows:

> OpBCD 1, 6 'Turns on Outputs 1 and 2
> OpBCD 2, 1 'Turns on Output 8
> OpBCD 3, 91 'Turns on Output 24, 28, and 31
OpenDB Statement

Opens a database or Excel workbook.

Syntax

OpenDB #fileNumber, { SQL | Accel | Excel }, [ DBserverName As String ],
{ DBname As String | filename As String }

Parameters

fileNumber
Integer number from 501 to 508

SQL | Accel | Excel
Selects a database type you want to open from [SQL], [Access], and [Excel].

DBserverName
If you select [SQL], the SQL server name is specified.
If omitted, LOCAL server is specified. The SQL server on the network cannot be specified.
If you select [Access] or [Excel], the SQL server name is not specified.

DBname | filename
If you select [SQL] as a database, a database name on the SQL server is specified.
If you select [Access], Access file name is specified.
If omitted the path of Access file name, it searches in the current folder.
See ChDisk for the details.
If you select [Excel], Excel file name is specified.
If you omitted Excel file name, it searches in the current folder.
See ChDisk for the details.

Description

Opens the specified database using the specified file number.
The specified database must exist on the disk of PC with installed RC+. Otherwise, it causes an error. The specified file number can be used to identify the database while it is open, but cannot be used to refer to the different database until you close the database with the CloseDB command. The file number is used with the database operation commands (SelectDB, Print#, Input#, CloseDB).
Access and Excel files of Microsoft office 2010 64-bit cannot be used.

Note
- Connection of PC with installed RC+ is required.

See Also
SelectDB, CloseDB, UpdateDB, DeleteDB, Input #, Print #

OpenDB Statement Example

Using the SQL database
The following example uses the SQL server 2000 sample database, Northwind and loads the data from a table.

```EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9```

```EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9```
Using Access database
The following example uses Microsoft Access 2007 sample database “Students” and loads the data from a table.

```
Integer count, i, eid
String Lastname$, Firstname$, dummy$

OpenDB #502, Access, "c:\MyDataBase\Students.accdb"
count = SelectDB(#502, "Students")
For i = 0 To count - 1
    Input #502, eid, dummy$, Lastname$, Firstname$
    Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #502
```

Using Excel workbook
The following example uses Microsoft Excel workbook “StudentsList“ and loads the data from a sheet.

```
Integer count, i, eid
String Lastname$, Firstname$

OpenDB #503, Excel, "c:\MyDataBase\Students.xls"
count = SelectDB(#503, "[Students$]")
For i = 0 To count - 1
    Input #503, eid, Lastname$, Firstname$
    Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #503
```
OpenCom Statement

Open an RS-232 communication port.

Syntax

```
OpenCom #portNumber
```

Parameters

- `portNumber`: Integer expression for RS-232C port number to open.
  - The range of port number is:
    - Real Part: 1 to 8
    - Windows Part: 1001 to 1008

Description

You need to connect the specified RS-232C port to the controller.

To use the SPEL+ real part ports, option I/O board must be installed to the Controller.
To use Windows part ports, RC+ setting must be done. For details, refer to the description about RC-232C in the *EPSON RC+ User’s Guide 5.12 [Setup] Menu*.

See Also

- ChkCom, CloseCom, SetCom

OpenCom Statement Example

```plaintext
Integer PortNo

PortNo = 1001
OpenCom #PortNo
Print #PortNo, "Data from COM1"
CloseCom #PortNo
```
OpenCom Function

Acquires the task number that executes OpenCom.

Syntax

OpenCom (portNumber)

Parameters

portNumber Integer expression for RS-232C port number. The range of port number is:
Real Part 1 to 8
Windows Part 1001 to 1008

Description

Acquires the task number that executes OpenCom.

See Also

ChkCom, CloseCom, OpenCom, SetCom

OpenCom Function Example

Print OpenCom(PortNo)
OpenNet Statement

Open a TCP/IP network port.

Syntax

```
OpenNet #portNumber As { Client | Server }
```

Parameters

```
portNumber       Integer expression for TCP/IP port number to open. Range is from 201 to 216.
```

Description

OpenNet opens a TCP/IP port for communication with another computer on the network.
One system should open as Server and the other as Client. It does not matter which one executes first.

See Also

ChkNet, CloseNet, SetNet

OpenNet Statement Example

For this example, two controllers have their TCP/IP settings configured as follows:

**Controller #1:**

Port: #201
Host Name: 192.168.0.2
TCP/IP Port: 1000

```
Function tcpip
  OpenNet #201 As Server
  WaitNet #201
  Print #201, "Data from host 1"
Fend
```

**Controller #2:**

Port: #201
Host Name: 192.168.0.1
TCP/IP Port: 1000

```
Function tcpip
  String data$
  OpenNet #201 As Client
  WaitNet #201
  Input #201, data$
  Print "received ", data$, " from host 1"
Fend
```
OpenNet Function

Acquires the task number that executes OpenNet.

**Syntax**

```
OpenNet (portNumber)
```

**Parameters**

- `portNumber` Integer expression for TCP/IP port number. Range is from 201 to 216.

**Description**

Acquires the task number that executes OpenNet.

**See Also**

ChkNet, CloseNet, OpenNet, SetNet

**OpenNet Function Example**

```
Print OpenNet (PortNo)
```
Oport Function

Oport Function

Returns the state of the specified output.

Syntax

Oport(outnum)

Parameters

outnum      Integer expression representing I/O output bits.

Return Values

Returns the specified output bit status as either 0 or 1.

0: Off status
1: On status

Description

Oport provides a status check for the outputs. It functions much in the same way as the Sw instruction does for inputs. Oport is most commonly used to check the status of one of the outputs which could be connected to a feeder, conveyor, gripper solenoid, or a host of other devices which works via discrete I/O. Obviously the output checked with the Oport instruction has 2 states (1 or 0). These indicate whether the specified output is On or Off.

Note

Difference between Oport and Sw

It is very important for the user to understand the difference between the Oport and Sw instructions. Both instructions are used to get the status of I/O. However, the type of I/O is different between the two. The Sw instruction works inputs. The Oport instruction works with the standard and expansion hardware outputs. These hardware ports are discrete outputs which interact with devices external to the controller.

See Also

In, InBCD, MemIn, MemOn, MemOff, MemOut, MemSw, Off, On, OpBCD, Out, Sw, Wait
OPort Function Example
The example shown below turns on output 5, then checks to make sure it is on before continuing.

```spel
Function main
    TMOut 10
    OnErr errchk
    Integer errnum
    On 5    'Turn on output 5
    Wait Oport(5)
    Call mkpart1
    Exit Function

erchk:
    errnum = Err(0)
    If errnum = 94 Then
        Print "TIME Out Error Occurred during period"
        Print "waiting for Oport to come on. Check"
        Print "Output #5 for proper operation. Then"
        Print "restart this program."
    Else
        Print "ERROR number ", errnum, "Occurred"
        Print "Program stopped due to errors!"
    EndIf
    Exit Function
Fend
```

Other simple examples are as follows from the command window:

```
> On 1
> Print Oport(1)
1
> Off 1
> Print Oport(1)
0
> 
```
Or Operator

Performs a bitwise or logical OR operation on two operands.

Syntax

\[
expr1 \text{ Or } expr2
\]

Parameters

\[
expr1, expr2 \text{ Integer or Boolean expressions.}
\]

Return Values

Bitwise OR value of the operands if the expressions are integers. Logical OR if the expressions are Boolean.

Description

For integer expressions, the Or operator performs the bitwise OR operation on the values of the operands. Each bit of the result is 1 if one or both of the corresponding bits of the two operands is 1. For Boolean expressions, the result is True if either of the expressions evaluates to True.

See Also

And, LShift, Mod, Not, RShift, Xor

Or Operator Example

Here is an example of a bitwise OR.

\[
> \text{print 1 or 2}
\]

3

Here is an example of a logical OR.

\[
\text{If } a = 1 \text{ Or } b = 2 \text{ Then}
\]

\[
c = 3
\]

EndIf
Out Statement

Simultaneously sets 8 output bits.

Syntax

Out portNumber, outData [, Forced]

Parameters

portNumber Integer expression representing I/O output bytes. The portnum selection corresponds to the
following outputs:

<table>
<thead>
<tr>
<th>Portnum</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0-7</td>
</tr>
<tr>
<td>1</td>
<td>8-15</td>
</tr>
</tbody>
</table>

outData Integer number between 0 and 255 representing the output pattern for the output group selected
by portNumber. If represented in hexadecimal form the range is from &H0 to &HFF. The
lower digit represents the least significant digits (or the 1st 4 outputs) and the upper digit
represents the most significant digits (or the 2nd 4 outputs).

Forced Optional. Usually omitted.

Description

Out simultaneously sets 8 output lines using the combination of the portNumber and outdata values specified
by the user to determine which outputs will be set. The portNumber parameter defines which group of 8
outputs to use where portNumber = 0 means outputs 0 to 7, portNumber = 1 means outputs 8 to 15, etc.

Once a portnum is selected (i.e. a group of 8 outputs has be selected), a specific output pattern must be
defined. This is done using the outData parameter. The outData parameter may have a value between 0 to
255 and may be represented in Hexadecimal or Integer format. (i.e. &H0 to &HFF or 0 to 255)

The table below shows some of the possible I/O combinations and their associated outData values assuming
that portNumber is “0”, and “1” accordingly.

<table>
<thead>
<tr>
<th>OutData Value</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>02</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>03</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>08</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>09</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>10</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>11</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>99</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>
Output Settings When *portNumber*=1 (Output number)

<table>
<thead>
<tr>
<th>OutData Value</th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>02</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>03</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>08</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>09</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>10</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>11</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>99</td>
<td>Off</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>255</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
<td>On</td>
</tr>
</tbody>
</table>

**Notes**

**Difference between OpBCD and Out**

The Out and OpBCD instructions are very similar in the SPEL+ language. However, there is one major difference between the two. This difference is shown below:

- The OpBCD instruction uses the Binary Coded Decimal format for specifying 8 bit value to use for turning the outputs on or off. Since Binary Coded Decimal format precludes the values of &HA, &HB, &HC, &HD, &HE or &HF from being used, all combinations for setting the 8 output group cannot be satisfied.
- The Out instruction works very similarly to the OpBCD instruction except that Out allows the range for the 8 bit value to use for turning outputs on or off to be between 0 and 255 (0 to 99 for OpBCD). This allows all possible combinations for the 8 bit output groups to be initiated according to the users specifications.

**Forced Flag**

This flag is used to turn On the I/O output at Emergency Stop and Safety Door Open from NoPause task, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), or background tasks.

Be sure that the I/O outputs change by Emergency Stop and Safety Door Open when designing the system.

**See Also**

In, InBCD, MemOff, MemOn, MemOut, MemSw, Off, On, Oport, Sw, Wait

**Out Statement Example**

The example shown below shows main task start a background task called “iotask”. The “iotask” is a simple task to flip flop between turning output bits 0 to 3 On and then Off. The Out instruction makes this possible using only 1 command rather than turning each output On and Off individually.

```
Function main
    Xqt iotask
    Do
        Go P1
        Go P2
    Loop
    Fend

Function iotask
    Do
        Out 0, &H0F
        Out 0, &H00
        Wait 10
    Loop
    Fend
```

496  EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
Other simple examples from the command window are as follows:

> **Out** 1, 6       'Turns on Outputs 9 & 10
> **Out** 2, 1       'Turns on Output 8
> **Out** 3, 91      'Turns on Outputs 24, 25, 27, 28, and 30
Out Function

Returns the status of one byte of outputs.

Syntax

Out(portNumber)

Parameters

portNumber Integer expression representing I/O output bytes. Where the portNumber selection corresponds to the following outputs:

<table>
<thead>
<tr>
<th>Portnum</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0-7</td>
</tr>
<tr>
<td>1</td>
<td>8-15</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Return Values

The output status 8 bit value for the specified port.

See Also

Out Statement

Out Function Example

Print Out(0)
The output data of real value is the floating-point data (IEEE754 compliant) of 32 bits. Set the status of output port 2 word (32 bits).

Syntax

```
OutReal WordPortNumber, OutputData [, Forced]
```

Parameters

- **WordPortNumber**: Integer expression representing I/O output words.
- **OutputData**: Specifies the integer expression representing the output data (Real type value).
- **Forced**: Optional. Normally omitted.

Description

Outputs the specified IEEE754 Real value to the output word port specified by word port number and the following output word port. Output word label can be used for the word port number parameter.

Note

**Forced Flag**

This flag is used to turn On the I/O output at Emergency Stop and Safety Door Open from NoPause task or NoEmgAbort task (special task initiated by specifying NoPause or NoEmgAbort at Xqt).

Carefully design the system because the I/O output changes by Emergency Stop and Safety Door Open.

See Also

- In, InW, InBCD, InReal, Out, OutW, OpBCD, OutReal Function

OutReal Statement Example

```
OutReal 32, 2.543
```
OutReal Function

Retrieve the output port status as the 32 bits floating-point data (IEEE754 compliant).

Syntax

OutReal (WordPortNumber)

Parameters

WordPortNumber  Integer expression representing I/O output words.

Return Values

Returns the specified output port status in 32 bits floating-point data (IEEE754 compliant).

See Also

In, InW, InBCD, InReal, Out, OutW, OpBCD, OutReal

OutReal Function Example

Real rdata01

rdata01 = OutReal(0)
OutW Statement

Simultaneously sets 16 output bits.

Syntax

\[
\text{OutW wordPortNum, outputData [, Forced]}
\]

Parameters

- **wordPortNum**: Integer expression representing I/O output words.
- **outputData**: Specifies output data (integers from 0 to 65535) using an expression or numeric value.
- **Forced**: Optional. Usually omitted.

Description

Changes the current status of user I/O output port group specified by the word port number to the specified output data.

Note

**Forced Flag**

This flag is used to turn On the I/O output at Emergency Stop and Safety Door Open from NoPause task, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), or background tasks. Be sure that the I/O outputs change by Emergency Stop and Safety Door Open when designing the system.

See Also

In, InW, Out

OutW Statement Example

\[
\text{OutW 0, 25}
\]
OutW Function

Returns the status of one word (2 bytes) of outputs.

Syntax

\[
\text{OutW}(\text{wordPortNum})
\]

Parameters

- \text{wordPortNum} \quad \text{Integer expression representing I/O output words.}

Return Values

The output status 16 bit value for the specified port.

See Also

- OutW Statement

OutW Function Example

\[
\text{OutW} \quad 0, \ &H1010
\]
P# (1. Point Definition)

Defines a robot point by assigning it to a point expression.

**Syntax**

\[ \text{point} = \text{pointExpr} \]
\[ \text{pointLabel} = \text{pointExpr} \]

**Parameters**

- **point**: Expression including numeric number or ( ) (parenthesis)
  - \( P\text{number} \)
  - \( P(\text{expr}) \)
- **pointLabel**: Point label
- **pointExpr**: One of the following point data
  - \( P \) point number, Point label, Here, Pallet, Point data function
  - (Here function, XY function, JA function, Pulse function, etc.)
  - For details of \( \text{pointExpr} \), refer to P# (2. Point Expression)

**Description**

Define a robot point by setting it equal to another point or point expression.

**See Also**

Local, Pallet, PDef, PDel, Plist

**Point Definition Example**

The following examples are done from the command window:

Assign coordinates to \( P1 \):

\[ > \quad P1 = 300,200,-50,100 \]

Specify left arm posture:

\[ > \quad P2 = -400,200,-80,100/L \]

Add 20 to X coordinate of \( P2 \) and define resulting point as \( P3 \):

\[ > \quad P3 = P2 +X(20) \]
\[ > \quad \text{plist 3} \]
\[ \quad P3 = -380,200,-80,100/L \]

Add 90 to U coordinate of Pallet(3, 5), and define resulting point as \( P6 \):

\[ > \quad P5 = \text{Here} \]
\[ > \quad P6 = \text{pallet}(3,5) +U(90) \]
## P# (2. Point Expression)

Specifies a robot point for assignment and motion commands.

### Syntax

```
point [ { + | - } point ] [local] [hand] [elbow] [wrist] [j4flag] [j6flag] [j1flag] [j2flag] [relativeOffsets] [absoluteCoords]
```

### Parameters

- **point**
  - The base point specification. This can be one of the following:
    - `P<number>`
    - `P(expr)`
    - `pointLabel`
    - `Pallet(palletNumber, palletIndex)`
    - `Here XY(X, Y, Z, U, [V], [W])`
    - `JA(J1, J2, J3, J4, [J5], [J6])`
    - `Pulse(J1, J2, J3, J4, [J5], [J6])`
- **local**
  - Optional. Local number from 1 to 15 preceded by a forward slash (/0 to /15) or at sign (@0 to @15). The forward slash means that the coordinates will be in the local. The at sign means that the coordinates will be translated into local coordinates.
- **hand**
  - Optional for SCARA robot (including RS series) and 6-axis robots (including N series). Specify /L or /R for lefty or righty hand orientation.
- **elbow**
  - Optional for 6-axis robots (including N series). Specify /A or /B for above or below orientation.
- **wrist**
  - Optional for 6-axis robots (including N series). Specify /F or /NF for flip or no flip orientation.
- **j4flag**
  - Optional for 6-axis robots (including N series). Specify /J4F0 or /J4F1.
- **j6flag**
  - Optional for 6-axis robots (including N series). Specify /J6F0 to /J6F127.
- **j1flag**
  - Optional for RS series and 6-axis robots (excluding N series). Specify /J1F0 or /J1F1.
- **j2flag**
- **j1angle**
  - Optional for RS and N series. Specify /J1A (real value).
- **j4angle**
  - Optional for N series. Specify /J4A (real value).
- **relativeOffsets**
  - Optional. One or more relative coordinate adjustments.
    - `{+ | - } {X | Y | Z | U | V | W | RZ | RY | RX | R | S | T | ST } (expr)`
    - The TL offsets are relative offsets in the current tool coordinate system.
- **absoluteCoords**
  - Optional. One or more absolute coordinates.
    - `{X | Y | Z | U | V | W | R | S | T | ST } (expr)`

### Description

Point expressions are used in point assignment statements and motion commands.

- `Go P1 + P2`
- `P1 = P2 + XY(100, 100, 0, 0)`
Using relative offsets
You can offset one or more coordinates relative to the base point. For example, the following statement moves the robot 20 mm in the positive X axis from the current position:

\[ \text{Go Here } +X(20) \]

If you execute the same statement again, the robot will move an additional 20 mm along the X axis, because this is a relative move.

To make a relative rotation around the coordinate axis of the 6-axis robots (including N series), execute the statement as follows. The following statement rotates the tool 20 degrees in the X-axis positive direction based on the current tool orientation.

\[ \text{Go Here } +RX(20) \]

You can also use relative tool offsets:

\[ \text{Go Here } +TLX(20) -TLY(5.5) \]

When the 6-axis robot (including N series) moves to a point calculated by such as pallet or relative offsets, the wrist part may rotate to an unintended direction. The point calculation above does not depend on robot models and results in motion without converting the required point flag.

LJM function prevents the unintended wrist rotation.

\[ \text{Go LJM(Here } +X(20)) \]

Using absolute coordinates
You can change one or more coordinates of the base point by using absolute coordinates. For example, the following statement moves the robot to the 20 mm position on the X axis:

\[ \text{Go Here } :X(20) \]

If you execute the same statement again, the robot will not move because it is already in the absolute position for X from the previous move.

Relative offsets and absolute coordinates make is easy to temporarily modify a point. For example, this code moves quickly above the pick point by 10 mm using a relative offset for Z or 10 mm, then moves slowly to the pick point.

\[
\begin{align*}
\text{Speed fast} \\
\text{Jump pick } +Z(10) \\
\text{Speed slow} \\
\text{Go pick}
\end{align*}
\]

This code moves straight up from the current position by specifying an absolute value of 0 for the Z joint:

\[
\begin{align*}
\text{LimZ 0} \\
\text{Jump Here } :Z(0)
\end{align*}
\]

Using Locals
You can specify a local number using a forward slash or at sign. Each has a separate function.

Use the forward slash to mark the coordinates in a local. For example, adding a /1 in the following statement says that P1 will be at location 0,0,0,0 in local 1.

\[ P1 = XY(0, 0, 0, 0) /1 \]

Use the at sign to translate the coordinates into local coordinates.
For example, here is how to set the current position to P1:

\[ P1 = \text{Here @1} \]
P# (2. Point Expression)

See Also
Go, LJ M, Local, Pallet, Pdel, Plist, Hand, Elbow, Wrist, J4Flag, J6Flag, J1Flag, J2Flag

Point Expression Example
Here are some examples of using point expressions in assignments statements and motion commands:

\[
P1 = XY(300, 200, -50, 100) \\
P2 = P1 /R \\
P3 = pick /1 \\
P4 = P5 + P6 \\
P(i) = XY(100, 200, CZ(P100), 0) \\
Go P1 -X(20) :Z(-20) /R \\
Go Pallet(1, 1) -Y(25.5) \\
Move pick /R \\
Jump Here :Z(0) \\
Go Here :Z(-25.5) \\
Go JA(25, 0, -20, 180) \\
pick = XY(100, 100, -50, 0) \\
P1 = XY(300, 200, -50, 100, -90, 0) \\
P2 = P1 /F /B \\
P2 = P1 +TLV(25)
\]
PAgl Function

Returns a joint value from a specified point.

**Syntax**

```
PAgl (point, jointNumber)
```

**Parameters**

- `point`: Point expression.
- `jointNumber`: Specifies the joint number (integer from 1 to 9) using an expression or numeric value. The additional S axis is 8 and T axis is 9.

**Return Values**

Returns the calculated joint position (real value, deg for rotary joint, mm for prismatic joint).

**See Also**

Agl, CX, CY, CZ, CU, CV, CW, CR, CS, CT, PPls

**PAgl Function Example**

```
Real joint1

joint1 = PAgl(P10, 1)
```
Pallet Statement

Defines and displays pallets.

Syntax

(1) Pallet [Outside.] [ palletNumber, P1, P2, P3 [, P4 ], columns, rows ]
(2) Pallet [Outside.] palletNumber, coordinateData 2, coordinateData 3
               [, coordinateData 4 ], columns1, rows2
(3) Pallet

Parameters

Outside Optional. Allow row and column indexes outside of the range of the specified rows and columns.

palletNumber Pallet number represented by an integer number from 0 to 15.

P1, P2, P3 Point variables which define standard 3 point pallet position.

P4 Optional. Point variable which is used with P1, P2 and P3 to define 4 point pallet.

columns Integer expression representing the number of points on the P1(coordinateData 1)-to-P2(coordinateData 2) side of the pallet. Range is from 1 to 32767.

rows Integer expression representing the number of points on the P1(coordinateData 1)-to-P3(coordinateData 3) side of the pallet. Range is from 1 to 32767.

coordinateData 1, 2, 3 Point data which is used for pallet definition (standard 3-point definition)

coordinateData 4 Optional. Point data which is used with coordinateData 1 to 3 for 4-point pallet definition.

Return Values

(3) Displays all defined pallets when parameters are omitted.

Description

Defines a pallet by teaching the robot, as a minimum, points P1 (coordinateData 1), P2 (coordinateData 2) and P3 (coordinateData 3) and by specifying the number of points from P1 (coordinateData 1) to P2 (coordinateData 2) and from P1 (coordinateData 1) to P3 (coordinateData 3).

If the pallet is a well ordered rectangular shape, only 3 of the 4 corner points need to be specified. However, in most situations it is better to use 4 corner points for defining a pallet.

To define a pallet, first teach the robot either 3 or 4 corner points, then define the pallet as follows:

A pallet defined with 4 points: P1, P2, P3 and P4 is shown below. There are 3 positions from P1-P2 and 4 positions from P1-P3. This makes a pallet which has 12 positions total. To define this pallet the syntax is as follows:

Pallet 1, P1, P2, P3, P4, 3, 4

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

P1   P2

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

P3   P4

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

1   2   3

7   8   9

4   5   6

10  11  12
Points that represent divisions of a pallet are automatically assigned division numbers, which, in this example, begin at P1. These division numbers are also required by the Pallet Function.

When Outside is specified, row and column indexes outside of the range of rows and columns can be specified.

For example:

```
Pallet Outside 1, P1, P2, P3, 4, 5
Jump Pallet(1, -2, 10)
```

### Notes

**The Maximum Pallet Size**

The total number of points defined by a specific pallet must be less than 32,767.

**Incorrect Pallet Shape Definitions**

Be aware that incorrect order of points or incorrect number of divisions between points will result in an incorrect pallet shape definition.

**Pallet Plane Definition**

The pallet plane is defined by the Z axis coordinate values of the 3 corner points of the pallet. Therefore, a vertical pallet could also be defined.

**Pallet Definition for a Single Row Pallet**

A single row pallet can be defined with a 3 point Pallet statement or command. Simply teach a point at each end and define as follows: Specify 1 as the number of divisions between the same point.

```
> Pallet 2, P20, P21, P20, 5, 1
```

Defines a 5×1 pallet

**UVW Coordinate Values**

When the UVW coordinate values of the 3 (or 4) points specified with the Pallet statement vary, the UVW coordinate values of the point 1 and the coordinate system data 1 are used. The UVW coordinate values of the point numbers from 2 to 4 and the coordinate system numbers from 2 to 4 are ignored.

**Additional Axes Coordinate Values**

When the coordinate values of the 3 (or 4) points specified with the Pallet statement include the additional ST axis coordinate values, Pallet includes these additional coordinates in the position calculations. In the case where the additional axis is used as the running axis, the motion of the running axis is considered and calculated with the Pallet definition. You need to define a pallet larger than the robot motion range considering the position of the running axis. Even if you define additional axes that are not affected by the pallet definition, be careful of the positions of additional axes when defining the pallet.
Pallet Statement

See Also
Pallet Function

Pallet Statement Example
The following instruction from the command window sets the pallet defined by P1, P2 and P3 points, and divides the pallet plane into 15 equally distributed pallet point positions, with the pallet point number 1, the pallet point number 2 and the pallet point number 3 sitting along the P1-to-P2 side.

```plaintext
> pallet 1, P1, P2, P3, 3, 5
> jump pallet(1, 2)
'Jump to position on pallet
```

The resulting Pallet is shown below:

```
P3
  13 14 15
  10 11 12
   7  8  9
   4  5  6
   1  2  3
P1   P2
```
Pallet Function

Specifications a position in a previously defined pallet.

Syntax

(1) Pallet ( palletNumber, palletPosition )
(2) Pallet ( palletNumber, column, row )

Parameters

palletNumber      Pallet number represented by integer expression from 0 to 15.

PalletPosition    The pallet position represented by an integer from 1 to 32767.

column            The pallet column represented by an integer expression from −32768 to 32767.

row               The pallet row represented by an integer expression from −32768 to 32767.

Description

Pallet returns a position in a pallet which was previously defined by the Pallet statement. Use this function with motion commands such as Go and Jump to cause the arm to move to the specified pallet position.

The pallet position number can be defined arithmetically or simply by using an integer.

Notes

Pallet Motion of 6-axis Robot (including N series)

When the 6-axis robot (including N series) moves to a point calculated by such as pallet or relative offsets, the wrist part may rotate to an unintended direction. The point calculation above does not depend on robot models and results in motion without converting the required point flag.

LJM function prevents the unintended wrist rotation.

Pallet Motion of RS series

In the same way as the 6-axis, when the RS series robot moves to a point calculated by such as pallet or relative offsets, Arm #1 may rotate to an unintended direction. LJM function can be used to convert the point flag to prevent the unintended rotation of Arm #1.

In addition, the U axis of RS series may go out of the motion range when the orientation flag is converted, and it causes an error.

To prevent this error, LJM function adjusts the U axis target angle to inside the motion range. It is available when the orientation flag “2” is selected.

UVW Coordinate Values

When the UVW coordinate values of the 3 (or 4) points specified with the Pallet statement vary, the UVW coordinate values of the point 1 and the coordinate system data 1 are used.

The UVW coordinate values of the point numbers from 2 to 4 and the coordinate system numbers from 2 to 4 are ignored.

Additional Axes Coordinate Values

When the coordinate values of the 3 (or 4) points specified with the Pallet statement include the additional ST axis coordinate values, Pallet includes these additional coordinates in the position calculations. In the case where the additional axis is used as the running axis, the motion of the running axis is considered and calculated with the Pallet definition. You need to define a pallet larger than the robot motion range considering the position of the running axis. Even if you define additional axes that are not affected by the pallet definition, be careful of the positions of additional axes when defining the pallet.

See Also

LJM, Pallet
Pallet Function Example

The following program transfers parts from pallet 1 to pallet 2.

Function main
  Integer index
  Pallet 1, P1, P2, P3, 3, 5  'Define pallet 1
  Pallet 2, P12, P13, P11, 5, 3  'Define pallet 2
  For index = 1 To 15
    Jump Pallet(1, index)  'Move to point index on pallet 1
    On 1  'Hold the work piece
    Wait 0.5
    Jump Pallet(2, index)  'Move to point index on pallet 2
    Off 1  'Release the work piece
    Wait 0.5
  Next I
Fend

Function main
  Integer i, j
  P0 = XY(300, 300, 300, 90, 0, 180)
  P1 = XY(200, 280, 150, 90, 0, 180)
  P2 = XY(200, 330, 150, 90, 0, 180)
  P3 = XY(-200, 280, 150, 90, 0, 180)
  Pallet 1, P1, P2, P3, 10, 10
  Motor On
  Power High
  Speed 50; Accel 50, 50
  SpeedS 1000; AccelS 5000
  Go P0
  P11 = P0 - TLZ(50)
  For i = 1 To 10
    For j = 1 To 10
      'Specify points
      P10 = P11
      P12 = Pallet(1, i, j)  'Depart point
      P11 = P12 - TLZ(50)  'Start approach point
      'Converting each point to LJM
      P10 = LJM(P10)
      P11 = LJM(P11, P10)
      P12 = LJM(P12, P11)
      'Execute motion
      Jump3 P10, P11, P12 C0
    Next j
  Next i
Fend
Function main2

P0 = XY(300, 300, 300, 90, 0, 180)
P1 = XY(400, 0, 150, 90, 0, 180)
P2 = XY(400, 500, 150, 90, 0, 180)
P3 = XY(-400, 0, 150, 90, 0, 180)
Pallet 1, P1, P2, P3, 10, 10

Motor On
Power High
Speed 50; Accel 50, 50
SpeedS 1000; AccelS 5000

Go P0

Do

' Specify points
P10 = Here - TLZ(50)
P12 = Pallet(1, Int(Rnd(9)) + 1, Int(Rnd(9)) + 1) ' Target point
P11 = P12 - TLZ(50) ' Start approach point

If TargetOK(P11) And TargetOK(P12) Then ' Point check
    ' Converting each point to LJM
    P10 = LJM(P10)
P11 = LJM(P11, P10)
P12 = LJM(P12, P11)

    ' Execute motion
    Jump3 P10, P11, P12 C0

EndIf

Loop

Fend
PalletClr Statement

Clears a defined pallet.

Syntax

PalletClr  palletNumber

Parameters

palletNumber  Pallet number represented by integer expression from 0 to 15.

See Also

Pallet

PalletClr Example

PalletClr  1
ParseStr Statement / Function

Parses a string and return array of tokens.

Syntax

```
ParseStr inputString$, tokens$(), delimiters$
numTokens = ParseStr(inputString$, tokens$(), delimiters$)
```

Parameters

- `inputString$` String expression to be parsed.
- `tokens$()` Output array of strings containing the tokens.
  The array declared by ByRef cannot be specified.
- `delimiters$` String expression containing one or more token delimiters.

Return Values

When used as a function, the number of tokens parsed is returned.

See Also

Redim, String

ParseStr Statement Example

```
String toks$(0)
Integer i

ParseStr "1 2 3 4", toks$(), " ", " 

For i = 0 To UBound(toks)
    Print "token ", i, " = ", toks$(i)
Next i
```
### Pass Statement

Executes simultaneous four joint Point to Point motion, passing near but not through the specified points.

#### Syntax

```
Pass point [, {On | Off | MemOn | MemOff} bitNumber [, point ... ]] [LJM [orientationFlag]]
```

#### Parameters

- **point**
  - `Pnumber` or `P(expr)` or point label.
  - When the point data is continued and in the ascending order or the descending order, specify two point numbers binding with colon as `P(1:5)`.

- **bitNumber**
  - The I/O output bit or memory I/O bit to turn on or off. Integer number between 0 - 511 or output label.

- **LJM**
  - Optional. Convert the depart point, approach point, and target destination using LJM function.

- **orientationFlag**
  - Optional. Specifies a parameter that selects an orientation flag for LJM function.

#### Description

Pass moves the robot arm near but not through the specified point series.

To specify a point series, use points (P0,P1, ...) with commas between points.

To turn output bits on or off while executing motion, insert an On or Off command delimited with commas between points. The On or Off is executed before the robot reaches the point immediately preceding the On or Off.

If Pass is immediately followed by another Pass, control passes to the following Pass without the robot stopping at the preceding Pass final specified point.

If Pass is immediately followed by a motion command other than another Pass, the robot stops at the preceding Pass final specified point, but Fine positioning will not be executed.

If Pass is immediately followed by a command, statement, or function other than a motion command, the immediately following command, statement or function will be executed prior to the robot reaching the final point of the preceding Pass.

If Fine positioning at the target position is desired, follow the Pass with a Go, specifying the target position as shown in the following example:

```
Pass P5; Go P5; On 1; Move P10
```

The larger the acceleration / deceleration values, the nearer the arm moves toward the specified point. The Pass instruction can be used such that the robot arm avoids obstacles.
With LJM parameter, the program using LJM function can be more simple. For example, the following four-line program

```plaintext
P11 = LJM(P1, Here, 1)
P12 = LJM(P2, P11, 1)
P13 = LJM(P3, P12, 1)
Pass P11, P12, P13
```

can be… one-line program.

```plaintext
Pass P1, P2, P3 LJM 1
```

LJM parameter is available for 6-axis (including N series) and RS series robots. When using `orientationFlag` with the default value, it can be omitted.

```plaintext
Pass P1, P2, P3 LJM
```

**See Also**

Accel, Go, Jump, Speed

**Pass Statement Example**

The example shows the robot arm manipulation by Pass instruction:

```plaintext
Function main
  Jump P1
  Pass P2  ' Move the arm toward P2, and perform the next instruction before reaching P2.
  On 2
  Pass P3
  Pass P4
  Off 0
  Pass P5
Fend
```
Pause Statement

Temporarily stops program execution all tasks for which pause is enabled.

Syntax
Pause

Description
When the Pause is executed, program execution for all tasks with pause enabled (tasks that do not use NoPause or NoEmgAbort in Xqt command) is suspended. Also, if any task is executing a motion statement, it will be paused even if pause is not enabled for that task. However, Pause cannot stop the background tasks.

Note
QP and its Effect on Pause
The QP instruction is used to cause the arm to stop immediately upon Pause or to complete the current move and then Pause the program. See the QP instruction help for more information.

Pause Statement Example
The example below shows the use of thePause instruction to temporarily stop execution. The task executes program statements until the line containing the Pause command. At that point the task is paused. The user can then click the Run Window Continue Button to resume execution.

Function main
  Xqt monitor
  Go P1
  On 1
  Jump P2
  Off 1
  Pause  ' Suspend program execution
  Go P40
  Jump P50
Fend
PauseOn Function

Returns the pause status.

**Syntax**

```
PauseOn
```

**Return Values**

True if the status is pause, otherwise False.

**Description**

PauseOn function is used only for NoPause, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), and background tasks.

**See Also**

ErrorOn, EstopOn, SafetyOn, Wait, Xqt

**PauseOn Function Example**

The following example shows a program that monitors the controller pause and switches the I/O On/Off when pause occurs. However, when the status changes to pause by Safety Door open, the I/O does not turn On/Off.

```pascal
Function main
  Xqt PauseMonitor, NoPause
  :
  Fend

Function PauseMonitor
  Boolean IsPause
  IsPause = False
  Do
    Wait 0.1
    If SafetyOn = On Then
      If IsPause = False Then
        Print "Safety On"
        IsPause = True
      EndIf
    ElseIf PauseOn = On Then
      If IsPause = False Then
        Print "InPause"
        If SafetyOn = Off Then
          Off 10
        On 12
        EndIf
      EndIf
      IsPause = True
    EndIf
    Else
      If IsPause = True Then
        Print "OutPause"
        On 10
        Off 12
      EndIf
    EndIf
  Loop

Fend
```
PDef Function

Returns the definition status of a specified point.

Syntax

PDef (point)

Parameters

point  An integer value or Pnumber or P(expr) or point label.

Cautions for compatibility

No variables can be specified for point parameter
To use variables, write PDef(P(varName)).

Return Values

True if the point is defined, otherwise False.

See Also

Here Statement, Pdel

PDef Function Example

If Not PDef(1) Then
   Here P1
EndIf
Integer i
For i = 0 to 10
   If PDef(P(i)) Then
      Print “P(;i;) is defined”
   EndIf
Next
PDel Statement

Deletes specified position data.

Syntax

\[ \text{PDel } \text{firstPointNum} [, \text{lastPointNum} ] \]

Parameters

- \text{firstPointNum} \quad \text{The first point number in a sequence of points to delete. } \text{firstPointNum} \text{ must be an integer.}
- \text{lastPointNum} \quad \text{The last point number in a sequence of points to delete. } \text{lastPointNum} \text{ must be an integer.}

Description

Deletes specified position data from the controller's point memory for the current robot. Deletes all position data from \text{firstPointNum} up to and including \text{lastPointNum}. To prevent Error 2 from occurring, \text{firstPointNum} must be less than \text{lastPointNum}.

PDel Statement Example

\begin{verbatim}
> p1=10,300,-10,0/L
> p2=0,300,-40,0
> p10=-50,350,0,0
> pdel 1,2           'Delete points 1 and 2
> plist
P10  =   -50.000,  350.000,    0.000,    0.000 /R /0
> pdel 50           'Delete point 50
> pdel 100,200      'Delete from point 100 to point 200
> 
\end{verbatim}
PDescription Statement

Define a comment of specified point data.

Syntax

PDescription point data, Newcomment

Parameters

Point data An integer value or Pnumber or P(expr) or point label.

No variables can be specified for point data parameter.
To use variables, write PDescription Statement (P(varName)), “new comment”.

New comment String expression representing comment of specified point.

Description

PDescription save a description in specified point data of controller memory.
Description saved in memory of the controller is deleted from memory when creating or executing a program. Execute the “SavePoints” to save in point file if necessary.

See Also

PDef Function, PDescription$ Function, PLabel, PLabel$ Function

PDescription Statement Example

PDescription 1, "Comment"
PDescription$ Function

Returns description of point that defined to the specified point number.

Syntax

PDescription$( pointData)

Parameters

pointData  An integer value or Pnumber or P(expr) or point label.

No variables can be specified for point data parameter.
To use variables, write PDescription$(P(varName)).

Return Values

Returns descriptions of specified number as a string.

See Also

PDef Function, PDescription, PLabel, PLabel$ Function

PDescription$ Function Example

Print PDescription$(1)
Print PDescription$(P(i))
PeakSpeedClear Statement

Clears and initializes the peak speed for one or more joints.

Syntax
PeakSpeedClear [j1 [, j2 [, j3 [, j4 [, j5 [, j6 [, j7 [, j8 [, j9]]]]]]]]

Parameters
j1 – j9  Integer expression representing the joint number. If no parameters are supplied, then the peak speed values are cleared for all joints. The additional S axis is 8 and T axis is 9. If non-existent joint number is supplied, an error occurs.

Description
PeakSpeedClear clears the peak speed values for the specified joints. You must execute PeakSpeedClear before executing PeakSpeed. This command does not support the PG additional axes.

See Also
AvgSpeed, PeakSpeed

PeakSpeedClear Statement Example
<Example 1>
The following is the example to display the speed values of specified joints after clearing the peak speed values of all joints.

> PeakSpeedClear
> Go P1
> PeakSpeed 1  
  -0.273
> PeakSpeed  
  -0.273  -0.164 
  -0.080  0.258 
  -0.005  0.401 
  0.000  0.000 
  0.000
>

<Example 2>
The following is the example to display the peak speed values of specified joints after clearing the peak speed values of J1, J4, and J5 for the vertical multi-axis robots.

> PeakSpeedClear 4, 1, 5
> Go P1
> PeakSpeed 1  
  -0.273
> PeakSpeed 4  
  0.258
## PeakSpeed Statement

Displays the peak speed values for the specified joint.

### Syntax

```
PeakSpeed [jointNumber]
```

### Parameters

- **jointNumber**: Optional. Integer expression representing the joint number.
  The additional S axis is 8 and T axis is 9.

### Return Values

Displays current peak speed values for all joints.

### Description

PeakSpeed statement displays the value of the maximum absolute speed for the joint with a sign. The peak speed is a real number from -1 to 1 with 1 being the maximum speed.

Execute PeakSpeedClear first, and then execute PeakSpeed to display the peak speed value for the joint.

When using the virtual controller or conducting dry-run, the average of the absolute speed values is calculated from the commanded speed instead of the actual speed.

This command does not support the PG additional axes.

### See Also

AvgSpeed, PeakSpeedClear, PeakSpeed Function

### PeakSpeed Statement Example

```
> PeakSpeedClear
> Go P1
> PeakSpeed 1
>     -0.273
> PeakSpeed
>     -0.273  0.163
>     -0.080  0.258
>     -0.005 -0.401
>     0.000  0.000
>     0.000
>     
```
PeakSpeed Function

Returns the peak speed for the specified joint.

Syntax
PeakSpeed (jointNumber)

Parameters
jointNumber  Integer expression representing the joint number.
The additional S axis is 8 and T axis is 9.

Return Values
Real value from -1 to 1.

Description
PeakSpeed function returns the value of the maximum absolute speed for the joint with a sign. The peak speed is a real number from -1 to 1 with 1 being the maximum speed.

Execute PeakSpeedClear statement first, and then execute PeakSpeed statement to display the peak speed value for the joint.

When using the virtual controller or conducting dry-run, the average of the absolute speed values is calculated from the commanded speed instead of the actual speed.
This command does not support the PG additional axes.

See Also
AvgSpeed, PeakSpeedClear, PeakSpeed

PeakSpeed Function Example
This example uses the PeakSpeed function in a program:

Function DisplayPeakSpeed
  Integer i
  PeakSpeedClear
  Go P1
  Print "Peak Speeds:"
  For i = 1 To 6
    Print "Joint ", i, " = ", PeakSpeed (i)
  Next i
Fend

526  EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
PerformMode Statement

Sets the mode of the robot.

Syntax

(1) PerformMode [modeNumber] [, robotNumber]
(2) PerformMode

Parameters

modeNumber Specify the operation mode with an integer value (1 to 3) or with the following constant. This parameter is optional only when the statement is executed from the command window.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODE_STANDARD</td>
<td>1</td>
<td>Sets the Standard mode</td>
</tr>
<tr>
<td>MODE_HIGH_SPEED</td>
<td>2</td>
<td>Sets the High-speed mode</td>
</tr>
<tr>
<td>MODE_LOW_OSCILLATION</td>
<td>3</td>
<td>Sets the Low-oscillation mode</td>
</tr>
</tbody>
</table>

robotNumber Specify the robot number by an integer value.

If omitted, currently selected robot will be used.

Result

When specified by the syntax (1), the mode will be set by the mode number that is specified.
When specified by the syntax (2), the mode number of the currently selected robot will be displayed.

Description

PerformMode is a function to change the preference of manipulator performance (mode) according to the intended use. This function supports following three modes.

Standard

The cycle time, the duty, and the oscillation at the motion stop are balanced.
This mode is available for any kind of application.

High-speed

This mode is specialized to reduce the operating time of a task.
Although this mode aggravates the duty and oscillation at the motion stop compared to the standard mode, it can reduce operation time.
Recommended application: Transportation

Low-oscillation

This mode is specialized to reduce the oscillation at the motion stop.
Although this mode increases the operating time compared to the standard mode, it can reduce the oscillation at the motion stop.
Recommended application: Transportation and assembly of precision components
Performance comparison

<table>
<thead>
<tr>
<th>Mode</th>
<th>Operating time (*1)</th>
<th>Oscillation</th>
<th>Duty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>Normal</td>
<td>Normal</td>
<td>Normal</td>
</tr>
<tr>
<td>High-speed</td>
<td>Improved</td>
<td>Decreased</td>
<td>Decreased</td>
</tr>
<tr>
<td>Low-oscillation</td>
<td>Decreased</td>
<td>Improved</td>
<td>Improved</td>
</tr>
</tbody>
</table>

(*1) Traveling time of the manipulator moving from the current position to the target point.

Note

- Target motion commands: PTP motion commands (Go, BGo, TGo, Jump, JTran)

  * Following performance of the CP motion are not affected by Precede statement.

  Trajectory accuracy

  Upper limit values of AccelS, AccelR, SpeedS, SpeedR

  Frequency of the acceleration setting error and the speed setting error

Conditions that automatically initialize the mode (to the Standard mode)

The table below shows the conditions which automatically initializes the mode.

<table>
<thead>
<tr>
<th>Change of the Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller power ON</td>
</tr>
<tr>
<td>Controller reboot</td>
</tr>
<tr>
<td>Motor ON</td>
</tr>
<tr>
<td>Switched the Power (Low ↔ High)</td>
</tr>
<tr>
<td>Build / Rebuild</td>
</tr>
<tr>
<td>Reset</td>
</tr>
</tbody>
</table>

See Also

Bo, Go, Jump, JTran, PerformMode Function, TGo

PerformMode Statement Example

```epson
PerformMode MODE_STANDARD
Go P1
PerformMode 2
Go P2
```
PerformMode Function

Returns the status of the robot operation mode.

Syntax

```
PerformMode ([robotNumber])
```

Parameters

robotNumber Specify the robot number to check the status by an integer value.
If omitted, currently selected robot will be used.

Return Values

Returns the integer value representing the currently set operation mode.
1 = Standard mode
2 = High-speed mode
3 = Low-oscillation mode

See Also

PerformMode

PerformMode Function Example

```
Print PerformMode(1)
```
**Stop the PG axes immediately.**

**Syntax**

```
PG_FastStop
```

**Description**

The PG_FastStop stops the current PG robot immediately with no deceleration. To stop normally, use the PG_SlowStop statement.

**See Also**

PG_Scan, PG_SlowStop

**Example**

The following program moves the PG axis for 10 seconds and stops it.

```plaintext
Function main
    Motor On
    PG_Scan 0
    Wait 10
    PG_FastStop       ' Immediately stops the continuous motion
    Fend
```
PG_LSpeed Statement

Sets the pulse speed of the time when the PG axis starts accelerating and finishes decelerating.

Syntax

PG_LSpeed accelSpeed As Integer [, decelSpeed As Integer ],

Parameters

speed Integer expression that contains the pulse speed (1 to 32767 pulse/second)
decalSpeed Integer expression that contains the pulse speed (1 to 32767 pulse/second)

Description

PG_LSpeed specifies the pulse speed when the PG axis starts accelerating and finishes decelerating. It is useful when setting the initial/ending speed of a stepping motor to higher within the range of max starting frequency to offer the best performance of motor, or setting the speed to lower to prevent the stepping motor from stepping out. The default is 300 pulse/second and do not change to use.

If omitted the finishing speed of deceleration, the speed set value is used.

The PG_LSpeed value initializes to the default values when any one of the following conditions occurs:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

See Also

PG_LSpeed function

PG_LSpeed Statement Example

You can use the PG_LSpeed in the command window or in the program. The following examples show the both cases.

Function pglspeedtst
  Motor On
  Power High
  Speed 30;Accel 30,30
  PG_LSpeed 1000
  Go P0
Fend

To set the PG_LSpeed value from the command window.

  > PG_LSpeed 1000,1100
  >
PG_LSpeed Function

Returns the pulse speed at the time when the current PG axis starts accelerating and finishes decelerating.

Syntax

\[ \text{PG\_LSpeed} \left( \text{paramNumber} \right) \]

Parameters

\begin{itemize}
\item \textit{paramNumber} One of the numbers below that specifies the number of set value.
\item If omitted, 1 is used.
\item 1: Pulse speed at acceleration starts
\item 2: Pulse speed at deceleration finishes
\end{itemize}

Return Values

Integer value from 1 to 32767 in units of pulse/second.

See Also

PG_LSpeed

PG_LSpeed Function Example

```
Integer savPGLSpeed

savPGLSpeed = \text{PG\_LSpeed}(1)
```

PG_Scan Statement

Starts the continuous spinning motion of the PG robot axes.

Syntax

\[ \text{PG}\_\text{Scan} \ \text{direction} \ \text{As Integer} \]

Parameters

- **direction**: Spinning direction
  - 0: \(+\) (CW) direction
  - 1: \(-\) (CCW) direction

Description

The PG_Scan starts the continuous spinning motion of the current PG robot. To execute the continuous spinning motion, you need to enable the PG parameter continuous spinning by the robot configuration. When the program execution task is completed, the continuous spinning stops.

See Also

- PG_FastStop

PG_Scan Statement Example

The following example spins the PG axis for 10 seconds and stops it suddenly.

```
Function main
  Motor On
  Power High
  Speed 10; Accel 10,10
  \textbf{PG}\_\textbf{Scan} 0
  Wait 10
  PG\_SlowStop
Fend
```
PG_SlowStop Statement

Stops slowly the PG axis spinning continuously.

Syntax

    PG_SlowStop

Description

    PG_SlowStop decelerates the continuous spinning motion of the current PG robot and bring it to a stop.

See Also

    PG_Scan, PG_FastStop

PG_SlowStop Statement Example

The following example spins the PG axis for 10 seconds and stop it suddenly.

    Function main
        Motor On
        PG_Scan 0
        Wait 10
            PG_SlowStop  ' Stops suddenly the continuous spinning motion
    Fend
PLabel Statement

Defines a label for a specified point.

Syntax

PLabel pointNumber, newLabel

Parameters

pointNumber An integer expression representing a point number.
newLabel A string expression representing the label to use for the specified point.

See Also

PDef Function, PDescription, PDescription$ Function, PLabel$ Function, PNumber Function

PLabel Statement Example

PLabel 1, "pick"
PLabel$ Function

Returns the point label associated with a point number.

Syntax

\[
PLabel$(point)
\]

Parameters

point An integer value or Pnumber or P(expr) or point label.

Cautions for compatibility
No variables can be specified for point parameter
To use variables, write PLabel$(P(varName)).

See Also
PDef Function, PDescription, PDescription$ Function,PLabel, PNumber Function

PLabel$ Function Example

\[
\begin{align*}
&\text{Print } PLabel$(1) \\
&\text{Print } PLabel$(P(i))
\end{align*}
\]
Plane Statement

Specifies and displays the approach check plane.

Syntax

1. Plane PlaneNum [, robotNumber], pCoordinateData
2. Plane PlaneNum [, robotNumber], pOrigin, pXaxis, pYaxis
3. Plane PlaneNum [, robotNumber]
4. Plane

Parameters

- PlaneNum: Integer value representing the plane number from 1 to 15.
- robotNumber: Integer values representing the robot number.
- If omitted, the current robot is used.
- pCoordinateData: Point data representing the coordinate data of the approach check plane.
- pOrigin: Integer expression representing the origin point using the robot coordinate system.
- pXaxis: Integer expression representing a point along the X axis using the robot coordinate system if X alignment is specified.
- pYaxis: Integer expression representing a point along the Y axis using the robot coordinate system if Y alignment is specified.

Return Values

- When using syntax (3), the setting of the specified plane is displayed.
- When using syntax (4), the settings of all plane numbers for the current robot are displayed.

Description

Plane is used to set the approach check plane. The approach check plane is for checking whether the robot end effector is in one of the two areas divided by the specified approach check plane. The position of the end effector is calculated by the current tool. The approach check plane is set using the XY plane of the base coordinate system. The approach check plane detects the end effector when it approaches the area on the +Z side of the approach check plane.

When the approach check plane is used, the system detects approaches in any motor power status during the controller is ON.

The details of each syntax are as follows.

1. Specifies a coordinate system to create the approach check plane using the point data representing the translation and rotation based on the base coordinate system, and sets the approach check plane.

   Example:
   ```
   Plane 1, XY(x, y, z, u, v, w)
   Plane 1, P1
   ```

2. Defines the approach check plane (XP coordinate) by specifying the origin point, point along the X axis, and point along the Y axis. Uses the X, Y, Z coordinates and ignores U, V, W coordinates. Calculates the Z axis in righty and sets the approach checking direction.

   Example:
   ```
   Plane 1, P1, P2, P3
   ```

3. Displays the setting of the specified approach check plane.

4. Displays all the approach check plane.
You can use the GetRobotInsidePlane function and the InsidePlane function to get the result of the approach check plane. The GetRobotInsidePlane function can be used as the condition for a Wait command. You can provide the detection result to the I/O by setting the remote output setting.

To use one plane with more than one robot, you need to define planes from each robot coordinate system.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Notes

Tool Selection
The approach check is executed for the current tool. When you change the tool, the approach check may display the tool approach from inside to outside of the plane or the other way although the robot is not operating.

Additional axis
For the robot which has the additional ST axes (including the running axis), the approach check plane to set doesn’t depend on the position of an additional axis, but is based on the robot base coordinate system.

See Also
Box, GetRobotInsidePlane, InsidePlane, PlaneClr, PlaneDef

Tip

Set Plane statement from Robot Manager
EPSON RC+ has a point and click dialog box for defining the approach check plane. The simplest method to set the Plane values is by using the Plane page on the Robot Manager.

Plane Statement Example
These are examples to set the approach check plane using Plane statement.

Check direction is the lower side of the horizontal plane that is −20 mm in Z axis direction in the robot coordinate system:

> plane 1, xy(100, 200, -20, 90, 0, 180)

Approach check plane is the XY coordinate created by moving 50 mm in X axis and 200 mm in Y axis, rotating 45 degrees around Y axis:

> plane 2, xy(50, 200, 0, 0, 45, 0)

Set the approach check plane using the tool coordinate system of the robot. (6-axis robot)

> plane 3, here
Plane Function

Returns the specified approach check plane.

Syntax

Plane(PlaneNum [, robotNumber])

Parameters

- **PlaneNum**: Integer expression representing the plane number from 1 to 15.
- **robotNumber**: Integer values representing the robot number
  
  If omitted, the current robot is used.

Return Values

Returns coordinate data for specified approach check plane.

See Also

GetRobotInsidePlane, InsidePlane, Plane, PlaneClr, PlaneDef

Plane Function Example

P1 = Plane(1)
PlaneClr Statement

Clears (undefines) a Plane definition.

Syntax

```
PlaneClr PlaneNum [, robotNumber ]
```

Parameters

- `PlaneNum`: Integer expression representing the plane number from 1 to 15.
- `robotNumber`: Integer value representing the robot number
  If omitted, the current robot is used.

Description

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also
GetRobotInsidePlane, InsidePlane, Plane, PlaneDef

PlaneClr Statement Example

```
PlaneClr 1
```
PlaneDef Function

Returns the setting of the approach check plane.

Syntax

PlaneDef (PlaneNum [, robotNumber])

Parameters

PlaneNum  
Integer expression representing the plane number from 1 to 15.

robotNumber  
Integer value representing the robot number
If omitted, the current robot is used.

Return Values

True if approach detection plane is defined for the specified plane number, otherwise False.

See Also

GetRobotInsidePlane, Box, InsidePlane, Plane, PlaneCIR

PlaneDef Function Example

Function DisplayPlaneDef(planeNum As Integer)
    
    If PlaneDef(planeNum) = False Then
        Print "Plane ", planeNum, "is not defined"
    Else
        Print "Plane 1: ",
        Print Plane(PlaneNum)
    EndIf
EndFunction
PList Statement

Displays point data in memory for the current robot.

Syntax

1. PList
2. PList pointNumber
3. PList startPoint,
4. PList startPoint, endPoint

Parameters

- pointNumber: The number range is 0 to 999.
- startPoint: The start point number. The number range is 0 to 999.
- endPoint: The end point index. The number range is 0 to 999.

Return Values

Point data.

Description

Plist displays point data in memory for the current robot.

When there is no point data within the specified range of points, no data will be displayed.

When a start point number is specified larger than the end point number, then an error occurs.

(1) PList
   Displays the coordinate data for all points.

(2) PList pointNumber
   Displays the coordinate data for the specified point.

(3) PList startPoint,
   Displays the coordinate data for all points starting with startPoint.

(4) PList startPoint, endPoint
   Displays the coordinate data for all points starting with startPoint and ending with endPoint.

PList Statement Example

Display type depends on the robot type and existence of additional axes.
The following examples are for a Scara robot without additional axes.

Displays the specified point data:

```plaintext
> plist 1
P1   = XY( 200.000, 0.000, -20.000, 0.000 ) /R /0
>
```

Displays the point data within the range of 10 and 20. In this example, only three points are found in this range.

```plaintext
> plist 10, 20
P10  = XY( 290.000, 0.000, -20.000, 0.000 ) /R /0
P12  = XY( 300.000, 0.000, 0.000, 0.000 ) /R /0
P20  = XY( 285.000, 10.000, -30.000, 45.000 ) /R /0
>
```

Displays the point data starting with point number 10.
> `plist` 10,
P10 = `XY( 290.000, 0.000, -20.000, 0.000 ) /R /0
P12 = `XY( 300.000, 0.000, 0.000, 0.000 ) /R /0
P20 = `XY( 285.000, 10.000, -30.000, 45.000 ) /R /0
P30 = `XY( 310.000, 20.000, -50.000, 90.000 ) /R /0
Sets the local attribute for a point.

**Syntax**

```
PLocal(point) = localNumber
```

**Parameters**

- `point`: An integer value or `Pnumber` or `P(expr)` or point label.

  Cautions for compatibility
  No variables can be specified for `point` parameter
  To use variables, write `PLocal(P(varName))`.

- `localNumber`: An integer expression representing the new local number. Range is 0 to 15.

**See Also**

- PLocal Function

**PLocal Statement Example**

```
PLocal(pick) = 1
```
PLocal Function

Returns the local number for a specified point.

Syntax

\[ \text{PLocal}(point) \]

Parameters

- \( point \) An integer value or \( P\text{number} \) or \( P(\text{expr}) \) or point label.

Cautions for compatibility

- No variables can be specified for \( point \) parameter
- To use variables, write \( \text{PLocal}(P(\text{varName})) \).

Return Values

- Local number for specified point.

See Also

- PLocal

PLocal Function Example

```
Integer localNum

localNum = PLocal(pick)
```
Pls Function

Returns the current encoder pulse count for each joint at the current position.

Syntax

Pls(jointNumber)

Parameters

jointNumber  The specific joint for which to get the current encoder pulse count.
            The additional S axis is 8 and T axis is 9.

Return Values

Returns a number value representing the current encoder pulse count for the joint specified by jointNumber.

Description

Pls is used to read the current encoder position (or Pulse Count) of each joint. These values can be saved and then used later with the Pulse command.

See Also

CX, CY, CZ, CU, CV, CW, Pulse

Pls Function Example

Shown below is a simple example to get the pulse values for each joint and print them.

Function plstest
    Real t1, t2, z, u
    t1 = pls(1)
    t2 = pls(2)
    z = pls(3)
    u = pls(4)
    Print "T1 joint current Pulse Value: ", t1
    Print "T2 joint current Pulse Value: ", t2
    Print "Z joint current Pulse Value: ", z
    Print "U joint current Pulse Value: ", u
Fend
PNumber Function

Returns the point number associated with a point label.

Syntax

\[ \text{PNumber}(\text{pointLabel}) \]

Parameters

- **pointLabel**: A point label used in the current point file or string expression containing a point label.

See Also

- PDef Function, PLabel$ Function

PNumber Function Example

```plaintext
Integer pNum
String pointName$

pNum = PNumber(pick)
pNum = PNumber("pick")
pointName$ = "place"
pNum = PNumber(pointName$)
```
PosFound Function

Returns status of Find operation.

Syntax

PosFound

Return Values

True if position was found during move, False if not.

See Also

Find

PosFound Function Example

    Find Sw(5) = ON
    Go P10 Find
    If PosFound Then
        Go FindPos
    Else
        Print "Error: Cannot find the sensor signal."
    EndIf
Power Statement

Switches Power Mode to high or low and displays the current status.

Syntax
(1) Power { High | Low } [, Forced]
(2) Power

Parameters
High | Low  The setting can be High or Low. The default is Low.
Forced  Optional. This parameter is usually omitted.

Return Values
Displays the current Power status when parameter is omitted.

Description
Switches Power Mode to High or Low. It also displays the current mode status.

Low - When Power is set to Low, Low Power Mode is On. This means that the robot will run slow (below 250 mm/sec) and the servo stiffness is set light so as to remove servo power if the robot bumps into an object.

High - When Power is set to High, Low Power Mode is Off. This means that the robot can run at full speed with the full servo stiffness.

The following operations will switch to low power mode. In this case, speed and acceleration settings will be limited to the default value. The default value is described in the each manipulator specification table. See also the EPSON RC+ Users Guide: 2. Safety.

<table>
<thead>
<tr>
<th>Conditions to cause Power Low:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller Startup</td>
</tr>
<tr>
<td>Motor On</td>
</tr>
<tr>
<td>SFree, SLock, Brake</td>
</tr>
<tr>
<td>Reset, Reset Error</td>
</tr>
<tr>
<td>Stop button or QuitAll stops tasks</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Settings limited to the default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
</tr>
<tr>
<td>Accel</td>
</tr>
<tr>
<td>SpeedS</td>
</tr>
<tr>
<td>AccelS</td>
</tr>
</tbody>
</table>
**Notes**

**Low Power Mode (Power Low) and Its Effect on Max Speed:**
In low power mode, motor power is limited, and effective motion speed setting is lower than the default value. If, when in Low Power mode, a higher speed is specified from the Command window (directly) or in a program, the speed is set to the default value. If a higher speed motion is required, set Power High.

**High Power Mode (Power High) and Its Effect on Max Speed:**
In high power mode, higher speeds than the default value can be set.

**Forced Flag**
The power mode can be changed during robot operation (including the pause state).
If the mode is switched to high power mode while the robot is moving in low power mode, the subsequent motion will be changed to high speed with the specified speed.
If the mode is switched to low power mode while the robot is moving in high power mode, the overspeed error or low power torque error may occur.
Stop the robot and specify the Forced flag to switch to low power mode.

**See Also**

Accel, AccelS, Speed, SpeedS

**Power Statement Example**
The following examples are executed from the command window:

```
> Speed 50        ' Specifies high speed in Low Power mode
> Accel 100, 100  ' Specifies high accel
> Jump P1         ' Moves in low speed and low accel
> Speed           ' Displays current speed values
Low Power Mode
   50     50
   50     50
> Accel           ' Displays current accel values
Low Power Mode
   100    100
   100    100
   100    100
> Power High      ' Sets high power mode
> Jump P2         ' Moves robot at high speed
```
Power Function

Returns status of power.

Syntax

Power [(robotNumber)]

Parameters

robotNumber Specify the robot number to check the status by an integer value.
If omitted, currently selected robot will be used.

Return Values

0 = Power Low, 1 = Power High.

See Also

Power Statement

Power Function Example

If Power = 0 Then
    Print "Low Power Mode"
EndIf
PPIs Function

Return the pulse position of a specified joint value from a specified point.

Syntax

\[
PPIs \ (point, \ jointNumber)
\]

Parameters

- **point**: Point expression.
- **jointNumber**: Expression or numeric value specifying the joint number (integer from 1 to 9)
  The additional S axis is 8 and T axis is 9.

Return Values

Returns the calculated joint position (long value, in pulses).

See Also

- Agl, CX, CY, CZ, CU, CV, CW, Pagl

PPIs Function Example

```plaintext
Long pulses1

pulses1 = PPIs(P10, 1)
```
Print Statement

Outputs data to the current display window, including the Run window, Operator window, Command window, and Macro window.

Syntax
Print expression [,expression... ][,]

Parameters
expression Optional. A number or string expression.
, (comma) Optional. If a comma is provided at the end of the statement, then a CRLF will not be added.

Return Values
Variable data or the specified character string.

Description
Print displays variable data or the character string on the display device.

An end of line CRLF (carriage return and line feed) is automatically appended to each output unless a comma is used at the end of the statement.

Note
Make Sure Print is used with Wait or a motion within a loop
The Controller may freeze up if only Print is used in loop (loops with no Wait or no motion).
Be sure to use Print with Wait command or a motion command within a loop.

Bad example
Do
  Print "1234"
Loop

Good example
Do
  Print "1234"
  Wait 0.1
Loop

See Also
Print #

Print Statement Example
The following example extracts the U Axis coordinate value from a Point P100 and puts the coordinate value in the variable uvar. The value is then printed to the current display window.

Function test
  Real uvar
  uvar = CU(P100)
  Print "The U Axis Coordinate of " + Chr$(34) + "P100" + Chr$(34) + " is ", uvar
Fend
Print # Statement

Outputs data to the specified file, communications port, database, or device.

Syntax

Print #portNumber, expression [,expression... ] [ , ]

Parameters

portNumber  ID number representing a file, communications port, database, or device. File number can be specified in ROpen, WOpen, and AOpen statements. Communications port number can be specified in OpenCom (RS232) and OpenNet (TCP/IP) statements. Database number can be specified in OpenDB statement. Device ID integers are as follows.
21 RC+
24 TP (TP1 only)
20 TP3

expression  A numeric or string expression.

comma  Optional. If a comma is provided at the end of the statement, then a CRLF will not be added.

Description

Print # outputs variable data, numerical values, or character strings to the communication port or the device specified by portNumber.

Notes

Maximum data length

This command can handle up to 256 bytes. However, if the target is a database, it can handle up to 4096 bytes. If the target is the communications port (TCP/IP), it can handle up to 1024 bytes.

Exchange variable data with other controller

- When more than one string variable or both of numeric variable and string variable is specified, a comma ("," ) character has to be added expressly to the string data.

The following programs are examples to exchange the string variable and numeric variable between the Controllers using a communication port.

Sending end (Either pattern is OK.)

Print #PortNum, "$Status", InData, OutData
Print #PortNum, "$Status", ",", InData, OutData

Receiving end

Input #PortNum, Response$, InData, OutData

File write buffering

File writing is buffered. The buffered data can be written with Flush statement. Also, when closing a file with Close statement, the buffered data can be written.
Be sure to use Print # with Wait command or a motion command within a loop

Do not use only Print # in a loop

The Controller may freeze up if only Print # is used in loop (loops with no Wait or no motion).

Depending on the Controller status, information may not be displayed properly even if the Wait command or a motion command is used. If the output is TP1, set Wait time to 0.3 (seconds) or more. In other cases, set Wait time to 0.1 (seconds) or more.

**Bad example**

```plaintext
Do
  Print #24, "1234"
Loop
```

**Good example**

```plaintext
Do
  Print #24,"1234"
  Wait 0.3
Loop
```

---

**See Also**

Input#, Print

**Print # Statement Example**

The following are some simple Print # examples:

```plaintext
Function printex
  String temp$ 
  Print #1, "5"  ' send the character "5" to serial port 1 temp$ = "hello"
  Print #1, temp$ 
  Print #2, temp$ 
  Print #1 " Next message for " + Chr$(34) + "port 1" + Chr$(34)
  Print #2 " Next message for " + Chr$(34) + "port 2" + Chr$(34)
Fend
```
PTCLR Statement

Clears and initializes the peak torque for one or more joints.

**Syntax**

```plaintext
PTCLR [j1 [, j2 [, j3 [, j4 [, j5 [, j6 [, j7 [, j8 [, j9]]]]]]]]
```

**Parameters**

- **j1 – j9**: Integer expression representing the joint number. If no parameters are supplied, then the peak torque values are cleared for all joints. The additional S axis is 8 and T axis is 9. If non-existent joint number is supplied, an error occurs.

**Description**

PTCLR clears the peak torque values for the specified joints. You must execute PTCLR before executing PTRQ.

**See Also**

ATRQ, PTRQ

**PTCLR Statement Example**

**<Example 1>**

The following is the example to display the torque values of specified joints after clearing the peak torque values of all joints.

```plaintext
> ptclr
> go p1
> ptrq 1
  0.227
> ptrq 4
  0.227  0.118
  0.249  0.083
  0.000  0.000
>
```

**<Example 2>**

The following is the example to display the torque values of specified joints after clearing the peak torque values of J1, J4, and J5 for the vertical multi-axis robots.

```plaintext
> ptclr 4, 1, 5
> go p1
> ptrq 1
  0.227
> ptrq 4
  0.083
```
PTPBoost Statement

Specifies or displays the acceleration, deceleration and speed algorithmic boost parameter for small distance PTP (point to point) motion.

Syntax

(1) PTPBoost  \textit{boost} [, \textit{departBoost}] [, \textit{approBoost}]
(2) PTPBoost

Parameters

\begin{itemize}
\item \textit{boost} \quad Integer expression from 0 to 100.
\item \textit{departBoost} \quad Optional. Jump depart boost value. Integer expression from 0 to 100.
\item \textit{approBoost} \quad Optional. Jump approach boost value. Integer expression from 0 to 100.
\end{itemize}

Return Values

When parameters are omitted, the current PTPBoost settings are displayed.

Description

PTPBoost sets the acceleration, deceleration and speed for small distance PTP motion. It is effective only when the motion distance is small. The PTPBoostOK function can be used to confirm whether or not a specific motion distance to the destination is small enough to be affected by PTPBoost or not.

PTPBoost does not need modification under normal circumstances. Use PTPBoost only when you need to shorten the cycle time even if vibration becomes larger, or conversely when you need to reduce vibration even if cycle time becomes longer.

When the PTPBoost value is large, cycle time becomes shorter, but the positioning vibration increases. When PTPBoost is small, the positioning vibration becomes smaller, but cycle time becomes longer. Specifying inappropriate PTPBoost causes errors or can damage the manipulator. This may degrade the robot, or sometimes cause the manipulator life to shorten.

The PTPBoost value initializes to its default value when any one of the following is performed:

\begin{itemize}
\item Controller Startup
\item Motor On
\item SFree, SLock, Brake
\item Reset, Reset Error
\item Stop button or QuitAll stops tasks
\end{itemize}

See Also

PTPBoost Function, PTPBoostOK

PTPBoost Statement Example

\begin{verbatim}
PTPBoost 50, 30, 30
\end{verbatim}
PTPBoost Function

Returns the specified PTPBoost value.

Syntax

```
PTPBoost(paramNumber)
```

Parameters

- **paramNumber**: Integer expression which can have the following values:
  - 1: boost value
  - 2: jump depart boost value
  - 3: jump approach boost value

Return Values

Integer value from 0 to 100.

See Also

- PTPBoost Statement, PTPBoostOK

PTPBoost Function Example

```
Print PTPBoost(1)
```
PTPBoostOK Function

PTPBoostOK Function

Returns whether or not the PTP (Point to Point) motion from a current position to a target position is a small travel distance.

Syntax

PTPBoostOK(targetPos)

Parameters

targetPos

Point expression for the target position.

Return Values

True if is it possible to move to the target position from the current position using PTP motion, otherwise False.

Description

Use PTPBoostOK to the distance from the current position to the target position is small enough for PTPBoost to be effective.

See Also

PTPBoost

PTPBoostOK Function Example

If PTPBoostOK(P1) Then
  PTPBoost 50
EndIf
Go P1
PTPTime Function

Returns the estimated time for a point to point motion command without executing it.

Syntax

1. `PTPTime(destination, destArm, destTool)`
2. `PTPTime(start, startArm, startTool, destination, destArm, destTool)`

Parameters

- **start**: Point expression for the starting position.
- **destination**: Point expression for the destination position.
- **destArm**: Integer expression for the destination arm number.
- **destTool**: Integer expression for the destination tool number.
- **startArm**: Integer expression for the starting point arm number.
- **startTool**: Integer expression for the starting point tool number.

Return Values

Real value in seconds.

Description

Use PTPTime to calculate the time it would take for a point to point motion command (Go). Use syntax 1 to calculate time from the current position to the destination. Use syntax 2 to calculate time from a start point to a destination point.

The actual motion operation is not performed when this function is executed. The current position, arm, and tool settings do not change.

If the position is one that cannot be arrived at or if the arm or tool settings are incorrect, 0 is returned.

If a robot includes an additional axis and it is the servo axis, the function will consider the motion time of the additional axis.
If the additional axis is a PG axis, the motion time of the robot will be returned.

See Also

ATRQ, Go, PTRQ

PTPTime Function Example

```plaintext
Real secs

secs = PTPTime(P1, 0, 0, P2, 0, 1)
Print "Time to go from P1 to P2 is: ", secs

Go P1
secs = PTPTime(P2, 0, 1)
Print "Time to go from P1 to P2 is: ", secs
```
Perform a relative move of one joint in pulses.

**Syntax**

```
PTran joint, pulses
```

**Parameters**

- `joint` Integer expression representing which joint to move.
  The additional S axis is 8 and T axis is 9.
- `pulses` Integer expression representing the number of pulses to move.

**Description**

Use PTran to move one joint a specified number of pulses from the current position.

**See Also**

- Go, JTran, Jump, Move

**PTran Statement Example**

```
PTran 1, 2000
```
**PTRQ Statement**

Displays the peak torque for the specified joint.

**Syntax**

```
PTRQ [jointNumber]
```

**Parameters**

- `jointNumber` Optional. Integer expression representing the joint number.
  The additional S axis is 8 and T axis is 9.

**Return Values**

Displays current peak torque values for all joints.

**Description**

Use PTRQ to display the peak torque value for one or all joints since the PTCLR statement was executed.

Peak torque is a real number from 0 to 1.

**See Also**

ATRQ, PTCLR, PTRQ Function

**PTRQ Statement Example**

```
> ptclr
> go pl
> ptrq 1
   0.227
> ptrq
   0.227  0.118
   0.249  0.083
   0.000  0.000
>
```
PTRQ Function

Returns the peak torque for the specified joint.

Syntax

PTRQ(jointNumber)

Parameters

jointNumber Integer expression representing the joint number.
The additional S axis is 8 and T axis is 9.

Return Values

Real value from 0 to 1.

See Also

ATRQ, PTCLR, PTRQ Statement

PTRQ Function Example

This example uses the PTRQ function in a program:

```plaintext
Function DisplayPeakTorque
    Integer i
    Print "Peak torques:"
    For i = 1 To 4
        Print "Joint ", i, " = ", PTRQ(i)
    Next i
End
```
Pulse Statement

Moves the robot arm using point to point motion to the point specified by the pulse values for each joint.

Syntax

(1) **Pulse** \(J_1, J_2, J_3, J_4, J_5, J_6, J_7, J_8, J_9\)

(2) **Pulse**

Parameters

- **\(J_1 \sim J_4\)**: The pulse value for each of the first four joints. The pulse value has to be within the range defined by the Range instruction and should be an integer or long expression.
- **\(J_5, J_6\)**: Optional. For 6-axis robots (including N series) and Joint type 6-axis robots.
- **\(J_7\)**: Optional. For Joint type 7-axis robots.
- **\(J_8, J_9\)**: Optional. For the additional axis.

Return Values

When parameters are omitted, the pulse values for the current robot position are displayed.

Description

Pulse uses the joint pulse value from the zero pulse position to represent the robot arm position, rather than the orthogonal coordinate system. The Pulse instruction moves the robot arm using Point to Point motion.

The Range instruction sets the upper and lower limits used in the Pulse instruction.

Note

**Make Sure Path is Obstacle Free Before Using Pulse**

Unlike Jump, Pulse moves all axes simultaneously, including Z joint raising and lowering in traveling to the target position. Therefore, when using Pulse, take extreme care so that the hand can move through an obstacle free path.

Potential Error

**Pulse value exceeds limit:**

If the pulse value specified in Pulse instruction exceeds the limit set by the Range instruction, an error will occur.

See Also

Go, Accel, Range, Speed, Pls, Pulse Function

Pulse Statement Example

Following are examples on the Command window:

This example moves the robot arm to the position which is defined by each joint pulse.

\[> \text{pulse} \ 16000, 10000, -100, 10\]

This example displays the pulse numbers of 1st to 4th axes of the current robot arm position.

\[> \text{pulse} \]

\[\text{PULSE: 1: 27306 pls 2: 11378 pls 3: -3072 pls 4: 1297 pls}\]

[564] EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
Pulse Function

Returns a robot point whose coordinates are specified in pulses for each joint.

Syntax

\[ \text{Pulse} \left( J1, J2, J3, J4, J5, J6, J7, J8, J9 \right) \]

Parameters

- **J1 ~ J4**: The pulse value for joints 1 to 4. The pulse value must be within the range defined by the Range instruction and should be an integer or long expression.
- **J5, J6**: Optional. For 6-axis robots (including N series) and Joint type 6-axis robots.
- **J7**: Optional. For Joint type 7-axis robots.
- **J8, J9**: Optional. For the additional axis.

Return Values

A robot point using the specified pulse values.

See Also

Go, JA, Jump, Move, Pulse Statement, XY

Pulse Function Example

\[ \text{Jump } \text{Pulse}(1000, 2000, 0, 0) \]
QP Statement

Switches Quick Pause Mode On or Off and displays the current mode status.

Syntax
(1) QP  { On | Off }
(2) QP

Parameters
On | Off            Quick Pause can be either On or Off.

Return Values
Displays the current QP mode setting when parameter is omitted.

Description
If during motion command execution either the Pause switch is pressed, or a pause signal is input to the controller, quick pause mode determines whether the robot will stop immediately, or will Pause after having executed the motion command.

Immediately decelerating and stopping is referred to as a “Quick Pause”.

With the On parameter specified, QP turns the Quick Pause mode On.
With the Off parameter specified, QP turns the Quick Pause mode Off.

QP displays the current setting of whether the robot arm is to respond to the Pause input by stopping immediately or after the current arm operation is completed. QP is simply a status instruction used to display whether Quick Pause mode is on or off.

Notes
Quick pause mode defaults to on after power is turned on:
The Quick Pause mode set by the QP instruction remains in effect after the Reset instruction. However, when the PC power or Drive Unit power is turned off and then back on, Quick Pause mode defaults to On.

QP and the Safe Guard Input:
Even if QP mode is set to Off, if the Safe Guard Input becomes open the robot will pause immediately.

See Also
Pause

QP Statement Example
This Command window example displays the current setting of whether the robot arm is to stop immediately on the Pause input. (i.e. is QP mode set On or Off)

```
> qp
QP ON

> qp on  ' Sets QP to Quick Pause Mode
>```
QPDeceIR Statement

Sets the deceleration speed of quick pause for the change of tool orientation during the CP motion.

Syntax
(1) QPDeceIR
(2) QPDeceIR

Parameters

QPDeceIR  Real value representing the deceleration speed of quick pause during the CP motion (deg/sec²).

Result
If omitted the parameter, the current QPDeceIR set value will be displayed.

Description
QPDeceIR statement is enabled when the ROT parameter is used in the Move, Arc, Arc3, BMove, TMove, and Jump3CP statements.

While quick pause is executed in these statements, a joint acceleration error may occur. This is because the deceleration speed of quick pause that is automatically set in a normal quick pause is over the joint allowable deceleration speed. Specifically, the error is likely to occur when the AccelR value in the CP motion is too high or jogging the robot near a singularity. In these cases, use the QPDeceIR and set a lower quick pause deceleration speed. But if the setting is too low, the distance for quick pause will increase. Therefore, set the possible value. Normally, you don’t need to set QPDeceIR.

You cannot use values lower than the deceleration speed of orientation change in the CP motion set with QPDeceIR and AccelR. If you do, a parameter out of range error occurs.

Also, after you set QPDeceIR, if a higher value than the set QP deceleration speed is set with the AccelR, the QPDeceIR will automatically set the QP deceleration speed same as the deceleration speed set with the AccelR.

The QPDeceIR Statement value initializes to the default max deceleration speed when any one of the following conditions occurs:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

See Also
QPDeceIR Function, QPDeceIR, AccelR

QPDeceIR Statement Example
The following program sets the QPDeceIR of the Move statement.

```
Function QPDeceIRTest
  AccelR 3000
  QPDeceIR 4000
  SpeedR 100
  Move P1 ROT
  ;
  Fend
```
QPDecelR Function

Returns the set deceleration speed of quick pause for the change of tool orientation during the CP motion.

Syntax

QPDecelR

Return Values

Real value that contains the set deceleration speed of quick pause for the tool orientation change in the CP motion (deg/s²).

See Also

QPDecelR, QPDecelS Function

QPDecelR Function Example

Real savQPDecelR

savQPDecelR = QPDecelR
QPDecelS Statement

Sets the deceleration speed of quick pause in the CP motion.

Syntax
(1) QPDecelS  QPDecelS [, departDecel, approDecel ]
(2) QPDecelS

Parameters
- **QPDecelS**: Real value that specifies the deceleration speed of quick pause in the CP motion. (mm/sec²)
- **departDecel**: Real value that specifies the deceleration speed of quick pause in the Jump3 depart motion (mm/sec²)
- **approDecel**: Real value that specifies the deceleration speed of quick pause in the Jump3 approach motion (mm/sec²)

Return Values
If omitted the parameter, the current QPDecelS set value is displayed.

Description
While quick pause is executed in the CP motion, a joint acceleration error may occur. This is because the deceleration speed of quick pause that is automatically set in a normal quick pause is over the joint allowable deceleration speed. Specifically, the error is likely to occur when the AccelS value in the CP motion is too high or jogging the robot near a singularity. In these cases, use the QPDecelS and set a lower quick pause deceleration speed. But if the setting is too low, the distance for quick pause will increase. Therefore, set the possible value. Normally, you don’t need to set QPDecelS.

You cannot use values lower than the deceleration speed of the CP motion set with AccelS. If you do, a parameter out of range error occurs.
Also, after you set QPDecelS, if a higher value than the set QP deceleration speed is set with the AccelS, the QPDecelS will automatically set the QP deceleration speed same as the deceleration speed set with the AccelS.

The QPDecelS Statement value initializes to the default max deceleration speed when any one of the following conditions occurs:

```
Controller Startup
Motor On
SFree, SLock, Brake
Reset, Reset Error
Stop button or QuitAll stops tasks
```

See Also
QPDecelS Function, QPDecelR, AccelS

QPDecelS Statement Example
The following program sets the QPDecelS of the Move statement.

```plaintext
Function QPDecelTest
  AccelS 3000
  QPDecelS 4000
  SpeedS 100
  Move P1
  .
  .
  .
Fend
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 569
QPDecelS Function

Returns the set deceleration speed of quick pause during the CP motion.

Syntax

```
QPDecelS ( paramNumber )
```

Parameters

- `paramNumber` Integer expression specifying the one of the following values.
  - 1: Quick pause deceleration speed during the CP motion
  - 2: Quick pause deceleration speed in depart motion during the Jump3 and Jump3CP
  - 3: Quick pause deceleration speed in approach motion during the Jump3 and Jump3CP

Return Values

Real value representing the quick pause deceleration speed (mm/s²).

See Also

QPDecelS, QPDecelR Function

QPDecelS Function Example

```
Real savQPDecelS

savQPDecelS = QPDecelS(1)
```
Quit Statement

Terminates execution of a specified task or all tasks.

Syntax

```
Quit \{ taskIdentifier | All \}
```

Parameters

- **taskIdentifier**: Task name or integer expression representing the task number.
  Task name is a function name used in an Xqt statement or a function started from the Run window or Operator window.
  
  Task number range is:
  - Normal tasks: 1 to 32
  - Background task: 65 to 80
  - Trap tasks: 257 to 267

- **All**: Specifies this parameter if all tasks except the background task should be terminated.

Description

Quit stops the tasks that are currently being executed, or that have been temporarily suspended with Halt.

Quit also stops the task when the specified task is NoPause task, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), or the background tasks.

Quit All stops all tasks including the tasks above other than the background tasks.

Quit All sets the robot control parameter as below:

**Robot Control parameter**

- Current robot Speed, SpeedR, SpeedS (Initialized to default values)
- Current robot QPDecelR, QPDecelS (Initialized to default values)
- Current robot LimZ parameter (Initialized to 0)
- Current robot CP parameter (Initialized to Off)
- Current robot SoftCP parameter (Initialized to Off)
- Current robot Fine (Initialized to default values)
- Current robot Power Low (Low Power Mode set to On)
- Current robot PTPBoost (Initialized to default values)
- Current robot TCLim, TCSpeed (Initialized to default values)
- Current robot PgLSpeed (Initialized to default values)

See Also

- Exit, Halt, Resume, Xqt

Quit Statement Example

This example shows two tasks that are terminated after 10 seconds.

```
Function main
    Xqt winc1   ' Start winc1 function
    Xqt winc2   ' Start winc2 function
    Wait 10
    Quit winc1 ' Terminate task winc1
    Quit winc2 ' Terminate task winc2
    Fend

Function winc1
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 571
Quit Statement

Do
  On 1; Wait 0.2
  Off 1; Wait 0.2
Loop
Fend

Function winc2
  Do
    On 2; Wait 0.5
    Off 2; Wait 0.5
  Loop
Fend
RadToDeg Function

Converts radians to degrees.

Syntax

```
RadToDeg(radians)
```

Parameters

- `radians` Real expression representing the radians to convert to degrees.

Return Values

A double value containing the number of degrees.

See Also

ATan, ATan2, DegToRad Function

RadToDeg Function Example

```
s = Cos(RadToDeg(x))
```
Randomize Statement

Initializes the random-number generator.

Syntax

1. Randomize seedValue
2. Randomize

Parameters

seedValue Specify a real value (0 or more) to be basis to retrieve a random number.

See Also

Rnd Function

Randomize Statement Example

```
Function main
  Real r
  Randomize
  Integer randNum

  randNum = Int(Rnd(10)) + 1
  Print "Random number is:", randNum
Fend
```
Range Statement

Specifies and displays the motion limits for each of the servo joints.

Syntax
(1) Range j1Min, j1Max, j2Min, j2Max, j3Min, j3Max, j4Min, j4Max
    [, j5Min, j5Max, j6Min, j6Max]
    [, j7Min, j7Max]
    [, j8Min, j8Max, j9Min, j9Max]

(2) Range

Parameters
j1Min The lower limit for joint 1 specified in pulses.
j1Max The upper limit for joint 1 specified in pulses.
j2Min The lower limit for joint 2 specified in pulses.
j2Max The upper limit for joint 2 specified in pulses.
j3Min The lower limit for joint 3 specified in pulses.
j3Max The upper limit for joint 3 specified in pulses.
j4Min The lower limit for joint 4 specified in pulses.
j4Max The upper limit for joint 4 specified in pulses.
j5Min Optional for 6-Axis robots (including N series) and Joint type 6-axis robots.
    The lower limit for joint 5 specified in pulses.
j5Max Optional for 6-Axis robots (including N series) and Joint type 6-axis robots.
    The upper limit for joint 5 specified in pulses.
j6Min Optional for 6-Axis robots (including N series) and Joint type 6-axis robots.
    The lower limit for joint 6 specified in pulses.
j6Max Optional for 6-Axis robots (including N series) and Joint type 6-axis robots.
    The upper limit for joint 6 specified in pulses.
j7Min Optional for Joint type 7-axis robots. The lower limit for joint 7 specified in pulses.
j7Max Optional for Joint type 7-axis robots. The upper limit for joint 7 specified in pulses.
j8Min Optional for the additional S axis. The lower limit for joint 8 specified in pulses.
j8Max Optional for the additional S axis. The upper limit for joint 8 specified in pulses.
j9Min Optional for the additional T axis. The lower limit for joint 9 specified in pulses.
j9Max Optional for the additional T axis. The upper limit for joint 9 specified in pulses.

Return Values
Displays the current Range values when Range is entered without parameters

Description
Range specifies the lower and upper limits of each motor joint in pulse counts. These joint limits are specified in pulse units. This allows the user to define a maximum and minimum joint motion range for each of the individual joints. XY coordinate limits can also be set using the XYLim instruction.

The initial Range values are different for each robot. The values specified by this instruction remain in effect even after the power is switched off. When parameters are omitted, the current Range values are displayed.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.
Range Statement

Potential Errors

Attempt to Move Out of Acceptable Range
If the robot arm attempts to move through one of the joint limits error will occur.

Axis Does Not Move
If the lower limit pulse is equal to or greater than the upper limit pulse, the joint does not move.

Note

Range of the lower/upper limits of Joint #6 in pulse differs depending on manipulator model
C4 : -419430399 to 419430399
C8, C12, N2, N6 : -26847955 to 26847955

See Also
JRange, SysConfig, XYLim

Range Statement Example
This simple example from the command window displays the current range configurations and then changes them.

```
> range
-18205, 182045, -82489, 82489, -36864, 0, -46695, 46695
>
> range 0, 32000, 0, 32224, -10000, 0, -40000, 40000
>```

Read Statement

Reads characters from a file or communications port.

**Syntax**

```
Read #portNumber, stringVar$, count
```

**Parameters**

- **portNumber**
  - ID number representing a file or communications port to read from.
  - File number can be specified in ROpen, WOpen, and AOpen statements.
  - Communication port number can be specified in OpenCom (RS-232C) or OpenNet (TCP/IP) statements.

- **stringVar$**
  - Name of a string variable that will receive the character string.

- **count**
  - Maximum number of bytes to read.

**See Also**

ChkCom, ChkNet, OpenCom, OpenNet, Write

**Read Statement Example**

```plaintext
Integer numOfChars
String data$

numOfChars = ChkCom(1)

If numOfChars > 0 Then
   Read #1, data$, numOfChars
EndIf
```
ReadBin Statement

Reads binary data from a file or communications port.

Syntax

ReadBin #portNumber, var
ReadBin #portNumber, array(), count

Parameters

portNumber | ID number representing a file or communications port to read from. File number can be specified in BOpen statement. Communication port number can be specified in OpenCom (RS-232C) or OpenNet (TCP/IP) statements.

var | Name of a byte, integer, or long variable that will receive the data.

array() | Name of a byte, integer, or long array variable that will receive the data. Specify a one dimension array variable.

count | Specify the number of bytes to read. The specified count has to be less than or equal to the number of array elements and also smaller than 256 bytes. If the communication port (TCP/IP) is the subject, the count has to be less than or equal to the number of array and also smaller than 1024 bytes.

See Also

Write, WriteBin

ReadBin Statement Example

Integer data
Integer dataArray(10)

numOfChars = ChkCom(1)

If numOfChars > 0 Then
    ReadBin #1, data
EndIf

NumOfChars = ChkCom(1)
If numOfChars > 10 Then
    ReadBin #1, dataArray(), 10
EndIf
Real Statement

Declares variables of type Real (4 byte real number).

Syntax

Real varName [(subscripts)] [, varName [(subscripts)]]...

Parameters

varName Variable name which the user wants to declare as type Real.

subscripts Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows

(ubound1, [ubound2], [ubound3])

ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

When specifying the upper bound value, make sure the number of total elements is within the range shown below:

<table>
<thead>
<tr>
<th>Type</th>
<th>Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local variable</td>
<td>2,000</td>
</tr>
<tr>
<td>Global Preserve variable</td>
<td>4,000</td>
</tr>
<tr>
<td>Global variable and module variable</td>
<td>100,000</td>
</tr>
</tbody>
</table>

Description

Real is used to declare variables as type Real. Local variables should be declared at the top of a function. Global and module variables must be declared outside functions.

Number of valid digits are six digits for Real type.

See Also

Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Short, String, UByte, UInt32, UInt64, UShort

Real Statement Example

The following example shows a simple program which declares some variables using Real.

```plaintext
Function realtest
    Real var1
    Real A(10)     'Single dimension array of real
    Real B(10, 10) 'Two dimension array of real
    Real C(5, 5, 5) 'Three dimension array of real
    Real arrayVar(10)
    Integer i
    Print "Please enter a Real Number:" Input var1
    Print "The Real variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter a Real Number:" Input arrayVar(i)
        Print "Value Entered was ", arrayVar(i)
    Next i
End
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
RealAccel Function

Returns the Accel value adjusted by OLAccel.

Syntax

RealAccel\( (paramNumber) \)

Parameters

\( paramNumber \)

Integer expression which can have the following values:
1: acceleration specification value
2: deceleration specification value
3: depart acceleration specification value for Jump
4: depart deceleration specification value for Jump
5: approach acceleration specification value for Jump
6: approach deceleration specification value for Jump

Return Values

Integer 1% or more

Usage

By using RealAccel, the maximum acceleration speed with which the robot can operate continuously can be acquired.

Steps are as follows:
1. Operate the robot with the OLAccel command On.
2. Execute the OLRate command and check if the overload ratio rises.
3. If the overload ratio rises, auto adjustment begins when the overload ratio exceeds 0.5.
4. After a certain period of time has passed, execute the OLRate command and check that the overload ratio does not rise.
5. After checking that the overload ratio does not rises, execute the RealAccel function.
6. The value returned by the RealAccel function is the maximum acceleration speed that the robot can operate continuously in the step (1).

* If the RealAccel function is executed while the overload ratio is rising, maximum acceleration speed of continuous motion cannot be acquired.
* If the overheat error occurs, maximum acceleration speed of continuous motion cannot be acquired by the above procedure.

See Also

Accel, OLAccel, OLRate

RealAccel Function Example

Following is the example of the RealAccel function used in the program.

```plaintext
Integer RealAccel1, RealDecel1

Accel 100, 100
OLAccel on

' Acquire the current acceleration speed.
RealAccel1 = RealAccel (1)
RealDecel1 = RealAccel (2)

Display the current acceleration speed
Print RealAccel1

Display the current deceleration speed
Print RealDecel1
```
RealPls Function

Returns the pulse value of the specified joint.

**Syntax**

```
RealPls(jointNumber)
```

**Parameters**

- `jointNumber`  
  The specific joint for which to get the current pulse count.  
  The additional S axis is 8 and T axis is 9.

**Return Values**

Returns an integer value representing the current encoder pulse count for the joint specified by `jointNumber`.

**Description**

RealPls is used to read the current encoder position (or Pulse Count) of each joint. These values can be saved and then used later with the Pulse command.

**See Also**

CX, CY, CZ, CU, CV, CW, Pulse

**RealPls Function Example**

```
Function DisplayPulses

    Long joint1Pulses

    joint1Pulses = RealPls(1)
    Print "Joint 1 Current Pulse Value: ", joint1Pulses
End
```
RealPos Function

Returns the current position of the specified robot.

**Syntax**

```plaintext
RealPos
```

**Return Values**

A robot point representing the current position of the specified robot.

**Description**

RealPos is used to read the current position of the robot.

**See Also**

CurPos, CX, CY, CZ, CU, CV, CW, RealPIs

**RealPos Function Example**

```plaintext
Function ShowRealPos
    Print RealPos
Fend

P1 = RealPos
```
RealTorque Function

Returns the current torque instruction value of the specified joint.

Syntax

```
RealTorque(jointNumber)
```

Parameters

- **jointNumber**: Specifies the joint number to acquire the torque instruction value using an expression or numeric value. The additional S axis is 8 and T axis is 9.

Return Values

Returns the real value (-1 to 1) representing the proportion in the maximum torque on current power mode. The positive value means the positive direction of the joint angle and the negative value means the negative direction.

See also

- TC, TCSpeed, TCLim

RealTorque Function Example

```
Print "Current Z axis torque instruction value (SCARA):",
RealTorque(3)
```
Recover Statement

Executes safeguard position recovery and returns status. This is for the experienced user and you need to understand the command specification before use.

Syntax

1. **Recover** `robotNumber | All`
2. **Recover** `robotNumber | All, WithMove | WithoutMove`

Parameters

- `robotNumber`: Robot number that you want to execute recovery for. If omitted, all robots are executed recovery.
- `All`: All robots execute recovery. If omitted, same as `All`.
- `WithMove`: A constant whose value is 0. Turns motor on and executes safeguard position recovery. If omitted, same as `WithMove`.
- `WithoutMove`: A constant whose value is 1. Turns the robot motor on. Not usually used. Realizes the special recovery with AbortMotion.

Return Values

Boolean value. True if recover was completed, False if not.

Description

To execute Recover statement from a program, you need to set the [Enable advanced task commands] checkbox in the [Setup] menu-[System Configuration]-[Controller]-[Preferences] page of the EPSON RC+.

Recover can be used after the safeguard is closed to turn on the robot motors and move the robot back to the position it was in when the safeguard was open with low power PTP motion. After Recover has completed successfully, you can execute the Cont method to continue the cycle.

When more than one robot is used in the controller and `All` is specified, all robots are recovered.

See Also

AbortMotion, Cont, Recover Function, RecoverPos
Recover Statement Example

CAUTION

■ When executing the Recover command from a program, you must understand the command specification and confirm that the system has the proper conditions for this command. Improper use such as continuous execution of a command within a loop may deteriorate the system safety.

Function main
  Xqt 2, monitor, NoPause
  Do
    Jump P1
    Jump P2
  Loop
  Fend

Function monitor
  Do
    If Sw(SGOpenSwitch) = On then
      Wait Sw(SGOpenSwitch) = Off and Sw(RecoverSwitch) = On
      Recover All
    EndIf
  Loop
  Fend
Recover Function

Executes safeguard position recovery and returns status. This is for the experienced user and you need to understand the command specification before use.

Syntax
(1) Recover
(2) Recover ( robotNumber | All )
(3) Recover ( robotNumber | All , WithMove | WithoutMove )

Parameters
robotNumber Robot number that you want to execute recovery for. If omitted, all robots are executed recovery.
All All robots execute recovery If omitted, same as All.
WithMove A constant whose value is 0. Turns motor on and executes safeguard position recovery. If omitted, same as WithMove.
WithoutMove A constant whose value is 1. Turns the robot motor on. Not usually used. Realizes the special recovery with AbortMotion.

Return Values
Boolean value. True if recover was completed, False if not.

Description
To execute Recover statement from a program, you need to select the [Enable advanced task commands] checkbox in the EPSON RC+ [Setup]-[System Configuration]-[Controller]-[Preferences].

Recover can be used after the safeguard is closed to turn on the robot motors and move the robot back to the position it was in when the safeguard was open with low power PTP motion. After Recover has completed successfully, you can execute the Cont method to continue the cycle.

When more than one robot is used in the controller and All is specified, all robots are recovered.

CAUTION
When executing the Recover command from a program, you must understand the command specification and confirm that the system has the proper conditions for this command. Improper use such as continuous execution of a command within a loop may deteriorate the system safety.

See Also
AbortMotion, Cont, Recover Statement, RecoverPos
Recover Function Example

Boolean sts
Integer answer

sts = Recover
If sts = True Then
    MsgBox "Ready to continue", MB_ICONQUESTION + MB_YESNO, "MyProject", answer
    If answer = IDYES Then
        Cont
    EndIf
EndIf
RecoverPos Function

Returns the position where a robot was in when safeguard was open.
This is for the experienced and you need to understand the command specification before use.

Syntax

RecoverPos ([robotNumber])

Parameters

robotNumber Integer value that specifies a robot number
If omitted, the current robot number is used.

Return Values

Returns the position the specified robot was in when the safeguard was open.
In the case where the safeguard was not open or the robot has completed the recovery, the coordinates of
the returned point data are 0.

Description

This function returns the robot recovery position when using the Cont or Recover commands.

See Also

AbortMotion, Cont, Recover, Recover Function, RealPos

RecoverPos Function Example

If the straight distance of recovery is less than 10 mm, it executes recovery. If more than 10 mm, it finishes
the program.

If Dist(RecoverPos, RealPos) < 10 Then
  Recover All
Else
  Quit All
EndIf
Redim Statement

Redimension an array at run-time.

Syntax

Redim [Preserve] arrayName (subscripts)

Parameters

Preserve Optional. Specifies to preserve the previous contents of the array. If omitted, the array will be cleared.

arrayName Name of the array variable; follows standard variable naming conventions. The array must have already been declared.

subscripts Optional. New dimensions of an array variable may be declared. You must supply the same number of dimensions as when the variable was declared. The subscripts syntax is as follows:

(ubound1, [ubound2], [ubound3])

ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

When specifying the upper bound value, make sure the number of total elements is within the range shown below:

<table>
<thead>
<tr>
<th></th>
<th>Others than String</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local variable</td>
<td>2,000</td>
<td>200</td>
</tr>
<tr>
<td>Global Preserve variable</td>
<td>4,000</td>
<td>400</td>
</tr>
<tr>
<td>Global variable and module variable</td>
<td>100,000</td>
<td>10,000</td>
</tr>
</tbody>
</table>

Description

Use Redim to change an array’s dimensions at run time. Use Preserve to retain previous values. The array variable declared by Byref cannot use Redim.

Frequent Redim will decrease the speed of program execution. Especially, we recommend using the minimum of Redim for the global preserve variables.

See Also

UBound
Redim Statement Example

Integer i, numParts, a(0)

Print "Enter number of parts "
Input numParts

Redim a(numParts)

For i=0 to UBound(a)
    a(i) = i
Next

' Redimension the array with 20 more elements
Redim Preserve a(numParts + 20)

' The first element values are retained
For i = 0 to UBound(a)
    Print a(i)
Next
Rename Statement

Renames a file.

Syntax

```
Rename oldFileName, newFileName
```

Parameters

- `oldFileName` String expression containing the path and name of the file to rename.
  See ChDisk for the details.
- `newFileName` The new name to be given to the file specified by `oldFileName`.
  See ChDisk for the details.

Description

Changes name of specified file `oldFileName` to `newFileName`.

If path is omitted, Rename searches for `oldFileName` in the current directory.

Rename is only enabled when `oldFileName` and `newFileName` are specified in the same drive.

A file may not be renamed to a filename that already exists in the same path.

Wildcard characters are not allowed in either `oldFileName` or `newFileName`.

See Also

Copy

Rename Statement Example

Example from the command window:

```
> Rename A.PRG  B.PRG
```
**RenDir Statement**

Rename a directory.

**Syntax**

```plaintext
RenDir oldDirName As String, newDirName As String
```

**Parameters**

- **oldDirName**
  A string expression specifying the path and name of the directory to rename.

- **newDirName**
  A string expression specifying the path and new name to be given to the directory specified by `oldDirName`.
  
  See ChDisk for the details of path.

**Description**

The same path used for `oldDirName` must be included for `newDirName`.

If both paths of the parameters above are omitted and directory name is only specified, the current directory is specified.

Wildcard characters are not allowed in either `oldDirName` or `newDirName`.

**Note**

- This statement is executable only with the PC disk.

**See Also**

- MkDir

**RenDir Statement Example**

```
RenDir "c:\mydata", "c:\mydatal"
```
Reset Statement

Resets the controller into an initialized state.

Syntax

(1) Reset
(2) Reset Error

Description

Reset resets the items shown below.
Reset Error finishes all non-background tasks and resets the error status and robot control parameters.
To execute the Reset Error statement from programs you need to set the [Enable advanced task commands] preference in the [Setup]-[System Configuration]-[Controller]-[Preference] page of the EPSON RC+.

- Emergency Stop Status (reset by Reset only)
- Error status
- Output Bits (reset by Reset only)
  - All Output Bits output set to Off except the I/O for Remote.
  - User can set Option Switch to turn this feature off.
- Robot Control parameter
  - Speed, SpeedR, SpeedS (Initialized to default values)
  - Accel, AccelR, AccelS (Initialized to default values)
  - QPDecelR, QPDecelS (Initialized to default values)
  - LimZ parameter (Initialized to 0)
  - CP parameter (Initialized to Off)
  - SoftCP parameter (Initialized to Off)
  - Fine (Initialized to default values)
  - Power Low (Low Power Mode set to On)
  - PTPBoost (Initialized to default values)
  - TCLim, TCSpeed (Initialized to default values)
  - PlgL.Speed (Initialized to default values)

For servo related errors, Emergency Stop status, and any other conditions requiring a Reset, no command other than Reset will be accepted. In this case first execute Reset, then execute other processing as necessary. For example, after an emergency stop, first verify safe operating conditions, execute Reset, and then execute Motor On.
Critical error state will not be canceled by Reset.
When critical error occurs, turn Off the controller and solve the cause of the error.
The Reset Statement cannot be executed from a background task or tasks started with the Trap Emergency or Trap Error. Emergency Stop status cannot be reset from programs.

Note

Reset Outputs Preference

([Setup]-[System Configuration]-[Controller]-[Preferences]) If the [Reset command turns off outputs] checkbox is selected, all outputs will be turned OFF when the Reset command is issued.
This is important to remember when wiring the system such that turning the outputs off should not cause tooling to drop or similar situations.

See Also

Accel, AccelS, Fine, LimZ, Motor, Off, On, PTPBoost, SFree, SLock, Speed, SpeedS

Reset Statement Example

Example from the command window.

> reset
>
ResetElapsedTime Statement

Resets the takt time measurement timer used in ElapsedTime Function.

Syntax

ResetElapsedTime

Description

Resets and starts the takt time measurement timer.

See Also

ElapsedTime Function

ResetElapsedTime Statement Example

```plaintext
ResetElapsedTime  ' Resets the takt time measurement timer
For i = 1 To 10   ' Executes 10 times
    GoSub Cycle
Next
Print ElapsedTime / 10 ' Measures a takt time and displays it
```
Restart Statement

Restarts the current main program group.  
This command is for the experienced user and you should understand the command specification before use.

Syntax

```
Restart
```

Description

Restart stops all tasks and re-executes the last main program group that was running. Background tasks continue to run.
All Trap settings are reset and even if Restart stops tasks, it doesn’t execute Trap Abort.
Restart resets the Pause status.
If you execute Restart during error status, reset the error first using a method such as the Reset Error statement.
Restart cannot be used during Emergency Stop status as it causes an error. Emergency Stop status cannot be reset from programs.

CAUTION

When executing the Restart command from a program, you must understand the command specification and confirm that the system has the proper conditions for this command. Improper use such as continuous execution of a command within a loop may deteriorate the system safety.

See Also

Quit, Reset, Trap, Xqt

Restart Statement Example

```
Function main
  Trap Error Xqt eTrap
  Motor On
  Call PickPlac
  Fend

Function eTrap

  Wait Sw(ERresetSwitch)
  Reset Error
  Wait Sw(RestartSwitch)
  Restart

  Fend
```
Resume Statement

Continues a task which was suspended by the Halt instruction.

Syntax
Resume { taskIdentifier | All }

Parameters

- **taskIdentifier**: Task name or integer expression representing the task number. Task name is a function name used in an Xqt statement or a function started from the Run window or Operator window.

  Task number range is:
  - Normal tasks: 1 to 32
  - Background task: 65 to 80
  - Trap tasks: 257 to 267

- **All**: Specifies that all tasks should be resumed.

Description
Resume continues the execution of the tasks suspended by the Halt instruction.

See Also
Halt, Quit, Xqt

Resume Statement Example
This shows the use of Resume instruction after the Halt instruction.

```plaintext
Function main
  Xqt 2, flicker  ' Execute flicker as task 2
  Do
    Wait 3        ' Allow flicker to execute for 3 seconds
    Halt flicker   ' Halt the flicker task
    Wait 3
    Resume flicker ' Resume the flicker task
  Loop
Fend

Function flicker
  Do
    On 1
    Wait 0.2
    Off 1
    Wait 0.2
  Loop
Fend
```
Return Statement

The Return statement is used with the GoSub statement. GoSub transfers program control to a subroutine. Once the subroutine is complete, Return causes program execution to continue at the line following the GoSub instruction which initiated the subroutine.

Syntax
Return

Description
The Return statement is used with the GoSub statement. The primary purpose of the Return statement is to return program control back to the instruction following the GoSub instruction which initiated the subroutine in the first place.

The GoSub instruction causes program control to branch to the user specified statement line number or label. The program then executes the statement on that line and continues execution through subsequent line numbers until a Return instruction is encountered. The Return instruction then causes program control to transfer back to the line which immediately follows the line which initiated the GoSub in the first place. (i.e. the GoSub instruction causes the execution of a subroutine and then execution Returns to the statement following the GoSub instruction.)

Potential Error
Return Found Without GoSub

A Return instruction is used to "return" from a subroutine back to the original program which issued the GoSub instruction. If a Return instruction is encountered without a GoSub having first been issued then an error will occur. A standalone Return instruction has no meaning because the system doesn't know where to Return to.

See Also
OnErr, GoSub, GoTo

Return Statement Example
The following example shows a simple function which uses a GoSub instruction to branch to a label called checkio and check the first 16 user inputs. Then the subroutine returns back to the main program.

```
Function main
    Integer var1, var2
    GoSub checkio
    On 1
    On 2
    Exit Function

checkio:   ' Subroutine starts here
    var1 = In(0)
    var2 = In(1)
    If var1 <> 0 Or var2 <> 0 Then
        Print "Message to Operator here"
    EndIf
    finished:       ' Subroutine ends here and returns to line 40
        Return
    Fend
```
Right$ Function

Returns a substring of the rightmost characters of a string.

Syntax

```plaintext
Right$(string, count)
```

Parameters

- `string`: String variable or character string of up to 255 characters from which the rightmost characters are copied.
- `count`: The number of characters to copy from `string` starting with the rightmost character.

Return Values

Returns a string of the rightmost `count` characters from the character string specified by the user.

Description

Right$ returns the rightmost `count` characters of a string specified by the user. Right$ can return up to as many characters as are in the character string.

See Also

Asc, Chr$, InStr, Left$, Len, Mid$, Space$, Str$, Val

Right$ Function Example

The example shown below shows a program which takes a part data string as its input and splits out the part number, part name, and part count.

```plaintext
Function SplitPartData(DataIn$ As String, ByRef PartNum$ As String, ByRef PartName$ As String, ByRef PartCount As Integer)
    PartNum$ = Left$(DataIn$, 10)
    DataIn$ = Right$(DataIn$, Len(DataIn$) - pos)
    pos = Instr(DataIn$, ",")
    PartName$ = Mid$(DataIn$, 11, 10)
    PartCount = Val(Right$(DataIn$, 5))
End
```

Some other example results from the Right$ instruction from the Command window.

```plaintext
> Print Right$("ABCDEFG", 2)
FG

> Print Right$("ABC", 3)
ABC
```
RmDir Statement

Removes an empty subdirectory from a controller disk drive.

**Syntax**

```
Rmdir dirName
```

**Parameters**

- `dirName`: String expression for the path and name of the directory to remove.
  - If the directory name is specified without a path, then the subdirectory in the current directory is specified.
  - See ChDisk for the details of path.

**Description**

Removes the specified subdirectory. Prior to executing Rmdir all of the subdirectory's files must be deleted.

- The current directory or parent directory cannot be removed.
- When executed from the Command window, quotes may be omitted.

**Note**

- This statement is executable only with the PC disk.

**Rmdir Statement Example**

Example from the command window:

```plaintext
> Rmdir \mydata
```
Rnd Function

Returns a random number.

Syntax

\[
\text{Rnd} (\text{maxValue})
\]

Parameters

\( \text{maxValue} \) Real expression that represents the maximum return value.

Return Values

Random real number from 0 to range.

Description

Use Rnd to generate random number values.

See Also

Int, Randomize

Rnd Function Example

Here's a Rnd example that generates a random number between 1 and 10.

Function main
    Real r
    Integer randNum

    Randomize
    randNum = Int(Rnd(9)) + 1
    Print "Random number is: ", randNum
Fend
Robot Statement

Selects the current robot.

Syntax

```
Robot number
```

Parameters

```
number
```
Number of the desired robot. The value ranges from 1 to the number of installed robots.

Description

Robot allows the user to select the default robot for subsequent motion instructions.

On a system with one robot, the Robot statement does not need to be used.

See Also

```
Accel, AccelS, Arm, ArmSet, Go, Hofs, Home, HOndr, Local, Move, Pulse, Robot Function, Speed, SpeedS
```

Robot Statement Example

```
Function main
  Integer I
  For I = 1 to 100
    Robot 1
    Go P(i)
    Robot 2
    Go P(i)
  Next I
Fend
```
# Robot Function

Returns the current robot number.

**Syntax**

```robot```

**Return Values**

Integer containing the current robot number.

**See Also**

Robot Statement

**Robot Function Example**

```print "The current robot is: ", robot```

RobotInfo Function

Returns status information for the robot.

Syntax

RobotInfo(index)

Parameters

index Integer expression that represents the index of the information to retrieve.

Return Values

The specified information is returned as an integer.

Description

The information for each bit of the returned value is shown in the table below:

<table>
<thead>
<tr>
<th>Index</th>
<th>Bit</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>&amp;H1</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>&amp;H2</td>
<td>Resettable error has occurred</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>&amp;H4</td>
<td>Non-resettable error has occurred</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>&amp;H8</td>
<td>Motors are on</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>&amp;H10</td>
<td>Current power is high</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>&amp;H20</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>&amp;H40</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>&amp;H80</td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>&amp;H100</td>
<td>Robot is halted</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>&amp;H200</td>
<td>Robot not halted (executing motion or in quick pause)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>&amp;H400</td>
<td>Robot stopped by pause or safeguard</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>&amp;H4000</td>
<td>TILL condition was satisfied by preceding motion command</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>&amp;H8000</td>
<td>SENSE condition was satisfied by preceding motion command</td>
</tr>
<tr>
<td>16-31</td>
<td></td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>&amp;H1</td>
<td>Robot is tracking (Conveyor tracking)</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>&amp;H2</td>
<td>Robot is waiting for recovery motion (WaitRecover status)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>&amp;H4</td>
<td>Robot is being recovered</td>
</tr>
<tr>
<td>3-31</td>
<td></td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>&amp;H1</td>
<td>Robot is at home position</td>
</tr>
<tr>
<td></td>
<td>1-31</td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>&amp;H1</td>
<td>Joint 1 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>&amp;H2</td>
<td>Joint 2 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>&amp;H4</td>
<td>Joint 3 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>&amp;H8</td>
<td>Joint 4 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>&amp;H10</td>
<td>Joint 5 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>&amp;H20</td>
<td>Joint 6 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>&amp;H40</td>
<td>Joint 7 servo is engaged</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>&amp;H80</td>
<td>S axis servo is engaged</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>&amp;H100</td>
<td>T axis servo is engaged</td>
</tr>
<tr>
<td>9-31</td>
<td></td>
<td></td>
<td>Undefined</td>
</tr>
<tr>
<td>4</td>
<td>N/A</td>
<td>0 - 32</td>
<td>Number of tasks executing robot commands</td>
</tr>
<tr>
<td></td>
<td></td>
<td>-1</td>
<td>0 = command executing from command window or macro</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1 = no task is using the manipulator</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>&amp;H1</td>
<td>Joint 1 brake is on</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>&amp;H2</td>
<td>Joint 2 brake is on</td>
</tr>
</tbody>
</table>
### RobotInfo Function

<table>
<thead>
<tr>
<th>Index</th>
<th>Bit</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td>&amp;H4</td>
<td>Joint 3 brake is on</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>&amp;H8</td>
<td>Joint 4 brake is on</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>&amp;H10</td>
<td>Joint 5 brake is on</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>&amp;H20</td>
<td>Joint 6 brake is on</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>&amp;H40</td>
<td>Joint 7 brake is on</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>&amp;H80</td>
<td>S axis brake is on</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>&amp;H100</td>
<td>T axis brake is on</td>
</tr>
<tr>
<td>9-31</td>
<td></td>
<td></td>
<td>Undefined</td>
</tr>
</tbody>
</table>

### See Also

CtrlInfo, RobotInfo$, TaskInfo

### RobotInfo Function Example

```plaintext
If (RobotInfo(3) And &H1) = &H1 Then
  Print "Joint 1 is locked"
Else
  Print "Joint 1 is free"
EndIf
```
RobotInfo$ Function

Returns text information for the robot.

Syntax

RobotInfo$(index)

Parameters

index Integer expression that represents the index of the information to retrieve.

Return Values

A string containing the specified information.

Description

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Robot name</td>
</tr>
<tr>
<td>1</td>
<td>Model name</td>
</tr>
<tr>
<td>2</td>
<td>Default point file name</td>
</tr>
<tr>
<td>3</td>
<td>Undefined</td>
</tr>
<tr>
<td>4</td>
<td>Serial number of robot</td>
</tr>
</tbody>
</table>

See Also

CtrlInfo, RobotInfo, TaskInfo

RobotInfo$ Function Example

Print "Robot Name: ", RobotInfo$(0)
RobotModel$ Function

Returns the robot model name.

Syntax
RobotModel$

Return Values
A string containing the model name. This is the name that is shown on the rear panel of the robot.

See Also
RobotType

RobotModel$ Function Example
Print "The robot model is ", RobotModel$
RobotName$ Function

Returns the robot name.

Syntax

RobotName$

Return Values

A string containing the robot name.

See Also

RobotInfo, RobotModel$

RobotName$ Function Example

Print "The robot name is ", RobotName$
RobotSerial$ Function

Returns the robot serial number.

**Syntax**

```
RobotSerial$
```

**Return Values**

A string containing the robot serial number.

**See Also**

RobotInfo, RobotName$, RobotModel$

**RobotSerial$ Function Example**

```
Print "The robot serial number is ", RobotSerial$
```
RobotType Function

Returns the robot type.

Syntax
RobotType

Return Values
1: Joint
2: Cartesian
3: SCARA
5: 6-AXIS
6: RS series
7: N series

See Also
RobotModel$

RobotType Function Example

If RobotType = 3 Then
    Print "Robot type is SCARA"
EndIf
ROpen Statement

Opens a file for reading.

Syntax

\[ \text{ROpen fileName As \#fileNumber} \]
\[ \cdot \]
\[ \text{Close \#fileNumber} \]

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>A string expression containing the file name to read from including the path. If only file name is specified, a file in the current directory is specified. See ChDisk for the details.</td>
</tr>
<tr>
<td>fileNumber</td>
<td>Integer expression from 30 to 63</td>
</tr>
</tbody>
</table>

Description

Opens the specified fileName for reading and identifies it by the specified fileNumber. This statement is used to open and read data from the specified file.

Notes

- PC disk only
- A network path is available.

The fileNumber identifies the file as long as the file is open and until it is closed the same file number cannot be used to the other files.

The fileNumber is used for the file operation commands (Input#, Read, Seek, Eof, Close)

Close statement closes the file and releases the file number.

It is recommended that you use the FreeFile function to obtain the file number so that more than one task are not using the same number.

See Also

Close, Input #, AOpen, BOpen, UOpen, WOpen, FreeFile

ROpen Statement Example

```plaintext
Integer fileNum, i, j

fileNum = FreeFile
WOpen "TEST.DAT" As \#fileNum
For i = 0 To 100
    Print \#fileNum, i
Next i
Close \#fileNum

fileNum = FreeFile
ROpen "TEST.DAT" As \#fileNum
For i = 0 to 100
    Input \#fileNum, j
    Print "data = ", j
Next i
Close \#fileNum
```
RSet$ Function

Returns the specified string with leading spaces added up to the specified length.

Syntax

RSet$ (string, length)

Parameters

string

String expression.

length

Integer expression for the total length of the string returned.

Return Values

Specified string with leading spaces appended.

See Also

LSet$, Space$

RSet$ Function Example

temp$ = "123"
temp$ = RSet$(temp$, 10)  ' temp$ = "       123"
RShift Function

Shifts numeric data to the right by a user specified number of bits.

Syntax

RShift(number, shiftBits)

Parameters

- **number**: Numeric expression to be shifted.
- **shiftBits**: The number of bits (integer from 0 to 31) to shift number to the right.

Return Values

Returns a numeric result which is equal to the value of number after shifting right shiftBits number of bits.

Description

RShift shifts the specified numeric data (number) to the right (toward a lower order digit) by the specified number of bits (shiftBits). The high order bits shifted are replaced by 0.

The simplest explanation for RShift is that it simply returns the result of number / 2^{shiftBits}. (Number is divided by 2 shiftBit times.)

Note

Numeric Data Type:

The numeric data (number) may be any valid numeric data type. RShift works with data types: Byte, Double, Int32, Integer, Long, Real, Short, UByte, UInt32, and UShort.

See Also

And, LShift, LShift64, Not, Or, RShift64, Xor

RShift Function Example

The example shown below shows a program which shows all the possible RShift values for an Integer data type starting with the integer set to “0”.

```plaintext
Function rshiftst
    Integer num, snum, i
    num = 32767
    For i = 1 to 16
        Print "i =", i
        snum = RShift(num, 1)
        Print "RShift(32767, ", i, ") = ", snum
    Next i
End
```

Some other example results from the RShift instruction from the command window.

```plaintext
> Print RShift(10,1)
5
> Print RShift(8,3)
1
> Print RShift(16,2)
4
```
**RShift64 Function**

Shifts numeric data to the right by a user specified number of bits.

**Syntax**

\[
\text{RShift64}(\text{number, shiftBits})
\]

**Parameters**

- `number`: Numeric expression to be shifted.
- `shiftBits`: The number of bits (integer from 0 to 63) to shift `number` to the right.

**Return Values**

Returns a numeric result which is equal to the value of `number` after shifting right `shiftBits` number of bits.

**Description**

RShift64 shifts the specified numeric data (`number`) to the right (toward a lower order digit) by the specified number of bits (`shiftBits`). The high order bits shifted are replaced by 0.

The simplest explanation for RShift64 is that it simply returns the result of \( \text{number} / 2^{\text{shiftBits}} \). (Number is divided by \( 2^{\text{shiftBits}} \) times.)

**Note**

**Numeric Data Type:**

The numeric data (`number`) may be any valid numeric data type. RShift64 works with Int64 and UInt64 data types.

**See Also**

And, LShift, LShift64, Not, Or, RShift, Xor

**RShift64 Function Example**

The example shown below shows a program which shows all the possible RShift64 values for an Integer data type starting with the integer set to “0”.

```plaintext
Function rshift64tst
    UInt64 num, snum, i
    num = 18446744073709551615
    For i = 1 to 63
        Print "i = ", i
        snum = RShift64(num, i)
        Print "RShift64(18446744073709551615, ", i, ") = ", snum
    Next i
Fend
```

Some other example results from the RShift64 instruction from the command window.

```plaintext
> Print RShift64(10,1)
  5
> Print RShift64(8,3)
  1
> Print RShift64(16,2)
  4
```
RTrim$ Function

Returns a string equal to specified string without trailing spaces.

Syntax

RTrim$(string)

Parameters

string String expression.

Return Values

Specified string with trailing spaces removed.

See Also

LTrim$, Trim$

RTrim$ Function Example

```plaintext
str$ = " data 
str$ = RTrim$(str$)  ' str$ = "..data"
```
RunDialog Statement

Runs an EPSON RC+ dialog from a SPEL+ program.

Syntax

(1) **RunDialog dialogID**
(2) **RunDialog DLG_ROBOTMNG [, robotAllowed]**

Parameters

**dialogID** Integer expression containing a valid dialog ID. These values are predefined constants as shown below.

<table>
<thead>
<tr>
<th>dialogID</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLG_ROBOTMNG</td>
<td>100</td>
<td>Run the Robot Manager dialog</td>
</tr>
<tr>
<td>DLG_IOMON</td>
<td>102</td>
<td>Run I/O Monitor</td>
</tr>
<tr>
<td>DLG_VGUIDE</td>
<td>110</td>
<td>Run Vision Guide dialog</td>
</tr>
</tbody>
</table>

**robotAllowed** This parameter is only available when DLG_ROBOTMNG is specified as **dialog ID**. Specifies a robot that is available in the Robot Manager in bit value.

<table>
<thead>
<tr>
<th>Example</th>
<th>Set value</th>
<th>bit15</th>
<th>bit14</th>
<th>...</th>
<th>bit2</th>
<th>bit1</th>
<th>bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robot 1</td>
<td>&amp;H0001</td>
<td>Off</td>
<td>Off</td>
<td></td>
<td>Off</td>
<td>Off</td>
<td>On</td>
</tr>
<tr>
<td>Robot 2</td>
<td>&amp;H0002</td>
<td>Off</td>
<td>Off</td>
<td></td>
<td>Off</td>
<td>On</td>
<td>Off</td>
</tr>
<tr>
<td>Robot 1 and 2</td>
<td>&amp;H0003</td>
<td>Off</td>
<td>Off</td>
<td></td>
<td>Off</td>
<td>On</td>
<td>On</td>
</tr>
<tr>
<td>Robot 16</td>
<td>&amp;H1000</td>
<td>On</td>
<td>Off</td>
<td></td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
</tr>
</tbody>
</table>

Description

Use RunDialog to run EPSON RC+ dialogs from a SPEL+ task. The task will be suspended until the operator closes the dialog.

When running dialogs that execute robot commands, you should ensure that no other tasks will be controlling the robot while the dialog is displayed, otherwise errors could occur.

See Also

InputBox, MsgBox

RunDialog Statement Example

```plaintext
If Motor = Off Then
   RunDialog DLG_ROBOTMNG
   If Motor = Off Then
      Print "Motors are off, aborting program"
      Quit All
   EndIf
EndIf
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9 615
SafetyOn Function

Return the Safety Door open status.

Syntax

SafetyOn

Return Values

True if the Safety Door is Open, otherwise False.

Description

SafetyOn function is used only for NoPause task, NoEmgAbort task (special task using NoPause or NoEmgAbort at Xqt), and background tasks.

See Also

ErrorOn, EstopOn, PauseOn, Wait, Xqt

SafetyOn Function Example

The following example shows a program that monitors the Safety Door open and switches the I/O On/Off when Safety Door open occurs.

Note

Forced Flag

This program example uses Forced flag for On/Off command. Be sure that the I/O outputs change during error, or at Emergency Stop or Safety Door Open when designing the system.

Function main

Xqt SafetyOnOffMonitor, NoPause
:
Fend

Function SafetyOnOffMonitor

Do
    Wait SafetyOn = On
    Print "Saftey Open"
    Off 10, Forced
    On 12, Forced

    Wait SafetyOn = Off
    Print "Saftey Close"
    On 10, Forced
    Off 12, Forced

Loop
Fend
SavePoints Statement

Saves point data in main memory to a disk file for the current robot.

Syntax

SavePoints filename

Parameters

filename

String expression containing the file into which points will be stored. The extension must be ".pts". You cannot specify a file path and filename doesn’t have any effect from ChDisk. See ChDisk for the details.

Description

SavePoints saves points for the current robot to the specified file in the current project directory. A “.pts” extension must always be specified. The SavePoints command will also add the point file to the project for the current robot if it did not already exist.

The point data is stored in the Compact Flash inside of the controller. Therefore, SavePoints starts writing into the Compact Flash. Frequent writing into the Compact Flash will shorten the Compact Flash lifetime. We recommend using SavePoints only for saving the point data.

Potential Errors

Out of Disk Space

If there is no space remaining an error will occur.

Point file for another robot.

If filename is a point file for another robot, an error will occur.

A Path Cannot be Specified

If filename contains a path, an error will occur. Only a file name in the current project can be specified.

Bad File name

If a file name is entered which has spaces in the name, or other bad file name characteristics an error will occur.

See Also

ImportPoints, LoadPoints

SavePoints Statement Example

ClearPoints
For i = 1 To 10
  P(i) = XY(i, 100, 0, 0)
Next i
SavePoints "TEST.PTS"
Seek Statement

Changes position of file pointer for a specified file.

Syntax

Seek #fileNumber, pointer

Parameters

fileNumber    Integer expression from 30 to 63
pointer       Integer expression for the desired position to seek, starting from 0 to the length of the file.

See Also

BOpen, Read, ROpen, UOpen, Write, WOpen

Seek Statement Example

    Integer fileNum
    String data$

    fileNumber = FreeFile
    UOpen "TEST.DAT" As #fileNum
    Seek #fileNum, 20
    Read #fileNum, data$, 2
    Close #fileNum
Select...Send Statement

Executes one of several groups of statements, depending on the value of an expression.

Syntax

```
Select selectExpr
    Case caseExpr
        statements
    [Case caseExpr
        statements ]
    [Default
        statements ]
Send
```

Parameters

- `selectExpr`: Any numeric or string expression.
- `caseExpr`: Any numeric or string expression that evaluates to the same type as `selectExpr`.
- `statements`: One or more valid SPEL+ statements or multi-statements.

Description

If any one `caseExpr` is equivalent to `selectExpr`, then the statements after the Case statement are executed. After execution, program control transfers to the statement following the Send statement.

If no `caseExpr` is equivalent to `selectExpr`, the Default statements are executed and program control transfers to the statement following the Send statement.

If no `caseExpr` is equivalent to `selectExpr` and Default is omitted, nothing is executed and program control transfers to the statement immediately following the Send statement.

`selectExpr` may include constants, variables, and logical operators that use And, Or and Xor. `caseExpr` also may include constants, variables, and logical operators that use And, Or and Xor. In this case, the calculation result of `caseExpr` is compared to that of `selectExpr` and do not specify the variable in `caseExpr` because the motion becomes complicated.

See Also

If...Then...Else

Select...Send Statement Example

Shown below is a simple example for Select...Send:

```
Function Main
    Integer I
    For i = 0 To 10
        Select I
            Case 0
                Off 1;On 2;Jump P1
            Case 3
                On 1;Off 2
                Jump P2;Move P3;On 3
            Case 7
                On 4
        Default
            On 7
        Send
    Next
Send
```

SelectDB Statement

Searches the data in the table in an opened database.

Syntax

```
SelectDB (#fileNumber, TableName, SelectCondition, SortMethod)
```

Parameters

- **#fileNumber**: Integer value from 501 to 508 representing the database number specified with the OpenDB statement.
- **TableName**: Table name you want to search in.
  - If the database type specified with #fileNumber is an Excel workbook, specify an Excel worksheet or named table.
  - When specifying an Excel sheet, add $ to end of the worksheet name and enclose the name with ".
  - When specifying an area with a name in an Excel worksheet, enclose the name with ".
- **SelectCondition**: Conditions of the search.
  - AND, OR are available to specify the multiple conditions.
  - If omitted, the all data in the table is searched.
- **SortMethod**: Order to show searched data.
  - Specify Sort key and Sort order (ascending order [ASC] / descending order [DESC]).
  - If the Sort order is omitted, the ascending Sort key order is specified.
  - If the SortMethod is omitted, the order is decided by the opened database.

Return Values

Returns total numbers of lines.

Description

Sorts the data which meets the SelectCondition in the specified table of the opened database based on the Sort conditions.

You should execute SelectDB before reading / writing data with the Input# and Print# statements.

If the opened database is an Excel workbook, write a row name to use for the search in the first line of the worksheet and area defined with the name.

For Excel 2007 workbook, the worksheet name must be specified. You cannot access to area defined with the name.

Note

- Connection of PC with installing RC+ is required.

See Also

OpenDB, CloseDB, UpdateDB, DeleteDB, Input #, Print #
SelectDB Example

The following example uses the SQL server 2000 sample database, Northwind. The Employees table is searched with the condition TitleOfCourtesy = Ms. with EmployeeID in descending order.

```plaintext
Integer count, i, eid
String Lastname$, Firstname$, Title$

OpenDB #501, SQL, "(LOCAL)", "Northwind"
count = SelectDB(#501, "Employees", "TitleOfCourtesy = 'Ms.'", "EmployeeID DESC")
For i = 0 To count - 1
   Input #501, eid, Lastname$, Firstname$, Title$
   Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
Next
CloseDB #501
```

Using Access database

The following example uses Microsoft Access 2007 sample database “Students” and loads the data whose ID is more than 10 from the table “Students” in the ID descending order.

```plaintext
Integer count, i, eid
String Lastname$, Firstname$, dummy$

OpenDB #502, Access, "c:\MyDataBase\Students.accdb"
count = SelectDB(#502, "Students", "ID > 10'", "ID")
For i = 0 To count - 1
   Input #502, eid, dummy$, Lastname$, Firstname$
   Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #502
```

Using Excel workbook

The following example uses Microsoft Excel workbook “Students” and loads the data in worksheet “Student” whose Age is under 25 with the ID in ascending order.

```plaintext
Integer count, i, eid
String Lastname$, Firstname$

OpenDB #503, Excel, "c:\MyDataBase\Students.xls"
count = SelectDB(#503, "[Students$]", "Age < 25", "ID ASC")
For i = 0 To count - 1
   Input #503, eid, Lastname$, Firstname$
   Print eid, ",", Lastname$, ",", Firstname$
Next
CloseDB #503
```
Sense Statement

Specifies and displays input condition that, if satisfied, completes the Jump in progress by stopping the robot above the target position.

Syntax

\[
\text{Sense [ condition ]}
\]

Parameters

condition

Input status specified as a trigger

\[(Event) \text{ comparative operator} \ (\ =, \ <>, \ >=, \ >, \ <, \ <=) \ [Integer \ expression]\]

The following functions and variables can be used in the Event:

Functions:
- Sw, In, InW, Oport, Out, OutW, MemSw, MemIn, MemInW, Ctr
- GetRobotInsideBox, GetRobotInsidePlane, AIO_In, AIO_InW, AIO_Out, AIO_OutW

Variables:
- Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort
- global, preserve variable, Global variable, module variable

In addition, using the following operators you can specify multiple event conditions.

Operator: And, Or, Xor

Example:
- Sense Sw(5) = On
- Sense Sw(5) = On And Sw(6) = Off

Description

Sense is used to stop approach motion during a Jump, Jump3, and Jump3CP instructions. The Sense condition must include at least one of the functions above.

When variables are included in the Sense condition, their values are computed when setting the Sense condition. No use of variable is recommended. Otherwise, the condition may be an unintended condition. Multiple Sense statements are permitted. The most recent Sense condition remains current until superseded with another Sense statement.

Jump, Jump3, Jump3CP with Sense Modifier

Checks if the current Sense condition is satisfied. If satisfied, the Jump instruction completes with the robot stopped above the target position. (i.e. When the Sense Condition is True, the robot arm remains just above the target position without executing approach motion. When the Sense condition is False, the robot arm completes the full Jump instruction motion through to the target position.

When parameters are omitted, the current Sense definition is displayed.
Notes

Sense Setting at Main Power On
At power on, the initial Sense condition is:
Sense Sw(0) = On     'Robot does not execute downward motion when Input bit 0 is on.

Use of JS and Stat to Verify Sense
Use JS or Stat to verify if the Sense condition has been satisfied after executing a motion command using
Sense modifier.

To use a variables in the event condition expression
- Available variables are Integer type (Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)
- Array variables are not available
- Local variables are not available
- If a variable value cannot satisfy the event condition for more than 0.01 seconds, the system cannot retrieve
  the change in variables.
- Up to 64 can wait for variables in one system (including the ones used in the event condition expressions
  such as Wait). If it is over 64, an error occurs during the project build.
- If you try to transfer a variable waiting for variables as a reference with Byref, an error occurs.
- When a variable is included in the right side member of the event condition expression, the value is
  calculated when the motion command start. We recommend not using variables in an integer expression
  to avoid making unintended conditions.

See Also
In, JS, Jump, Jump3, Jump3CP, MemIn, MemSw, Stat, Sw

Sense Example
This is a simple example on the usage of the Sense instruction.

Function test
.
.
TrySense:
  Sense Sw(1) = Off     'Specifies the arm stops above the target when
  'the input bit 1 is Off.
  Jump P1 C2 Sense
  If JS = True Then
    GoSub ERRPRC       'If the arm remains stationary
    GoTo TrySense      'above the point specified,
    GoTo ErrPRC       'then execute ERRPRC and go to TrySense.
  EndIf
  On 1; Wait 0.2; Off 1
.
.
Fend

<Other Syntax Examples>

> Sense Sw(1)=1 And MemSw(1)=1
> Sense Sw(0) Or (Sw(1) And MemSw(1))
SetCom Statement

Sets or displays parameters for RS-232C port.

Syntax

```
SetCom #portNumber [ , baud ] [ , dataBits ] [ , stopBits ] [ , parity ] [ , terminator ] [ , HWFlow ]
[ , SWFlow ] [ , timeOut ]
```

Parameters

- **portNumber**: Integer value representing a RS-232C port number
  - Real Part: 1 to 8
  - Windows Part: 1001 to 1008

- **baud**: Optional. Specifies the baud rate. Valid values are:
  - 110
  - 2400
  - 300
  - 4800
  - 600
  - 9600
  - 1200
  - 14400
  - 19200
  - 38400
  - 56000
  - 115200
  - (Default: 9600)

  When using the Windows Part port, some data may drop in the baud rate of 19200 or more.

- **dataBits**: Optional. Specifies the number of data bits per character. Valid values are 7 and 8.

- **stopBits**: Optional. Specifies the number of stop bits per character. Valid values are 1 and 2.

- **parity**: Optional. Specifies the parity. Valid values are O (Odd), E (Even), and N (None).

- **terminator**: Optional. Specifies the line termination characters. Valid values are CR, LF, CRLF.

- **HWFlow**: Optional. Specifies hardware control. Valid values are RTS and NONE.

- **SWFlow**: Optional. Specifies software control. Valid values are XON and NONE.

- **timeOut**: Optional. Specifies the maximum time for transmit or receive in seconds. If this value is 0, then there is no time out.

Description

When all the parameter is omitted, displays a communication port setting.

If the several ports are used in the communication at one time with more than 19200 baud rate, error 2929 or 2922 may occur. In this case, select the lower baud rate or avoid using several ports at one time.

When using the Windows Part port, some data may drop in the baud rate of 19200 or more.

If any data drops, select the lower baud rate or use the Real Part port.

Parameters are stored to the Compact Flash inside the Controller. When you execute SetCom, the data is written to the Compact Flash. If a data is written to the Compact Flash frequently, it may shorten the Compact Flash life. Using SetCom only when changing the parameter is recommended.

See Also

OpenCom, CloseCom, SetNet

SetCom Statement Example

```
SetCom #1, 9600, 8, 1, N, CRLF, NONE, NONE, 0
SetCom #2, 4800
```
SetLatch Statement

Sets the latch function of the robot position using the R-I/O input.

Syntax

SetLatch { #portNumber, triggerMode }

Parameters

#portNumber

Port number of the R-I/O input port to connect the trigger input signal. The table below shows the port numbers you can specify. Specify the port number of the unit that the object robot is connected.

<table>
<thead>
<tr>
<th>Control Unit</th>
<th>Port Number</th>
<th>Point</th>
<th>Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>2 points</td>
<td>24, 25</td>
<td></td>
</tr>
<tr>
<td>OUTPUT</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Drive Unit 1</th>
<th>Port Number</th>
<th>Point</th>
<th>Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>2 points</td>
<td>56, 57</td>
<td></td>
</tr>
<tr>
<td>OUTPUT</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Drive Unit 2</th>
<th>Port Number</th>
<th>Point</th>
<th>Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>2 points</td>
<td>280, 281</td>
<td></td>
</tr>
<tr>
<td>OUTPUT</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

The following constants are defines as the port number.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Port Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETLATCH_PORT_CU_0</td>
<td>24</td>
</tr>
<tr>
<td>SETLATCH_PORT_CU_1</td>
<td>25</td>
</tr>
<tr>
<td>SETLATCH_PORT_DU1_0</td>
<td>56</td>
</tr>
<tr>
<td>SETLATCH_PORT_DU1_1</td>
<td>57</td>
</tr>
<tr>
<td>SETLATCH_PORT_DU2_0</td>
<td>280</td>
</tr>
<tr>
<td>SETLATCH_PORT_DU2_1</td>
<td>281</td>
</tr>
</tbody>
</table>

triggerMode

The trigger input signal logic to connect with the R-I/O. The logic can be specified with the following constants.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETLATCH_TRIGGERMODE_TRAILINGEDGE</td>
<td>0</td>
<td>Negative logic</td>
</tr>
<tr>
<td>SETLATCH_TRIGGERMODE_LEADINGEDGE</td>
<td>1</td>
<td>Positive logic</td>
</tr>
</tbody>
</table>

With the negative logic, it latches the robot position at the switch edge from the input signal High to Low.

With the positive logic, it latches the robot position at the switch edge from the input signal from Low to High.

Description

Sets the condition of the robot position latch using the R-I/O input signals. One robot cannot wait the trigger signals of several ports simultaneously.
Executing SetLatch needs approx. 40 msec for processing.
SetLatch Statement

Note
If you specify a port number of the unit unrelated to the selected robot, the error “I/O input/output bit number is out of available range” occurs.

See Also
LatchEnable, LatchState Function, LatchPos Function

SetLatch Statement Example

```
Function main
  SetLatch 24, SETLATCH_TRIGGERMODE_LEADINGEDGE 'Positive logic
  LatchEnable On ' Enable the latch function
  Go P1
  Wait LatchState = True ' Wait a trigger
  Print LatchPos ' Display the latched position
  LatchEnable Off ' Disable the latch function
End
```
SetIn Statement

For Virtual IO, sets specified input port (8 bits) to the specified value.

Syntax

SetIn portNumber, value

Parameters

portNumber  Integer expression representing the input port number.
value  Integer expression between 0 and 255 to set the specified port to.

Description

SetIn provides the ability to set up to 8 bits of virtual inputs at once.

See Also

SetSW, SetInW

SetIn Statement Example

  > setin 0, 1  ' Sets the first bit of port 0 to On.
SetInW Statement

For Virtual IO, sets specified input word (16 bits) to the specified value.

Syntax

```
SetInW portNumber, value
```

Parameters

- **portNumber**
  - Integer expression representing the input port number.
- **value**
  - Number between 0 and 65535 to set the specified word to.

Note

**Rule of word port which contains the input bit of Real Time I/O**

The input bit of the Real Time I/O cannot be reflected.
Specify the setting value of the word ports which contain the input bit of Real Time I/O =1, 3, 17, 19 by an integer from 0 to 255.
The value larger than 255 will result in an error.

Description

SetInW provides the ability to set up to 16 bits of virtual inputs at once.

See Also

SetSw, SetIn

SetInW Statement Example

```
> setInw 0, 1  ' Sets the first bit of word 0 to On.
```
SetNet Statement

Sets parameters for a TCP/IP port.

Syntax

```
SetNet #portNumber, hostAddress, TCP_IP_PortNum, terminator, SWFlow, timeout, protocol
```

Parameters

- **portNumber**: Specifies which TCP/IP port to set parameters for. Valid values are 201 to 216.
- **hostAddress**: Specifies the host IP address.
- **TCP_IP_PortNum**: Specifies the TCP/IP port number for this node.
- **terminator**: Specifies the line termination characters. Valid values are CR, LF, CRLF.
- **SWFlow**: Specifies software control. Valid value is NONE.
- **timeOut**: Specifies the maximum time for transmit or receive in seconds. If this value is “0”, then there is no time out.
- **protocol**: Specifies the protocol (TCP/UDP) of communication.

Description

Parameters are stored to the Compact Flash inside the Controller. When you execute SetNet, the data is written to the Compact Flash. If a data is written to the Compact Flash frequently, it may shorten the Compact Flash life. Using SetNet only when changing the parameter is recommended.

See Also

OpenNet, CloseNet, SetCom

SetNet Statement Example

```
SetNet #201, "192.168.0.1", 2001, CRLF, NONE, 0
```
SetSw Statement

For Virtual IO, sets specified input bit to the specified value.

Syntax

SetSw bitNumber, value

Parameters

bitNumber  Integer expression representing the input bit number.
value      Integer expression with a value of 0 (Off) or 1 (On).

Description

SetSw provides the ability to turn on or off one input bit.

See Also

SetIn, SetInW

SetSw Statement Example

> setsw 2, on  ' Sets the 2nd input bit to On.
SFree Statement

Removes servo power from the specified servo axis.

Syntax

```
SFree jointNumber [ , jointNumber, ... ]
```

Parameters

- `jointNumber`: An integer expression representing a servo joint number (1 to 9). The additional S axis is 8 and T axis is 9.

Description

SFree removes servo power from the specified servo joints. This instruction is used for the direct teaching or the part installation by partially de-energizing a specific joint. To re-engage a joint execute the SLock instruction or Motor On.

SFree initializes the robot control parameter. See Motor On for the details.

Note

**SFree Sets Some System Items back to Their Initial State:**

SFree, for safety purposes, initializes parameters concerning the robot arm speed (Speed and SpeedS), acceleration (Accel and AccelS) and the LimZ parameter.

**Important**

**SFree and its Use with the Z Joint and U Joint for SCARA robots (including RS series)**

The Z joint has electromagnetic brakes so setting SFree for the Z joint does not immediately allow the Z joint to be moved. To move the Z joint by hand requires the brake to be released continuously by pressing the brake release switch on the top of the robot arm.

Some model has electronic brake on the U joint. When the robot has the U joint electronic brake, setting SFree for the U joint does not immediately allow the U joint to be moved. To move the U joint by hand requires the brake to be released continuously by pressing the brake release switch on the top of the robot arm.

**SFree is Not Valid with 6-Axis robots (including N series)**

When SFree is executed in 6-axis robots (including N series), an error occurs. To move the arm by hands, release the electromagnetic brake by using Brake Off after tuning OFF the motor by Motor Off.

**Executing motion commands while joints are in SFree state**

Attempting to execute a motion command while in the SFree condition will cause an error in the Controller’s default state. However, to allow motion while 1 or more of the joints are in the SFree state, select the [Allow motion with one or more joints free] checkbox from [Setup]-[System Configuration]-[Controller]-[Preferences].

**Do not use SFree during Conveyor Tracking**

Error 5057 or 5058 might occur if SFree is used during conveyor tracking. Use SFree after terminating conveyor tracking such as Cnv_AbortTrack.

**See Also**

- Brake, LimZ, Motor, SFree Function, SLock
SFree Statement Example

This is a simple example on the usage of the SFree command. To operate the robot in this example, the [Allow motion with one or more joints free] checkbox must be selected from [Setup]-[System Configuration]-[Controller]-[Preferences].

```plaintext
Function GoPick
  Speed pickSpeed
  SFree 1, 2  ' Release the excitation of J1 and J2,
               ' and control the Z and U joints for part installation.
  Go pick
  SLock 1, 2  ' Restore the excitation of J1 and J2.
Fend
```
SFree Function

Returns SFree status for a specified joint.

Syntax

\[ \text{SFree(jointNumber)} \]

Parameters

\[ \text{jointNumber} \]

Integer expression representing the joint number to check.
The additional S axis is 8 and T axis is 9.

Return Values

True if the joint is free, False if not.

See Also

SFree Statement

SetFree Statement Example

\[
\text{If SFree(1) Then}
\quad \text{Print "Joint 1 is free"}
\quad \text{EndIf}
\]
Sgn Function

Determines the sign of the operand.

Syntax

\[ \text{Sgn}(\text{Operand}) \]

Parameters

\( \text{Operand} \) A numeric expression.

Return Values

1: If the operand is a positive value.
0: If the operand is a 0
-1: If the operand is a negative value.

Description

The Sgn function determines the sign of the numeric value of the operand.

See Also

Abs, And, Atan, Atan2, Cos, Int, Mod, Or, Not, Sin, Sqr, Str$, Tan, Val, Xor

Sgn Function Example

This is a simple command window example on the usage of the Sgn function.

```plaintext
>print sgn(123)
1
>print sgn(-123)
-1
>
```
Short Statement

Declares variables of Short type. (2 byte integer variable).

Syntax

```
Short varName [(subscripts)] [, varName [(subscripts)]] ...
```

Parameters

- **varName**: Variable name which the user wants to declare.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows:

  ```
  (ubound1, [ubound2], [ubound3])
  ```

  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

  - Local variable: 2,000
  - Global Preserve variable: 4,000
  - Global variable and module variable: 100,000

Description

Short is used to declare variables as type integer. Integer variables can contain values from -32768 to 32767. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

- Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Real, String, UByte, UInt32, UInt64, UShort

Short Statement Example

The following example shows a simple program that declares some variables using Short.

```
Function shorttest
    Short A(10)  ' Single dimension array of Short
    Short B(10, 10)  ' Two dimension array of Short
    Short C(5, 5, 5)  ' Three dimension array of Short
    Short var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
    Fend
```
ShutDown Statement

Shuts down EPSON RC+ and optionally shuts down or restarts Windows.

Syntax
ShutDown [mode] [, Forced]

Parameters

<table>
<thead>
<tr>
<th>mode</th>
<th>Optional. An integer expression that represents the mode setting described below.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbolic constant</td>
<td>Value</td>
</tr>
<tr>
<td>Mode omitted</td>
<td>-1</td>
</tr>
<tr>
<td>SHUTDOWN_ALL</td>
<td>0</td>
</tr>
<tr>
<td>SHUTDOWN_RESTART</td>
<td>1</td>
</tr>
<tr>
<td>SHUTDOWN_EPSONRC</td>
<td>2</td>
</tr>
</tbody>
</table>

Forced
Optional. Use to force a shutdown.

Description
Use ShutDown to shut down RC+ and optionally shutdown or reboot Windows from your program. You can force a shutdown by using the Forced parameter.

Note
If you shutdown with the Forced parameter while tasks are running, you could lose data. Be sure to save data before shutdown.

See Also
Restart

ShutDown Statement Example

ShutDown 0 ' Shutdown EPSON RC+ and Windows
ShutDown Function

Shuts down EPSON RC+ and optionally shuts down or restarts Windows.

Syntax

```plaintext
ShutDown ([mode], [Forced])
```

Parameters

- **mode**: Optional. An integer expression that represents the mode setting described below.

<table>
<thead>
<tr>
<th>Symbolic constant</th>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode omitted</td>
<td>-1</td>
<td>Displays a dialog allowing the user to choose the shutdown option.</td>
</tr>
<tr>
<td>SHUTDOWN_ALL</td>
<td>0</td>
<td>Shuts down EPSON RC+ and Windows.</td>
</tr>
<tr>
<td>SHUTDOWN_RESTART</td>
<td>1</td>
<td>Shuts down EPSON RC+ and restarts Windows.</td>
</tr>
<tr>
<td>SHUTDOWN_EPSONRC</td>
<td>2</td>
<td>Shuts down EPSON RC+.</td>
</tr>
</tbody>
</table>

- **Forced**: Optional. Use to force a shutdown.

Description

Use ShutDown to shut down RC+ and optionally shutdown or reboot Windows from your program. You can force a shutdown by using the Forced parameter.

Note

If you shutdown with the Forced parameter while tasks are running, you could lose data. Be sure to save data before shutdown.

Return Values

Returns the following integer values.

-1 When a dialog is displayed and the user selects Cancel.
0 If shutdown fails
1 If shutdown is successful

ShutDown Function Example

```plaintext
If ShutDown(SHUTDOWN_EPSONRC) = 1 Then
    Print "Shutdown: OK"
Else
    Print "Shutdown: NG"
EndIf
```
Signal Statement

Send a signal to tasks executing WaitSig.

Syntax

```
Signal signalNumber
```

Parameters

```
signalNumber  Signal number to transmit.  Range is 0 to 63.
```

Description

Signal can be used to synchronize multi-task execution. Previous signals issued before WaitSig is executed are ignored.

See Also

WaitSig

Signal Statement Example

```
Function Main
  Xqt 2, SubTask
  Call InitSys
  Signal 1

Fend

Function SubTask
  WaitSig 1

Fend
```
SimGet Statement

Acquire the setting values of each object properties of simulator.

Syntax

SimGet Object.Property, Var
SimGet Robot.Hand.Property, Var

Parameters

Object String variable that indicates object names acquiring the property values.
Robot String variable that indicates the robot name which the hand specified by “Hand” is installed.
Hand String variable that indicates the hand name which acquires the property values.
Property Property name that acquires values. Descriptions of properties are described later.
Var Variable that indicates return value.

Description

Use this command to acquire the property setting value of each object of simulator.

Set the the following properties to acquire the object setting values.

<table>
<thead>
<tr>
<th>Property</th>
<th>Descriptions</th>
<th>Unit</th>
<th>Data type</th>
<th>Return value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PositionX</td>
<td>Acquire a position of X coordinate system.</td>
<td>(mm)</td>
<td>Double</td>
<td></td>
</tr>
<tr>
<td>PositionY</td>
<td>Acquire a position of Y coordinate system.</td>
<td>(mm)</td>
<td>Double</td>
<td></td>
</tr>
<tr>
<td>PositionZ</td>
<td>Acquire a position of Z coordinate system.</td>
<td>(mm)</td>
<td>Double</td>
<td></td>
</tr>
<tr>
<td>RotationX</td>
<td>Acquire rotation angle of X axis.</td>
<td>(degree)</td>
<td>Double</td>
<td></td>
</tr>
<tr>
<td>RotationY</td>
<td>Acquire rotation angle of Y axis.</td>
<td>(degree)</td>
<td>Double</td>
<td></td>
</tr>
<tr>
<td>RotationZ</td>
<td>Acquire rotation angle of Z axis.</td>
<td>(degree)</td>
<td>Double</td>
<td></td>
</tr>
<tr>
<td>CollisionCheck</td>
<td>Acquire enable/disable of collision detect.</td>
<td>-</td>
<td>Boolean</td>
<td>True or False</td>
</tr>
<tr>
<td>CollisionCheckSelf</td>
<td>Acquire enable/disable of self-collision detect of the robot.</td>
<td>-</td>
<td>Boolean</td>
<td>True or False</td>
</tr>
<tr>
<td>Visible</td>
<td>Acquire state of display/non-display.</td>
<td>-</td>
<td>Boolean</td>
<td>True or False</td>
</tr>
<tr>
<td>Type</td>
<td>Acquire the types of objects.</td>
<td>-</td>
<td>Integer</td>
<td>Layout: 0 Part: 1 Mounted Device: 3</td>
</tr>
</tbody>
</table>

You can acquire the properties by combinations shown in the list below.

<table>
<thead>
<tr>
<th>Property</th>
<th>Robot</th>
<th>Hand</th>
<th>Box</th>
<th>Sphere</th>
<th>Cylinder</th>
<th>Plane</th>
<th>CAD</th>
<th>Camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>PositionX</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PositionY</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PositionZ</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RotationX</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RotationY</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RotationZ</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CollisionCheck</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CollisionCheckSelf</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Visible</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Type</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
</tbody>
</table>
SimGet Statement

See Also
SimSet

SimGet Statement Example

'Acquire X coordinate value of SBox_1 object
Double boxPosX
SimGet SBox_1.PositionX, boxPosX

'Acquire the state of display/non-display of SBox_1 object
Boolean boxVisible
SimGet SBox_1.Visible, boxVisible

'Acquire the type of SBox_1 object
Integer boxType
SimGet SBox_1.Type, boxType
SimSet Statement

Set properties of each object of simulator. Operate the robot motion, objects, and simulator settings.

Syntax
(1) Property setting for object
   \texttt{SimSet Object.Property, Value} \\
   \texttt{SimSet Robot.Hand.Property, Value}

(2) Motion settings for robot (Pick & Place)
   \texttt{SimSet Robot.Pick, Object [, Tool]} \\
   \texttt{SimSet Robot.Place, Object}

(3) Operation settings for objects (specify the parent object)
   \texttt{SimSet Object.SetParent [, ParentObject]}

(4) Simulator settings (reset the collision detect)
   \texttt{SimSet ResetCollision}

Parameters
(1) Property setting for object
   \begin{itemize}
     \item \textbf{Object} String variable that indicates object names setting the property values.
     \item \textbf{Robot} String variable that indicates the robot name which the hand specified by “Hand” is installed.
     \item \textbf{Hand} String variable that indicates hand name which sets the property values.
     \item \textbf{Property} Property name that sets values. Descriptions of properties are described later.
     \item \textbf{Value} Formula with new values. Data type depends on properties.
   \end{itemize}

(2) Motion settings for robot (Pick & Place)
   \begin{itemize}
     \item \textbf{Robot} String variable that indicates the robot name to Pick or Place.
     \item \textbf{Object} String variable that indicates the object name to be Picked or Placed.
     \item \textbf{Tool} Formula that indicates Tool number which is used at the time of Picking.
   \end{itemize}

(3) Operation settings for object (specify the parent object)
   \begin{itemize}
     \item \textbf{Object} String variable that indicates the object name which sets the parent object.
     \item \textbf{ParentObject} String variable that indicates the parent object name.
   \end{itemize}
Description
Use this command to set properties of each object of simulator. Also, use the command to change the robot motion, objects, and simulator settings.

(1) Property setting for object
You can set the objects by specifying the properties shown below.

<table>
<thead>
<tr>
<th>Property</th>
<th>Descriptions</th>
<th>Unit</th>
<th>Data type</th>
<th>Return value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PositionX</td>
<td>Set a position of X coordinate.</td>
<td>(mm)</td>
<td>Double</td>
<td>Max: 100000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Min: -100000</td>
</tr>
<tr>
<td>PositionY</td>
<td>Set a position of Y coordinate.</td>
<td>(mm)</td>
<td>Double</td>
<td>Max: 100000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Min: -100000</td>
</tr>
<tr>
<td>PositionZ</td>
<td>Set a position of Z coordinate.</td>
<td>(mm)</td>
<td>Double</td>
<td>Max: 100000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Min: -100000</td>
</tr>
<tr>
<td>RotationX</td>
<td>Set rotation angle of X axis.</td>
<td>(degree)</td>
<td>Double</td>
<td>Max: 360</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Min: -360</td>
</tr>
<tr>
<td>RotationY</td>
<td>Set rotation angle of Y axis.</td>
<td>(degree)</td>
<td>Double</td>
<td>Max: 360</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Min: -360</td>
</tr>
<tr>
<td>RotationZ</td>
<td>Set rotation angle of Z axis.</td>
<td>(degree)</td>
<td>Double</td>
<td>Max: 360</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Min: -360</td>
</tr>
<tr>
<td>CollisionCheck</td>
<td>Set enable/disable of collision detect.</td>
<td>-</td>
<td>Boolean</td>
<td>True or False</td>
</tr>
<tr>
<td>CollisionCheckSelf</td>
<td>Set enable/disable of self-collision detect of the robot.</td>
<td>-</td>
<td>Boolean</td>
<td>True or False</td>
</tr>
<tr>
<td>Visible</td>
<td>Set state of display/non-display.</td>
<td>-</td>
<td>Boolean</td>
<td>True or False</td>
</tr>
</tbody>
</table>

You can set the property by combinations shown in the list below.

| Property     | Robot | Hand | Box | Sphere | Cylinder | Plane | CAD | Camera |
|--------------|-------|------|-----|--------|----------|-------|     |        |
| PositionX    | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| PositionY    | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| PositionZ    | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| RotationX    | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| RotationY    | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| RotationZ    | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| CollisionCheck | ✓     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |
| CollisionCheckSelf | ✓     | -    | -   | -      | -        | ✓     | ✓   | ✓      |
| Visible      | -     | ✓    | ✓   | ✓      | ✓        | ✓     | ✓   | ✓      |

(2) Motion settings for robot (Pick & Place)
You can set the following robot motions.

Pick
The robot specified by “Robot” grasps the object specified by “Object”.
Grasped object is registered as the part of the robot. Also, if any tool number is specified to “Tool”, you can operate grasped motion by using the specified tool number. If the “Tool” settings are omitted, use Tool0 to operate grasped motion.
You cannot grasp the object that is already registered as the part or set as an arm installation tool. Also, you cannot grasp the camera.
Place
The robot specified by “Robot” places the object specified by “Object”. The placed object is deregistered as the part of the robot.
You cannot place the objects which registrations are already deregistered.

You can grasp or place the object by combinations shown in the list below.

<table>
<thead>
<tr>
<th>Motion</th>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pick</td>
<td>Robot</td>
</tr>
<tr>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Place</td>
<td>✓</td>
</tr>
</tbody>
</table>

(3) Operation settings for object (specify the parent object)
You can set operations for the following objects

SetParent
Set the object specified by “ParentObject” as the parent object for the object specified by “Object”.
“ParentObject” can be omitted. In that case, the object specified by “Object” will be the parent object. If the object specified by “Object” is a child object of some object, the setting as the child object is deregistered.
If the object specified by “Object” is registered as part or arm installation tool, you cannot specify the parent object.

The objects that can specify the SetParent are as follows. For the camera object, only the object set as a fixed camera can use the SetParent.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetParent</td>
<td>Robot</td>
</tr>
<tr>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

You can use SetParent by combinations shown in the list below.

<table>
<thead>
<tr>
<th>Child Object</th>
<th>ParentObject</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Robot</td>
</tr>
<tr>
<td>Robot</td>
<td>✓</td>
</tr>
<tr>
<td>Hand</td>
<td>✓</td>
</tr>
<tr>
<td>Box</td>
<td>✓</td>
</tr>
<tr>
<td>Sphere</td>
<td>✓</td>
</tr>
<tr>
<td>Cylinder</td>
<td>✓</td>
</tr>
<tr>
<td>Plane</td>
<td>✓</td>
</tr>
<tr>
<td>CAD</td>
<td>✓</td>
</tr>
<tr>
<td>Camera</td>
<td>✓</td>
</tr>
</tbody>
</table>

(4) Simulator settings (reset the collision detect)
You can change the following simulator settings.

ResetCollision
Reset the collision detect. If the robot and the object do not collide after executing ResetCollision, reset the collision state and update the 3D display on the simulator. If the robot and the object collide, the collision state does not be reset and 3D display of the simulator will not be updated.

See Also
SimGet
SimSet Statement Example

`SimSet SBox_1.PositionX, 100.0`

`SimSet Robot1.Pick, SBox_1, 1`

`SimSet Robot1.Place, SBox_1`

`SimSet SBox_1.SetParent, CAD_1`

`SimSet SBox_1.SetParent`

`SimSet ResetCollision`
Sin Function

Returns the sine of a numeric expression.

Syntax
Sin(radians)

Parameters
radians Real expression in Radians.

Return Values
Numeric value representing the sine of the numeric expression radians.

Description
Sin returns the sine of the numeric expression. The numeric expression (radians) must be in radian units. The value returned by the Sin function will range from -1 to 1.

To convert from radians to degrees, use the RadToDeg function.

See Also
Abs, Atan, Atan2, Cos, Int, Mod, Not, Sgn, Sqr, Str$, Tan, Val

Sin Function Example
The following example shows a simple program which uses Sin.

Function sintest
 Real x
 Print "Please enter a value in radians:"
 Input x
 Print "Sin of ", x, " is ", Sin(x)
Fend
SingularityAngle Statement

Sets the singularity neighborhood angle necessary for the singularity avoiding function.

**Syntax**

```
SingularityAngle \{angle\}
```

**Parameters**

- `angle`
  
  Specify the Joint #5 angle (real number equals to or greater than 0.1. Unit: deg) by a formula or a value for determining the wrist singularity neighborhood of the vertical 6-axis robot (including N series).

**Result**

Current SingularityAngle value will be displayed if the parameter is omitted.

**Description**

This command is enabled only when the singularity avoiding function is being used. Default is 10 deg. This command can be used to adjust the start position of the singularity avoidance. If the value smaller than the default is specified, avoidance motion starts at the point closer to the singularity. Usually, it is not necessary to change the parameter. This may be useful to reduce errors which occur when passing the singularity.

If SingularityAngle parameter is changed, the current setting is effective until the next controller startup.

**See Also**

AvoidSingulariti, SingualrityAngle Function, SingularitySpeed

**SingularityAngle Statement Example**

```
SingularityAngle 7.0 'Sets the singularity neighborhood angle at 7 degrees
```
SingularityAngle Function

Returns the SingularityAngle setting value.

Syntax

SingularityAngle

Return Values

Returns the singularity neighborhood angle (Unit: deg).

See Also

AvoidSingularity, SingularityAngle, SingularitySpeed, SingularitySpeed Function

SingularityAngle Function Example

```
Real currSingularityAngle
currSingularityAngle = SingularityAngle
```
SingularityDist Statement

Sets the singularity neighborhood distance necessary for the singularity avoiding function.

Syntax

SingularityDist {distance}

Parameters

distance Specify the distance between the point P and Joint #1 rotation axis (real number equals to or larger than 0. Unit: mm) by a formula or a value for determining the shoulder singularity neighborhood or f the vertical 6-axis robot (including N series) and RS series.

Result

Current SingularityDist value will be displayed if the parameter is omitted.

Description

This command is enabled only when the singularity avoiding function is being used. Default is 30 mm. This command can be used to adjust the start position of the singularity avoidance. If the value smaller than the default is specified, avoidance motion starts at the point closer to the singularity. Usually, it is not necessary to change the parameter. This may be useful to reduce errors which occur when passing the singularity.

If SingularityDist parameter is changed, the current setting is effective until the next controller startup.

See Also

AvoidSingularity, SingularityAngle, SingulariAngle Function, SingularityDist Function, SingularitySpeed, SingularitySpeed Function

SingularityDist Statement Example

SingularityDist 10.0 'Sets the singularity neighborhood distance at 10 mm
SingularityDist Function

Returns the SingularityDist setting value.

Syntax
SingularityDist

Return Values
Returns the singularity neighborhood distance (Unit: mm).

See Also
SingularityDist, AvoidSingularity, SingularityAngle, SingularityAngle Function, SingularitySpeed, SingularitySpeed Function

SingularityDist Function Example

```plaintext
Real currSingularityDist
currSingularityDist = SingularityDist
```
SingularitySpeed Statement

Sets the singularity neighborhood angular velocity necessary for the singularity avoiding function.

Syntax

\[ \text{SingularitySpeed \{Angular velocity\}} \]

Parameters

\text{Angular velocity} \quad \text{Specify the percentage of the Joint \#4 angular velocity with respect to the maximum angular velocity (real number equals to or greater than 0.1. Unit: \%)} \text{by a formula or a value for determining the wrist singularity neighborhood of the vertical 6-axis robot (including N series).}

Result

Current SingularitySpeed value will be displayed if the parameter is omitted.

Description

This command is enabled only when the singularity avoiding function is being used. Default is 10\%. This command can be used to adjust the start position of the singularity avoidance. If the value smaller than the default is specified, avoidance motion starts at the point closer to the singularity. Usually, it is not necessary to change the parameter. This may be useful to reduce errors which occur when passing the singularity.

If SingularitySpeed parameter is changed, the current setting is effective until the next controller startup.

See Also

AvoidSingularity Function, SingularityAngle, SingularitySpeed

SingularitySpeed Example

\[ \text{SingularitySpeed 30.0} \quad \text{Sets the singularity neighborhood angular velocity at 30\%} \]
SingularitySpeed Function

Returns the SingularitySpeed setting value.

Syntax

    SingularitySpeed

Return Values

    Returns the singularity neighborhood angular velocity (Unit: %).

See Also

    SingularitySpeed, SingularityAngle, AvoidSingularity

SingularitySpeed Function Example

    Real currSingularitySpeed
    currSingularitySpeed = SingularitySpeed
SLock Statement

Restores servo power from servo free condition for the specified servo axis.

Syntax

```
SLock jointNumber [ , jointNumber,... ]
```

Parameters

- **jointNumber**: The servo joint number (1 to 9).
  The additional S axis is 8 and T axis is 9.

Description

SLock restores servo power to the specified servo joint, which was de-energized by the SFree instruction for
the direct teaching or part installation.

- If the joint number is omitted, all joints are engaged.
- Engaging the 3rd joint (Z) causes the brake to release.
- To engage all axes, Motor On may be used instead of SLock.
- Executing SLock while in Motor Off state will cause an error.
- SLock initializes the robot control parameter.
- See Motor On for the details.

- 6-axis robots (including N series) cannot be de-energized by the SFree instruction. When SLock is executed,
an error occurs.

See Also

- Brake, LimZ, Reset, SFree

SLock Example

This is a simple example on the usage of the SLock command. To operate the robot in this example, the
[Allow motion with one or more joints free] checkbox must be selected from [Setup]-[System
Configuration]-[Controller]-[Preferences].

```
Function test
  .
  .
  .
  SFree 1, 2    ' Release the excitation of J1 and J2,
                 ' and control the Z and U joints for part installation.
  Go P1
  SLock 1, 2    ' Restore the excitation of J1 and J2.
  .
  .
Fend
```
SoftCP Statement

Specifies the SoftCP motion mode.

Syntax

```
SoftCP { On | Off }
```

Parameters

- **On** | **Off**
  - On is used to enable SoftCP motion mode.
  - Off is used to disable SoftCP motion mode.

Description

SoftCP motion mode controls the vibration caused by CP motion with high acceleration/deceleration. Normal CP motion focuses on path-tracking and uniform-motion which increases the vibration when acceleration/deceleration is high. To reduce the vibration, acceleration/deceleration needs to be reduced with the SpeedS and AccelS commands. However, some applications don’t necessarily require the high performance of path-tracking and uniform-motion but need CP motion with less vibration when acceleration/deceleration is high. SoftCP motion mode dampens the path-tracking and uniform-motion performance more than in the normal CP motion mode and reduces the vibration in CP motion with high acceleration/deceleration.

SoftCP motion mode applies to the following CP motion commands:

- Move, BMove, TMove, Arc, Arc3, CVMove, Jump3CP

If the vibration doesn’t matter in the normal CP motion or the performances of path-tracking and uniform-motion are required, don’t apply SoftCP motion mode.

Caution

**When connecting CP motion and PTP motion in CP On**

When connecting CP motion and PTP motion as shown below, be sure to enable SoftCP. If it is not enabled, noise may occur from the robot depending on the motion. After connecting CP motion and PTP motion, disable SoftCP.

```
SoftCP On
Go P1 CP
Move P2
SoftCP Off
```

See Also

SoftCP Function

SoftCP Statement Example

```
SoftCP On
Move P1
Move P2
SoftCP Off
```
SoftCP Function

Returns the status of SoftCP motion mode.

Syntax

SoftCP

Return Values

0 = SoftCP motion mode off, 1 = SoftCP motion mode on.

See Also

SoftCP Statement

SoftCP Function Example

If SoftCP = Off Then
    Print "SoftCP is off"
EndIf
Space$ Function

Returns a string of space characters.

Syntax

Space$(count)

Parameters

count The number of spaces to put in the return string.

Return Values

Returns a string of count space characters.

Description

Space$ returns a string of count space characters as specified by the user. Space$ can return up to 255 characters (the maximum number of characters allowed in a string variable).

The Space$ instruction is normally used to insert spaces before, after, or between other strings of characters.

See Also

Asc, Chr$, InStr, Left$, Len, LSet$, Mid$, Right$, RSet$, Str$, Val

Space$ Function Example

> Print "XYZ" + Space$(1) + "ABC"
XYZ ABC

> Print Space$(3) + "ABC"
ABC
>
Speed Statement

Specifies or displays the arm speed for the point to point motion instructions Go, Jump and Pulse.

Syntax

(1) Speed percent [, departSpeed, approSpeed]
(2) Speed

Parameters

percent Integer expression between 1 and 100 representing the arm speed as a percentage of the maximum speed.
departSpeed Integer expression between 1 and 100 representing the depart motion speed for the Jump instruction. Available only with Jump command.
approSpeed Integer expression between 1 and 100 representing the approach motion speed for the Jump instruction. Available only with Jump command.

Return Values
Displays current Speed value when used without parameters.

Description

Speed specifies the arm speed for all point to point motion instructions. This includes motion caused by the Go, Jump and Pulse robot motion instructions. The speed is specified as a percentage of maximum speed with the range of acceptable values between 1-100. (1 represents 1% of the maximum speed and 100 represents 100% of maximum speed). Speed 100 represents the maximum speed possible.

Depart and approach speed values apply only to the Jump instruction. If omitted, each defaults to the percent value.

The speed value initializes to its default value when any one of the following is performed:

| Controller Startup |
| Motor On |
| SFree, SLock, Brake |
| Reset, Reset Error |
| Stop button or QuitAll stops tasks |

In Low Power Mode, the effective speed setting is lower than the default value. If a higher speed is specified directly (from the command window) or in a program, the speed is set to the default value. In High Power Mode, the motion speed setting is the value specified with Speed.

If higher speed motion is required, set high power mode using Power High and close the safety door. If the safety door is open, the Speed settings will be changed to their default value.

If Speed is executed when the robot is in low power mode, the following message is displayed. The following example shows that the robot will move at the default speed (5) because it is in Low Power Mode even though the speed setting value by Speed is 80.

```shell
> speed 80
> speed
Low Power Mode
  80
    80  80
```
See Also
 acc, go, jump, power, pass, pulse, speed

Speed Statement Example
 Speed can be used from the command window or in a program. Shown below are simple examples of both methods.

Function speedtst
 Integer slow, fast, i
 slow = 10
 fast = 100
 For i = 1 To 10
   Speed slow
   Go P0
   Go P1
   Speed fast
   Go P0
   Go P1
 Next i
Fend

From the command window the user can also set Speed values.

> Speed 100,100,50    'Z joint downward speed set to 50%
> Speed 50
> Speed
   Low Power State: Speed is limited to 5
   50
   50
   50
>
Speed Function

Returns one of the three speed settings.

Syntax

\[
\text{Speed}(\text{paramNumber})
\]

Parameters

\textit{paramNumber} Integer expression which evaluates to one of the values shown below. When omitted, 1 will be taken as the specified number.

1: PTP motion speed
2: Jump depart speed
3: Jump approach speed

Return Values

Integer value from 1 to 100.

See Also

Speed

Speed Function Example

```c
Integer savSpeed

savSpeed = \text{Speed}(1)
Speed 50
Go pick
Speed savSpeed
Fend
```
SpeedFactor Statement

Sets and returns the setting value of speed factor for manipulator motions.

Syntax
(1) SpeedFactor speedRatio
(2) SpeedFactor

Parameters
speedRatio
Integer expression or value between 1 and 100 representing the speed ratio of manipulator motion. (Unit: %)

Return Values
Displays current SpeedFactor value when used without parameters.

Description
SpeedFactor specifies the speed factor for all manipulators and motions set to the Controller. Usually, SpeedFactor is set to 100 % and speed for each manipulator/motion command is set by Speed or SpeedR. SpeedFactor is useful to set specific speed to all motions of all manipulators at one time. For example, the motion with Speed = 80% operates at 40% of the speed, when speed ratio is 50%.

SpeedFactor also changes the acceleration at the same rate in consideration of a balance of acceleration and deceleration of the manipulator motion.

SpeedFactor is equivalent to the speed ratio setting in the operator window and changes along with the value.

SpeedFactor will be initialized to 100% at the Controller startup.

See Also
SpeedFactor Function

SpeedFactor Statement Example

Function main
  Motor On
  Power High
  SpeedFactor 80

  Speed 100; Accel 100,100
  Go P1  ' Operates with Speed 80; Accel 80,80

  Speed 50; Accel 50,50
  Go P2  ' Operates with Speed 40; Accel 40,40

  Fend
SpeedFactor Function

Returns SpeedFactor setting value.

Syntax

```
SpeedFactor
```

Return Values

Integer value representing the SpeedFactor setting.

See Also

- SpeedFactor Statement

Speed Function Example

```
Real savSpeedFactor

savSpeedFactor = SpeedFactor
SpeedFactor 80
Go P1
Go P2
SpeedFactor savSpeedFactor
```
Sets or displays the tool rotation speed for CP motion when ROT is used.

**Syntax**

(1) `SpeedR rotSpeed`

(2) `SpeedR`

**Parameters**

`rotSpeed` Real expression in degrees / second.

Valid entries range of the parameters: 0.1 to 1000

**Return Values**

When parameters are omitted, the current SpeedR setting is displayed.

**Description**

SpeedR is effective when the ROT modifier is used in the Move, Arc, Arc3, BMove, TMove, and Jump3CP motion commands.

The SpeedR value initializes to the default value (low speed) when any one of the following conditions occurs:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

**See Also**

`AccelR, Arc, Arc3, BMove, Jump3CP, Power, SpeedR Function, TMove`

**SpeedR Statement Example**

```
  SpeedR 200
```
SpeedR Function

Returns tool rotation speed value.

Syntax

    SpeedR

Return Values

Real value in degrees / second.

See Also

    AccelR, SpeedR

SpeedR Function Example

    Real currSpeedR
    currSpeedR = SpeedR
SpeedS Statement

Specifies or displays the arm speed for use with the continuous path motion instructions such as Move, Arc, Arc3, Jump3, and Jump3CP.

Syntax

(1) **SpeedS** `speed [, departSpeed, approSpeed]`

(2) **SpeedS**

Parameters

- `speed`  
  Real expression representing the CP motion speed in units of mm/sec.
- `departSpeed`  
  Optional. Real expression representing the Jump3 depart speed in units of mm/sec.
- `approSpeed`  
  Optional. Real expression representing the Jump3 approach speed in units of mm/sec.

Valid entries range of the parameters:
- Other than N series: 0.1 to 2000
- N series, X5 series: 0.1 to 1120

Return Values

Displays current SpeedS value when used without parameters.

Description

SpeedS specifies the tool center point speed for use with all the continuous path motion instructions. This includes motion caused by the Move and Arc instructions.

SpeedS is specified in mm/Sec which represents a Tool Center Point velocity for the robot arm. The default value varies from robot to robot. See the robot manual for the default SpeedS values for your robot model. This is the initial SpeedS value set up automatically by the controller each time main power is turned on.

The SpeedS value initializes to its default value when any one of the following is performed:

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

In Low Power Mode, the effective SpeedS setting is lower than the default value. If a higher speed is specified directly (from the command window) or in a program, the speed is set to the default value. In High Power Mode, the motion SpeedS setting is the value of SpeedS.

If higher speed motion is required, set high power mode using Power High and close the safety door. If the safety door is open, the SpeedS settings will be changed to their default value.

See Also

AccelS, Arc, Jump3, Move, Speed
**SpeedS Statement Example**

SpeedS can be used from the command window or in a program. Shown below are simple examples of both methods.

```plaintext
Function speedtst
    Integer slow, fast, i
    slow = 50
    fast = 500
    For i = 1 To 10
        SpeedS slow
        Move P0
        Move P1
        SpeedS fast
        Move P0
        Move P1
    Next i
Fend
```

From the command window the user can also set SpeedS values.

```plaintext
> speeds 1000
> speeds 500
> speed 30 ' sets point to point speed
> go p0 ' point to point move
> speeds 100 ' sets straight line speed in mm/Sec
> move P1 ' moves in straight line
```
SpeedS Function

Returns the current SpeedS setting.

Syntax

\[ \text{SpeedS} \left( \text{paramNumber} \right) \]

Parameters

\text{paramNumber} \quad \text{Optional. Integer expression specifying which SpeedS value to return.}

1: CP speed
2: Jump3 depart speed
3: Jump3 approach speed

Return Values

Real number, in mm/sec

See Also

SpeedS Statement

SpeedS Function Example

\begin{verbatim}
Real savSpeeds
savSpeeds = SpeedS
Print "Jump3 depart speed = ", SpeedS(2)
\end{verbatim}
Sqr Function

Computes the non-negative square root value of the operand.

Syntax

\[ \text{Sqr}(\text{Operand}) \]

Parameters

Operand \quad \text{A real expression.}

Return Values

Square root value.

Description

The Sqr function returns the non-negative square root value of the operand.

Potential Error

Negative operand

If the operand is or has a negative numeric value, an error will occur.

See Also

Abs, And, Atan, Atan2, Cos, Int, Mod, Not, Or, Sgn, Sin, Str$, Tan, Val, Xor

Sqr Function Example

This is a simple Command window example on the usage of the Sqr function.

```
> print sqr(2)
  1.414214
>
```

The following example shows a simple program which uses Sqr.

```
Function sqrtest
  Real x
  Print "Please enter a numeric value:"
  Input x
  Print "The Square Root of \( x \) is \( \sqrt{\text{Sqr}(x)} \)
Fend
```
ST Function

Returns the coordinate value of the specified additional axis in the point data.

Syntax

```
ST ( sValue As Real, tValue As Real )
```

Parameters

- `sValue` Real value that specifies the S axis coordinate value
- `tValue` Real value that specifies the T axis coordinate value

Return Values

Coordinate values of the specified additional axis in the point data.

Description

This function is used when you are using the additional ST axes. When using this function like `Go ST(10,20)`, the additional axis will move to the specified coordinate but the manipulator will not move. If you want to move the manipulator as well, use like `Go XY(60, 30, -50, 45) : ST(10, 20)`.

For the details of the additional axis, refer to `EPSON RC+ Users Guide: 20. Additional Axis`.

See Also

- XY Function

ST Function Example

```
P10 = ST(10, 20)
```
StartMain Statement

Executes the main function from a background task.
This command is for the experienced user and you need to understand the command specification before use.

Syntax

StartMain mainFuncname

Parameters

mainFuncname Main function name you want to execute (main ~ main63)

Description

To execute StartMain, you need to set the [Enable advanced task commands] preference in the [Setup]- [System Configuration]-[Controller]-[Preferences] page.

If a task is executed using the Xqt statement from a background task, the executed task becomes a background task. With StartMain, you can execute the main function as a non-background task from a background task.

If you have already executed the main function or execute StartMain from a non-background task, an error occurs.

CAUTION

When executing StartMain command from a program, you must understand the command specification and confirm that the system has the proper conditions for this command. Improper use such as continuous execution of a command within a loop may deteriorate the system safety.

See Also

Xqt

StartMain Statement Example

Function bgmain

    ;
    If Sw(StartMainSwitch) = On And Sw(ErrSwitch) = Off Then
    StartMain main
    EndIf
    ;
Fend
Stat Function

Returns the execution status information of the controller.

Syntax
Stat(address)

Parameters
address  Defines which status bits to check. (0 to 2)

Return Values
Returns a 4 byte value that presents the status of the controller. Refer to table below.

Description
The Stat instruction returns information as shown in the table below:

<table>
<thead>
<tr>
<th>Address</th>
<th>Bit</th>
<th>Controller Status Indicated When Bit is On</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-15</td>
<td>&amp;H1 to &amp;H8000</td>
<td>Task (1–16) is being executed (Xqt) or in Halt State</td>
</tr>
<tr>
<td>16</td>
<td>&amp;H10000</td>
<td>Task(s) is being executed</td>
</tr>
<tr>
<td>17</td>
<td>&amp;H20000</td>
<td>Pause condition</td>
</tr>
<tr>
<td>18</td>
<td>&amp;H40000</td>
<td>Error Condition</td>
</tr>
<tr>
<td>19</td>
<td>&amp;H80000</td>
<td>Teach mode</td>
</tr>
<tr>
<td>20</td>
<td>&amp;H100000</td>
<td>Emergency Stop Condition</td>
</tr>
<tr>
<td>21</td>
<td>&amp;H200000</td>
<td>Low Power Mode (Power Low)</td>
</tr>
<tr>
<td>22</td>
<td>&amp;H400000</td>
<td>Safe Guard Input is Closed</td>
</tr>
<tr>
<td>23</td>
<td>&amp;H800000</td>
<td>Enable Switch is Open</td>
</tr>
<tr>
<td>24</td>
<td>&amp;H1000000</td>
<td>Undefined</td>
</tr>
<tr>
<td>25</td>
<td>&amp;H2000000</td>
<td>Undefined</td>
</tr>
<tr>
<td>26</td>
<td>&amp;H4000000</td>
<td>Test mode</td>
</tr>
<tr>
<td>27</td>
<td>&amp;H8000000</td>
<td>T2 mode</td>
</tr>
<tr>
<td>28-31</td>
<td>&amp;H1 to &amp;H80000</td>
<td>Log of Stop above target position upon satisfaction of condition in Jump...Sense statement. (This log is erased when another Jump statement is executed).</td>
</tr>
<tr>
<td>1 0</td>
<td>&amp;H1</td>
<td>Log of stop at intermediate travel position upon satisfaction of condition in Jump...Sense statement. (This log is erased when another Jump statement is executed).</td>
</tr>
<tr>
<td>1 1</td>
<td>&amp;H2</td>
<td>Log of stop at intermediate travel position upon satisfaction of condition in Go/Jump/Move...Till statement. (This log is erased when another Go/Jump/Move...Till statement is executed.</td>
</tr>
<tr>
<td>2 2</td>
<td>&amp;H4</td>
<td>Undefined</td>
</tr>
<tr>
<td>3 3</td>
<td>&amp;H8</td>
<td>Log of stop at intermediate travel position upon satisfaction of condition in Trap statement</td>
</tr>
<tr>
<td>4 4</td>
<td>&amp;H10</td>
<td>Motor On mode</td>
</tr>
<tr>
<td>5 5</td>
<td>&amp;H20</td>
<td>Current position is home position</td>
</tr>
<tr>
<td>6 6</td>
<td>&amp;H40</td>
<td>Low power state</td>
</tr>
<tr>
<td>7 7</td>
<td>&amp;H80</td>
<td>Undefined</td>
</tr>
<tr>
<td>8 8</td>
<td>&amp;H100</td>
<td>4th Joint motor is on</td>
</tr>
<tr>
<td>9 9</td>
<td>&amp;H200</td>
<td>3rd Joint motor is on</td>
</tr>
<tr>
<td>10 10</td>
<td>&amp;H400</td>
<td>2nd Joint motor is on</td>
</tr>
<tr>
<td>11 11</td>
<td>&amp;H800</td>
<td>1st Joint motor is on</td>
</tr>
<tr>
<td>12 12</td>
<td>&amp;H1000</td>
<td>6th Joint motor is on</td>
</tr>
<tr>
<td>13 13</td>
<td>&amp;H2000</td>
<td>5th Joint motor is on</td>
</tr>
<tr>
<td>14 14</td>
<td>&amp;H4000</td>
<td>Axis T motor is on</td>
</tr>
<tr>
<td>15 15</td>
<td>&amp;H8000</td>
<td>Axis S motor is on</td>
</tr>
<tr>
<td>16 16</td>
<td>&amp;H10000</td>
<td>7th Joint motor is on</td>
</tr>
<tr>
<td>17-31</td>
<td>&amp;H1 to &amp;H8000</td>
<td>Task (17–32) is being executed (Xqt) or in Halt State</td>
</tr>
<tr>
<td>2 16-15</td>
<td>&amp;H1 to &amp;H8000</td>
<td>Task (17–32) is being executed (Xqt) or in Halt State</td>
</tr>
</tbody>
</table>
Stat Function

See Also
EStopOn Function, TillOn Function, PauseOn Function, SafetyOn Function

Stat Function Example

Function StatDemo

    rbt1_sts = RShift((Stat(0) And &H070000), 16)
Select TRUE
    Case (rbt1_sts And &H01) = 1
        Print "Tasks are running"
    Case (rbt1_sts And &H02) = 2
        Print "Pause Output is ON"
    Case (rbt1_sts And &H04) = 4
        Print "Error Output is ON"
Send
Fend
Str$ Function

Converts a numeric value to a string and returns it.

Syntax

Str$(number)

Parameters

number Integer or real expression.

Return Values

Returns a string representation of the numeric value.

Description

Str$ converts a number to a string. Any positive or negative number is valid.

See Also

Abs, Asc, Chr$, InStr, Int, Left$, Len, Mid$, Mod, Right$, Sgn, Space$, Val

Str$ Function Example

The example shown below shows a program which converts several different numbers to strings and then prints them to the screen.

```plaintext
Function strtest
    Integer intvar
    Real realvar
    intvar = -32767
    Print "intvar = ", Str$(intvar)
    realvar = 567.9987
    Print "realvar = ", Str$(realvar)
Fend
```

Some other example results from the Str$ instruction from the command window.

```plaintext
> Print Str$(99999999999999)
1.000000E+014

> Print Str$(25.999)
25.999
```
String Statement

Declares variables of type String. (Character-string variables)

Syntax

```
String varName$ [(subscripts)] [, varName$ [(subscripts)]] ...
```

Parameters

- **varName$**
  Variable name which the user wants to declare as type String.

- **subscripts**
  Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows:
  
  ```
  (ubound1, [ubound2], [ubound3])
  ```
  
  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

  - Local variable: 200
  - Global Preserve variable: 400
  - Global variable and module variable: 10,000

Description

The String statement is used to declare variables of type String. String variables can contain up to 255 characters. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

String Operators

The following operators can be used to manipulate string variables:

- **+** Merges character strings together. Can be used in the assignment statements for string variables or in the Print instruction.
  
  Example: `name$ = fname$ + " " + lname$

- **=** Compares character strings. True is returned only when the two strings are exactly equal, including case.
  
  Example: `If temp1$ = "A" Then GoSub test`

- **<>** Compares character strings. True is returned when one or more characters in the two strings are different.
  
  Example: `If temp1$ <> "A" Then GoSub test`

Note

**Variable Names Must Include "$" Character:**

Variables of type String must have the character "$" as the last character in the variable name.

See Also

- Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Real, Short, UByte, UInt32, UInt64, UShort
String Statement Example

```plaintext
String  password$
String  A$(10)         ' Single dimension array of string
String  B$(10, 10)     ' Two dimension array of string
String  C$(5, 5, 5)    ' Three dimension array of string

Print "Enter password:"
Input password$
If UCase$(password$) = "EPSON" Then
   Call RunMaintenance
Else
   Print "Password invalid!"
EndIf
```
Sw Function

Returns or displays the selected input port status. (i.e. Discrete User I/O)

Syntax

\[ \text{Sw(bitNumber)} \]

Parameters

- **bitNumber**: Integer expression representing I/O input bits.

Return Values

Returns a 1 when the specified input is On and a 0 when the specified input is Off.

Description

Sw provides a status check for hardware inputs. Sw is most commonly used to check the status of one of the inputs which could be connected to a feeder, conveyor, gripper solenoid, or a host of other devices which works via discrete I/O. Obviously the input checked with the Sw instruction has 2 states (1 or 0). These indicate whether the device is On or Off.

See Also

In, InBCD, MemOn, MemOff, MemSw, Off, On, OpBCD, Oport, Out, Wait

Sw Function Example

The example shown below simply checks the discrete input #5 and branches accordingly. On is used instead of 1 for more clarity.

```plaintext
Function main
Integer i, feed5Ready
feed5Ready = Sw(5)  // Check if feeder is ready
If feed5Ready = On Then
   Call mkpart1
Else
   Print "Feeder #5 is not ready. Please reset and"
   Print "then restart program"
EndIf
Fend
```

Other simple examples are as follows from the command window:

```plaintext
> print sw(5)
1
> 
```
SyncLock Statement

Synchronizes tasks using a mutual exclusion lock.

Syntax

SyncLock syncID [, timeOut]

Parameters

- **syncID**: Integer expression representing signal number to receive. Range is from 0 to 63.
- **timeOut**: Optional. Real expression representing the maximum time to wait for lock.

Description

Use SyncLock to lock use of a common resource so that only one task at a time can use it. When the task is finished with the resource, it must call SyncUnlock to release the lock so other tasks can use it.

A task can only unlock a syncID that it previously locked.

A task must execute SyncUnlock to release the lock.

If the task is finished, then the lock it previously locked will releases.

When SynLock is second consecutive used to a same signal number, an error occurs.

If the `timeOut` parameter is used, then the Twcmd_tw function must be used to check if the lock was successful.

Note

In EPSON RC+ 6.0 and 7.0, the lock is automatically released when the task is finished while it is not in EPSON RC+5.0.

See Also

- Signal, SyncLock, Tw, Wait, WaitPos
SyncLock Statement

SyncLock Statement Example
The following example uses SyncLock and SyncUnlock to allow only one task at a time to write a message
to a communication port.

Function Main
    Xqt Func1
    Xqt Func2
Fend

Function Func1
    Long count
    Do
        Wait .5
        count = count + 1
        LogMsg "Msg from Func1, " + Str$(count)
    Loop
Fend

Function Func2
    Long count
    Do
        Wait .5
        count = count + 1
        LogMsg "Msg from Func2, " + Str$(count)
    Loop
Fend

Function LogMsg(msg$ As String)
    SyncLock 1
    OpenCom #1
    Print #1, msg$
    CloseCom #1
    SyncUnlock 1
Fend

The following example uses SyncLock with optional time out. Tw is used to check if the lock was successful.
By using a time out, you can execute other code periodically while waiting to lock a resource.

Function MySyncLock(syncID As Integer)
    Do
        SyncLock syncID, .5
        If Tw = 0 Then
            Exit Function
        EndIf
        If Sw(1) = On Then
            Off 1
        EndIf
    Loop
Fend
SyncUnlock Statement

Unlocks a sync ID that was previously locked with SyncLock.

**Syntax**

```
SyncUnlock syncID
```

**Parameters**

- `syncID` Integer expression representing signal number to receive. Range is from 0 to 63.

**Description**

Use SyncUnlock to unlock a sync ID previously locked with SyncLock. A task can only unlock a syncID that it previously locked.

**See Also**

Signal, SyncLock, Wait, WaitPos

**SyncUnlock Statement Example**

```
Function Main
    Xqt task
    Xqt task
    Xqt task
    Xqt task
    Fend

Function task
    Do
        SyncLock 1
        Print "resource 1 is locked by task", MyTask
        Wait .5
        SyncUnlock 1
    Loop
    Fend
```
SyncRobots Statement

Start the reserved robot motion.

Syntax

```
SyncRobots robotNumber [, robotNumber] [, ...]
SyncRobots All
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>robotNumber</td>
<td>Integer expression that specifies a robot number you want to start the motion.</td>
</tr>
<tr>
<td>All</td>
<td>All robots whose motion is reserved</td>
</tr>
</tbody>
</table>

Description

SyncRobots is used to start the robot motion reserved with the `SYNC` parameter of each motion command. The robots specified by the SyncRobots start to move in the same timing. This is more useful than synchronizing the normal multi-task programs by waiting for the I/O signal event because there is no effect of switching tasks. It can synchronize the robot motion start more precisely.

If a robot number is specified whose motion is not reserved, an error occurs.

See Also

SyncRobots Function

SyncRobots Statement Example

The example below uses the `SYNC` parameter of a motion command and SyncRobots to start the motions of two robots simultaneously.

```plaintext
Function Main
   Xqt Func1
   Xqt Func2
   Do
      Wait 0.1
      If (SyncRobots And &H03) = &H03 Then
         Exit Do
      EndIf
   Loop
   SyncRobots 1,2
   Fend

Function Func1
   Robot 1
   Motor On
   Go P1 SYNC
   Fend

Function Func2
   Robot 2
   Motor On
   Go P1 SYNC
   Fend
```
SyncRobots Function

Returns the status of a robot whose motion is reserved.

Syntax

```
SyncRobots
```

Return Values

Returns the robot motion in a bit, and if not reserved, “0” is returned.

- bit 0: robotNumber 1
- bit 1: robotNumber 2
- ...
- bit 15: robotNumber 16

Description

SyncRobots function checks the motion reservation status of the `SYNC` parameter of the robot motion commands. The status the SyncRobots checks are displayed in the bit status corresponding to the robot number. Each bit shows either the robot motion is reserved (1) or not (2). You can start the robot motion reserved using the SyncRobots statement.

See Also

SyncRobots

SyncRobots Function Example

The example below uses the `SYNC` parameter of a motion command and SyncRobots to start the motions of two robots simultaneously.

```
Function Main
  Xqt Func1
  Xqt Func2
  Do
    Wait 0.1
    If (SyncRobots And &H03) = &H03 Then
      Exit Do
    EndIf
  Loop
  SyncRobots 1,2
Fend

Function Func1
  Robot 1
  Motor On
  Go P1 SYNC
Fend

Function Func2
  Robot 2
  Motor On
  Go P1 SYNC
Fend
```
SysConfig Statement

 Displays system configuration parameter.

Syntax
 SysConfig

Return Values
 Returns system configuration parameter.

Description
 Display current configured value for system control data. When the robot and controller is received from the factory or after changing the configuration, it is a good idea to save this data. This can be done with Backup Controller from the [Tools]-[Controller dialog].

The following data will be displayed. (The following data is for reference only since data will vary from controller to controller.)

' Version:
'   Firmware 1, 0, 0, 0

' Options:
'   External Control Point
'   RC+ API

' HOUR: 414.634

' Controller:
'   Serial #: 0001

' ROBOT 1:
'   Name: Mnp01
'   Model: PS3-AS10
'   Serial #: 0001
'   Motor On Time: 32.738
'   Motor 1: Enabled, Power = 400
'   Motor 2: Enabled, Power = 400
'   Motor 3: Enabled, Power = 200
'   Motor 4: Enabled, Power = 50
'   Motor 5: Enabled, Power = 50
'   Motor 6: Enabled, Power = 50

ARCH 0, 30, 30
ARCH 1, 40, 40
ARCH 2, 50, 50
ARCH 3, 60, 60
ARCH 4, 70, 70
ARCH 5, 80, 80
ARCH 6, 90, 90
ARMSET 0, 0, 0, 0, 0, 0
HOFs 0, 0, 0, 0, 0, 0
HORDR 63, 0, 0, 0, 0, 0
RANGE -7427414, 7427414, -8738134, 2621440, -3145728, 8301227, -5534152, 5534152, -3640889, 3640889, -6553600, 6553600
BASE 0, 0, 0, 0, 0
WEIGHT 2, 0
INERTIA 0.1, 0
XYLIM 0, 0, 0, 0, 0, 0

' Extended I/O Boards:
'   1: Installed
'   2: Installed
'   3: None installed
'   4: None installed

' Fieldbus I/O Slave Board:
'   Installed
'   Type: PROFIBUS

' Fieldbus I/O Master Board:
'   None installed

' RS232C Boards:
'   1: Installed
'   2: None installed

' PG Boards:
'   1: None installed
'   2: None installed
'   3: None installed
'   4: None installed

SysConfig Statement Example

> SysConfig
SysErr Function

Returns the latest error status or warning status.

Syntax

SysErr [(infoNo)]

Parameters

infoNo  Optional. Integer number representing the error code or warning code to get.
        0 : Error code (When the parameter is omitted, 0 is automatically selected.)
        1 : Warning code

Return Values

An integer representing the error code or warning code of the controller.

Description

SysErr is used only for NoEmgAbort task (special task using NoEmgAbort at Xqt) and background tasks.
Error codes or warning codes of controller are the error codes or warning codes displayed on the LCD.
When there are no errors or warnings, the return value will be 0.

See Also

ErrMsg$, ErrorOn, Trap, Xqt

SysErr Function Example

The following example shows a program that monitors the controller error and switches the I/O On/Off
according to the error number when error occurs.

Notes

Forced Flag

This program example uses Forced flag for On/Off command.
Be sure that the I/O outputs change during error, or at Emergency Stop or Safety Door Open when designing
the system.

After Error Occurrence

As this program, finish the task promptly after completing the error handling.

Function main

    Xqt ErrorMonitor, NoEmgAbort
    :
    Fend

Function ErrorMonitor

    Wait ErrorOn
    If 4000 < SysErr Then
        Print "Motion Error = ", SysErr
        Off 10, Forced
        On 12, Forced
    Else
        Print "Other Error = ", SysErr
        Off 11, Forced
        On 13, Forced
    EndIf

    Fend
**Tab$ Function**

Returns a string containing the specified number of tabs characters.

**Syntax**

\[ \text{Tab$} (\text{number}) \]

**Parameters**

- **number** Integer expression representing the number of tabs.

**Return Values**

String containing tab characters.

**Description**

Tab$ returns a string containing the specified number of tabs.

**See Also**

Left$, Mid$, Right$, Space$

**Tab$ Function Example**

```plaintext
Print "X", Tab$(1), "Y"
Print
For i = 1 To 10
  Print x(i), Tab$(1), y(i)
Next i
```
Tan Function

Returns the tangent of a numeric expression.

Syntax

Tan(radians)

Parameters

radians  Real expression given in radians.

Return Values

Real number containing the tangent of the parameter radians.

Description

Tan returns the Tangent of the numeric expression. The numeric expression (radians) may be any numeric value as long as it is expressed in radian units.

To convert from radians to degrees, use the RadToDeg function.

See Also

Abs, Atan, Atan2, Cos, Int, Mod, Not, Sgn, Sin, Sqr, Str$, Val

Tan Function Example

```epson
Function tantest
    Real num
    Print "Enter number in radians to calculate tangent for:"
    Input num
    Print "The tangent of ", num, "is ", Tan(num)
Fend
```

The examples shown below show some typical results using the Tan instruction from the Command window.

```
> print tan(0)
0.00
> print tan(45)
1.6197751905439
> 
```
TargetOK Function

Returns a status indicating whether or not the PTP (Point to Point) motion from the current position to a target position is possible.

Syntax
TargetOK(targetPos)

Parameters

targetPos  Point expression for the target position.

Return Values

True if it possible to move to the target position from the current position, otherwise False.

Description

Use TargetOK to verify that a target position and orientation can be reached before actually moving to it. The motion trajectory to the target point is not considered.

See Also

CurPos, FindPos, InPos, WaitPos

TargetOK Function Example

If TargetOK(P1) Then
    Go P1
EndIf

If TargetOK(P10 /L /F) Then
    Go P10 /L /F
EndIf
TaskDone Function

Returns the completion status of a task.

Syntax

```plaintext
TaskDone (taskIdentifier)
```

Parameters

- `taskIdentifier` Task name or integer expression representing the task number.
  Task name is a function name used in an Xqt statement or a function started from the Run window or Operator window.

Task number range is:
- Normal tasks: 1 to 32
- Background task: 65 to 80
- Trap tasks: 257 to 267

Return Values

True if the task has been completed, False if not.

Description

Use TaskDone to determine if a task has completed.

See Also

TaskState, TaskWait

TaskDone Function Example

```plaintext
Xqt 2, conveyor
Do
  .
  .
Loop Until TaskDone (conveyor)
```
TaskInfo Function

Returns status information for a task.

Syntax

TaskInfo( taskIdentifier, index)

Parameters

\textit{taskIdentifier} \quad \text{Task name or integer expression representing the task number.}

A task name is the function name used in an Xqt statement or a function started from the
Run window or Operator window.

Specifying a task number:

- Normal tasks : 1 to 32
- Background tasks : 65 to 80
- Trap tasks : 257 to 267

\textit{index} \quad \text{Integer expression that represents the index of the information to retrieve.}

Return Values

An integer containing the specified information.

Description

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Task number</td>
</tr>
</tbody>
</table>
| 1     | 0 – Normal task, NoPause task, or NoEmgAbort task
       | 1 – Background task |
| 2     | Task type
       | 0 - Normal task
       | Nothing specified in Xqt or start the task by Normal
       | 1 - NoPause task
       | Specified NoPause in Xqt and start the task
       | 2 - NoEmgAbort task
       | Specified NoEmgAbort in Xqt and start the task
       | 3 - Trap task
       | 4 - Background task |
| 3     | –1 - Specified task is not executing.
       | 1 - Specified task is executing.
       | 2 - Specified task is waiting for an event.
       | 3 - Specified task is paused or halted
       | 4 - Specified task is in quick pause state
       | 5 - Specified task is in error state |
| 4     | Timeout has occurred during wait for event (same as TW) |
| 5     | Event wait time (milliseconds) |
| 6     | Current robot number selected by the task |
| 7     | Current robot number being used by the task |

See Also

CtrlInfo, RobotInfo, TaskInfo$
TaskInfo Function Example

If (TaskInfo(1, 3) <> 0 Then
    Print "Task 1 is running"
Else
    Print "Task 1 is not running"
EndIf
TaskInfo$ Function

Returns text information for a task.

Syntax

TaskInfo$( taskIdentifier, index)

Parameters

- **taskIdentifier**: Task name or integer expression representing the task number. A task name is the function name used in an Xqt statement or a function started from the Run window or Operator window. Specifying a task number:
  - Normal tasks: 1 to 32
  - Background tasks: 65 to 80
  - Trap tasks: 257 to 267

- **index**: Integer expression that represents the index of the information to retrieve.

Return Values

A string containing the specified information.

Description

The following table shows the information that can be retrieved using TaskInfo$:

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Task name</td>
</tr>
<tr>
<td>1</td>
<td>Start date / time</td>
</tr>
<tr>
<td>2</td>
<td>Name of function currently executing</td>
</tr>
<tr>
<td>3</td>
<td>Line number in the program file that contains the function</td>
</tr>
</tbody>
</table>

See Also

CtrlInfo, RobotInfo, TaskInfo

TaskInfo$ Function Example

```plaintext
Print "Task 1 started: " TaskInfo$(1, 1)
```
Returns the current state of a task.

Syntax

```
TaskState( taskIdentifier )
```

Parameters

- **taskIdentifier**: Task name or integer expression representing the task number. A task name is the function name used in an Xqt statement or a function started from the Run window or Operator window.
  - Specifying a task number:
    - Normal tasks: 1 to 32
    - Background tasks: 65 to 80
    - Trap tasks: 257 to 267

Return Values

- 0: Task not running
- 1: Task is running
- 2: Task is waiting for an event
- 3: Task has been halted
- 4: Task has been paused in QuickPause
- 5: Task in error condition

Description

Use TaskState to get status for a given task. You can specify task number or task name.

See Also

- TaskDone, TaskWait

TaskState Function Example

```
If TaskState(conveyor) = 0 Then
    Xqt 2, conveyor
EndIf
```
TaskWait Statement

Waits to for a task to terminate.

Syntax

TaskWait (taskIdentifier)

Parameters

taskIdentifier

Task name or integer expression representing the task number. Task name is a function name used in an Xqt statement or a function started from the Run window or Operator window.

Task number range is:
- Normal tasks: 1 to 32
- Background task: 65 to 80
- Trap tasks: 257 to 267

See Also

TaskDone, TaskState

TaskWait Statement Example

Xqt 2, conveyor
TaskWait conveyor
TC Statement

Returns the torque control mode setting and current mode.

Syntax

(1) TC { On | Off }
(2) TC

Parameters

On | Off
On : Torque control mode ON
Off : Torque control mode OFF

Return Values

When the parameter is omitted, returns the current torque control mode.

Description

TC On/Off set the torque control mode available/unavailable.
The torque control mode sets the motor output limit to generate the constant force. This is used in pressing a hand to an object at constant force or making the close contact and coordinate moving of hand with an object.
Before setting the torque control available, configure the limits of torque control and speed control in TCLim and TCspeed.
Under the torque control, the robot moves as positioning to the target while an operation command is executed. When the robot contact an object and motor output is at the torque control limit, the robot stops its operation and keeps the constant torque.

In any of the following cases, the torque mode turns unavailable.

- Controller Startup
- Motor On
- SFree, SLock, Brake
- Reset, Reset Error
- Stop button or QuitAll stops tasks

See Also

TCLim, TCspeed

TC Statement Example

Speed 5
Go ApproachPoint

' Set the Z axis torque limit to 20 %
TCLim -1, -1, 20, -1
' Set the speed in torque control to 5 %
TcSpeed 5

TC On
Go ContactPoint
Wait 3
Go ApproachPoint
TC Off
TCLim Statement

Specifies the torque limit of each joint for the torque control mode.

Syntax

```
TCLim [j1Torque limit, j2Torque limit, j3Torque limit, j4Torque limit [, j5Torque limit]
```

Parameters

- **j1Torque limit**: Specifies the proportion to the maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j2Torque limit**: Specifies the proportion to the maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j3Torque limit**: Specifies the proportion to the maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j4Torque limit**: Specifies the proportion to the maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j5Torque limit**: Option. Specifies the proportion to the maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j6Torque limit**: Option. Specifies the proportion to the maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j7Torque limit**: Option. Specifies the proportion to the S axis maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

- **j8Torque limit**: Option. Specifies the proportion to the T axis maximum momentary torque (1 to 100 / unit: %) using an expression or numeric value.
  - -1: Disable the torque limit and turns the mode to normal position control.

Return Values

- When the parameters are omitted, returns the current torque limit.

Description

- Setting to the torque limit becomes available at TC On.

  When the limit value is too low, the robot doesn’t work and operation command stops before the robot reaches the target position.
TCLim Statement

In any of the following cases, TCLim set value is initialized.

Controller Startup
Motor On
SFree, SLock, Brake
Reset, Reset Error
Stop button or QuitAll stops tasks

See Also
TC, TCLim Function, TCSpeed

TCLim Statement Example

Speed 5
Go ApproachPoint

' Set the Z axis torque limit to 20 %
TCLim -1, -1, 20, -1
' Set the speed in torque control to 5 %
TcSpeed 5

TC On
Go ContactPoint
Wait 3
Go ApproachPoint
TC Off
TCLim Function

Returns the torque limit of specified joint.

Syntax

TCLim (jointNumber)

Parameters

jointNumber Specifies the joint number to retrieve the torque limit from using an expression or numeric value.
The additional S axis is 8 and T axis is 9.

Return Values

Returns the integer number representing the current torque limit (1 to 100). -1 means the torque limit is invalid.

See Also

TC, TCLim, TCSpeed

TCLim Function Example

Print "Current Z axis torque limit:", TCLim(3)
Returns the calculated current tool center point (TCP) speed.

Syntax

```
TCPSpeed
```

Return Values

Real value containing the calculated current tool center point speed in mm/second.

Description

Use TCPSpeed to get the calculated current speed of the tool center point in mm/second when executing a CP (Continuous Path) motion command. CP motion commands include Move, TMove, Arc, Arc3, CVMove, and Jump3CP. This is not the actual tool center point speed. It is the speed that the system has calculated for the tool center point at the time the function is called.

The actual follow-up delay of the motor has been excluded from this value.

If the robot is executing a PTP (Point to Point) motion command, this function returns “0”.

Even if you are using the additional axis, only the robot travel distance is returned. For example, it doesn’t include the travel speed of additional axis while you use the additional axis as running axis.

See Also

```
AccelS, CurPos, InPos, SpeedS
```

TCPSpeed Function Example

```
Function MoveTest
    AccelS 4000, 4000
    SpeedS 200
    Xqt ShowTCPSpeed
    Do
        Move P1
        Move P2
    Loop
Fend

Function ShowTCPSpeed
    Do
        Print "Current TCP speed is: ", TCPSpeed
        Wait .1
    Loop
Fend
```
TCSpeed Statement

Specifies the speed limit in the torque control.

Syntax

TCSpeed [speed]

Parameters

speed Specifies the proportion to the maximum speed (1 to 100 / unit: %) using an expression or numeric value.

Description

Under the torque control, the speed is limited to the TCSpeed setting despite of the speed settings of such as Speed command.

Error occurs if the speed goes over the limit in the torque control.

In any of the following cases, TCSpeed set value is initialized to 100%.

Controller Startup
Motor On
SFree, SLock, Brake
Reset, Reset Error
Stop button or QuitAll stops tasks

See Also

TC, TCLim, TCSpeed Function

TCSpeed Statement Example

```
Speed 5
Go ApproachPoint

* Set the Z axis torque limit to 20%
TCLim -1, -1, 20, -1
* Set the speed under the torque control to 5%
TcSpeed 5

TC On
Go ContactPoint
Wait 3
Go ApproachPoint
TC Off
```
TCSpeed Function

Returns the speed limit in the torque control.

Syntax
TCSpeed

Return Values
Returns the integer number (1 to 100) representing the current speed limit.

See Also
TC, TCSpeed, TCLim

TCSpeed Function Example

Integer var
var = TCSpeed
TeachOn Function

Returns the Teach mode status.

Syntax

TeachOn

Return Values

True if it is in the Teach mode, False if not.

Description

TeachOn function is only used in the background task.

See Also

ErrorOn, EstopOn, SafetyOn, Xqt

TeachOn Function Example

The following example monitors the controller as it starts in Teach mode, and turns On/Off the I/O.

Function BGM ain
    Do
        Wait 0.1
        If TeachOn = True Then
            On teachBit
        Else
            Off teachBit
        EndIf
        If SafetyOn = True Then
            On safetyBit
        Else
            Off safetyBit
        EndIf
        If PauseOn = True Then
            On PauseBit
        Else
            Off PauseBit
        EndIf
    Loop
Fend
TGo Statement

Executes Point to Point relative motion, in the current tool coordinate system.

Syntax

TGo destination [CP] [PerformMode modeNumber] [searchExpr] [!]...! [SYNC]

Parameters

**destination**
The target destination of the motion using a point expression.

**CP**
Optional. Specifies continuous path motion.

**PerformMode**
Optional. Specify the robot performance mode.

**modeNumber**
Specify the operation mode assigned to PerformMode with an integer value (1 to 3) or with the following constant. If PerformMode is specified, this parameter cannot be omitted.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode_Standard</td>
<td>1</td>
<td>Sets the Standard mode</td>
</tr>
<tr>
<td>Mode_High_Speed</td>
<td>2</td>
<td>Sets the High-speed mode</td>
</tr>
<tr>
<td>Mode_Low_Oscillation</td>
<td>3</td>
<td>Sets the Low-oscillation mode</td>
</tr>
</tbody>
</table>

**searchExpr**
Optional. A Till or Find expression.

- **Till**
  - Till Sw(expr) = {On | Off}

- **Find**
  - Find Sw(expr) = {On | Off}

- **!...!**
  Optional. Parallel Processing statements can be added to execute I/O and other commands during motion.

- **SYNC**
  Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Executes point to point relative motion in the current tool coordinate system.

Arm orientation attributes specified in the destination point expression are ignored. The manipulator keeps the current arm orientation attributes. However, for a 6-Axis manipulator (including N series), the arm orientation attributes are automatically changed in such a way that joint travel distance is as small as possible.

The Till modifier is used to complete TGo by decelerating and stopping the robot at an intermediate travel position if the current Till condition is satisfied.

The Find modifier is used to store a point in FindPos when the Find condition becomes true during motion.

When parallel processing is used, other processing can be executed in parallel with the motion command.

The CP parameter causes acceleration of the next motion command to start when the deceleration starts for the current motion command. In this case the robot will not stop at the destination coordinate and will continue to move to the next point.

Deceleration motion and acceleration motion of different modes can be combined when PerformMode is set while the path motion is enabled. Some combinations are not available depending on operation modes. For details, refer to PerformMode Statement.

See Also

Accel, CP, Find, !...! Parallel Processing, P#= (Point Assignment), PerformMode, Speed, Till, TMove, Tool
TGo Statement Example

> TGo XY(100, 0, 0, 0) 'Move 100 mm in X direction (in the tool coordinate system)
Function TGoTest

   Speed 50
   Accel 50, 50
   Power High

   Tool 0
   P1 = XY(300, 300, -20, 0)
   P2 = XY(300, 300, -20, 0) /L

   Go P1
   Print Here
   TGo XY(0, 0, -30, 0)
   Print Here

   Go P2
   Print Here
   TGo XY(0, 0, -30, 0)
   Print Here

Fend

[Output]
X:  300.000 Y:  300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y:  300.000 Z:  -50.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y:  300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
X:  300.000 Y:  300.000 Z:  -50.000 U:  0.000 V:  0.000 W:  0.000 /L /0
**Till Statement**

Specifies and displays event condition that, if satisfied, completes the motion command (Jump, Go, Move, etc.) in progress by decelerating and stopping the robot at an intermediate position.

**Syntax**

```
Till [ eventcondition ]
```

**Parameters**

- `eventcondition` Input status specified as a trigger

  `[Event] comparative operator ( =, <>, >, <, <=) [Integer expression]`

  The following functions and variables can be used in the `Event`:

  **Functions**

  - Sw, In, InW, Oport, Out, OutW, MemSw, MemIn, MemInW, Ctr
  - GetRobotInsideBox, GetRobotInsidePlane, Force, AIO_In, AIO_InW, AIO_Out, AIO_OutW

  **Variables**

  - Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort global
  - preserve variable, Global variable, module variable

  In addition, using the following operators you can specify multiple event conditions.

  **Operator**

  - And, Or, Xor

  **Example**

  - Till Sw(5) = On
  - Till Sw(5) = On And Till(6) = Off

**Description**

The Till statement can be used by itself or as a search expression in a motion command statement. The Till condition must include at least one of the functions above.

When variables are included, their values are computed when setting the Till condition. No use of variable is recommended. Otherwise, the condition may be an unintended condition. Multiple Till statements are permitted. The most recent Till condition remains current until superseded.

When parameters are omitted, the current Till definition is displayed.

**Notes**

**Till Setting at Main Power On**

At power on, the Till condition is initialized to Till Sw(0) = On.

**Use of Stat or TillOn to Verify Till**

After executing a motion command which uses the Till qualifier there may be cases where you want to verify whether or not the Till condition was satisfied. This can be done through using the Stat function or the TillOn function.

**To use a variables in the event condition expression**

- Available variables are Integer type (Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)
- Array variables are not available
- Local variables are not available
- If a variable value cannot satisfy the event condition for more than 0.01 seconds, the system cannot retrieve the change in variables.
- Up to 64 can wait for variables in one system (including the ones used in the event condition expressions such as Wait). If it is over 64, an error occurs during the project build.
- If you specify Byref to a waiting variable on any function call, an error occurs.
- When a variable is included in the right side member of the event condition expression, the value is calculated when starting the motion command. We recommend not using variables in an integer expression to avoid making unintended conditions.
See Also
Find, Go, In, InW, Jump, MemIn, MemSw, Move, Stat, Sw, TillOn

Till Statement Example
Shown below are some sample lines from programs using the Till instruction.

```
Till Sw(1) = Off                        ' Specifies Till condition (Input bit 1 off)
Go P1 Till                              ' Stop if previous line condition is satisfied
Till Sw(1) = On And Sw($1) = On        ' Specify new Till condition
Move P2 Till                            ' Stop if previous line condition satisfied
Move P5 Till Sw(10) = On               ' Stop if condition on this line is satisfied
```
TillOn Function

Returns the current Till status.

Syntax

TillOn

Return Values

True if the Till condition occurred in the previous motion command using Till.

Description

TillOn returns True if Till condition occurred.

TillOn is equivalent to ((Stat(1) And 2) <> 0).

See Also

EStopOn, SafetyOn, Sense, Stat, Till

TillOn Function Example

Go P0 Till Sw(1) = On
If TillOn Then
    Print "Till condition occurred during move to P0"
EndIf
Time Statement

Displays the current time.

Syntax

Time

Description

Displays the current time in 24 hour format.

See Also

Date, Time$

Time Statement Example

Example from the command window:

> Time
10:15:32
Time Function

Returns the controller accumulated operating time.

Syntax

\texttt{Time(unitSelect)}

Parameters

\texttt{unitSelect} \hspace{1em} An integer number ranging from 0 to 2. This integer specifies which unit of time the controller returns:

- 0: hours
- 1: minutes
- 2: seconds

Description

Returns the controller accumulated operating time as an integer.

See Also

Hour

Time Function Example

Shown below are a few examples from the command window:

\begin{verbatim}
Function main
    Integer h, m, s
    h = \texttt{Time(0)}  ' Store the time in hours
    m = \texttt{Time(1)}  ' Store the time in minutes
    s = \texttt{Time(2)}  ' Store the time in seconds
    Print "This controller has been used:"
    Print h, "hours, ",
    Print m, "minutes, ",
    Print s, "seconds"
Fend
\end{verbatim}
Time$ Function

Returns the current system time.

Syntax

```
Time$
```

Return Values

A string containing the current time in 24 hour format $hh:mm:ss$.

See Also

Date, Date$, Time

Time$ Function Example

```
Print "The current time is: ", Time$
```
TLCIr Statement

Cleans (undefines) a tool coordinate system.

Syntax

```
TLCIr  toolNumber
```

Parameters

- `toolNumber` Integer expression representing which of the 3 tools to clear (define).
  (Tool 0 is the default tool and cannot be cleared.)

Description

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

See Also

- Arm, ArmClr, ArmSet, ECPSet, Local, LocalClr, Tool, TLSet

TLCIr Statement Example

```
TLCIr  1
```
**TLDef Function**

Returns tool definition status.

**Syntax**

```
TLDef (toolNumber)
```

**Parameters**

- `toolNumber` Integer expression representing which tool to return status for.

**Return Values**

True if the specified tool has been defined, otherwise False.

**See Also**

Arm, ArmClr, ArmSet, ECPSel, Local, LocalClr, Tool, TLClr, TLM

**TLDef Function Example**

```
Function DisplayToolDef(toolNum As Integer)
    If TLDef(toolNum) = False Then
        Print "Tool ", toolNum, "is not defined"
    Else
        Print "Tool ", toolNum, ": ",
        Print TLSet(toolNum)
    EndIf
EndFunction
```
TLSet Statement

Defines or displays a tool coordinate system.

Syntax
(1) TLSet toolNum, toolDefPoint
(2) TLSet toolNum
(3) TLSet

Parameters

*toolNum*  
Integer number from 1 to 15 representing which of 15 tools to define. (Tool 0 is the default tool and cannot be modified.)

*toolDefPoint*  
*P* number or *P*(expr) or point label or point expression.

Return Values

When parameters are omitted, displays all TLSet Definition.
When only the tool number is specified, displays specified TLSet Definition.

Description

Defines the tool coordinate systems Tool 1, Tool 2 or Tool 3 by specifying tool coordinate system origin and rotation angle in relation to the Tool 0 coordinate system (Hand coordinate system).

TLSet 1, XY(50, 100, -20, 30)

TLSet 2, P10 +X(20)

In this case, the coordinate values of P10 are referenced and 20 is added to the X value. Arm attribute and local coordinate system numbers are ignored.

TLSET 1, XY(100, 60, -20, 30)
**TLSet for 6-Axis robots**

The origin of Tool 0 is the flange side of the sixth joint. When all joints are at the 0 degree position, the Tool 0 coordinate system's X axis is aligned with the robot coordinate system's Z axis, the Y axis is aligned with the robot coordinate system's X axis, and the Z axis is perpendicular to the flange face, and is aligned with the robot coordinate system's Y axis, as shown in the figure below:
Tool 0 coordinate systems are defined for ceiling and wall mounted robots as shown in the figures below.

**TLSet for N series robots**

When all joints are at the 0 degree position, the Tool 0 coordinate system's X axis is aligned with the robot coordinate system's -X axis, the Y axis is aligned with the robot coordinate system's Y axis, and the Z axis is aligned with the robot coordinate system's -Z axis, as shown in the figure below:
Tool 0 coordinate systems are defined for ceiling and wall mounted robots as shown in the figures below.

**Description**

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

**Note**

**TLSet values are maintained**

The TLSet values are preserved. Use TLClr to clear a tool definition.

**See Also**

Tool, Arm, ArmSet, TLClr

**TLSet Statement Example**

The example shown below shows a good test which can be done from the command window to help understand the difference between moving when a tool is defined and when no tool is defined.

```plaintext
> TLSet 1, XY(100, 0, 0, 0)  'Define tool coordinate system for Tool 1 (plus 100 mm in x direction from hand coordinate system)
> Tool 1                      'Selects Tool 1 as defined by TLSet
> TGo P1                      'Positions the Tool 1 tip position at P1
> Tool 0                      ' Tells robot to use no tool for future motion
> Go P1                       'Positions the center of the U-Joint at P1
```
Returns a point containing the tool definition for the specified tool.

Syntax

```
TLSet(toolNumber)
```

Parameters

- `toolNumber` Integer expression representing the number of the tool to retrieve.

Return Values

A point containing the tool definition.

See Also

TLSet Statement

TLSet Function Example

```
P1 = TLSet(1)
```
TMOut Statement

Specifies the number of seconds to wait for the condition specified with the Wait instruction to come true before issuing a timeout error (error 2280).

Syntax

```
TMOut seconds
```

Parameters

```
seconds  Real expression representing the number of seconds until a timeout occurs.
Valid range is 0 to 2147483 seconds in 1 second intervals.
```

Description

TMOut sets the amount of time to wait (when using the Wait instruction) until a timeout error is issued. If a timeout of 0 seconds is specified, then the timeout is effectively turned off. In this case the Wait instruction waits indefinitely for the specified condition to be satisfied.

The default initial value for TMOut is 0.

See Also

In, MemSw, OnErr, Sw, TW, Wait

TMOut Statement Example

```
TMOut 5
Wait MemSw(0) = On
```
TMove Statement

Executes linear interpolation relative motion, in the current tool coordinate system

Syntax

TMove destination [ROT] [CP] [ searchExpr ] [ !...! ] [SYNC]

Parameters

destination
The target destination of the motion using a point expression.

ROT
Optional. Decides the speed/acceleration/deceleration in favor of tool rotation.

CP
Optional. Specifies continuous path motion.

searchExpr
Optional. A Till or Find expression.

Till | Find
Till Sw(expr) = {On | Off}
Find Sw(expr) = {On | Off}

!...!
Optional. Parallel Processing statements can be added to execute I/O and other commands during motion.

SYNC
Reserves a motion command. The robot will not move until SyncRobots is executed.

Description

Executes linear interpolated relative motion in the current tool coordinate system.

Arm orientation attributes specified in the destination point expression are ignored. The manipulator keeps the current arm orientation attributes. However, for a 6-Axis manipulator (including N series), the arm orientation attributes are automatically changed in such a way that joint travel distance is as small as possible. This is equivalent to specifying the LJM modifier parameter for Move statement. Therefore, if you want to change the arm orientation larger than 180 degrees, execute it in several times.

TMove uses the SpeedS speed value and AccelS acceleration and deceleration values. Refer to Using TMove with CP below on the relation between the speed/acceleration and the acceleration/deceleration. If, however, the ROT modifier parameter is used, TMove uses the SpeedR speed value and AccelR acceleration and deceleration values. In this case SpeedS speed value and AccelS acceleration and deceleration value have no effect.

Usually, when the move distance is 0 and only the tool orientation is changed, an error will occur. However, by using the ROT parameter and giving priority to the acceleration and the deceleration of the tool rotation, it is possible to move without an error. When there is not an orientational change with the ROT modifier parameter and movement distance is not 0, an error will occur.

Also, when the tool rotation is large as compared to move distance, and when the rotation speed exceeds the specified speed of the manipulator, an error will occur. In this case, please reduce the speed or append the ROT modifier parameter to give priority to the rotational speed / acceleration / deceleration.

The Till modifier is used to complete TMove by decelerating and stopping the robot at an intermediate travel position if the current Till condition is satisfied.

The Find modifier is used to store a point in FindPos when the Find condition becomes true during motion.

When parallel processing is used, other processing can be executed in parallel with the motion command.
Note

Using TMove with CP

The CP parameter causes the arm to move to destination without decelerating or stopping at the point defined by destination. This is done to allow the user to string a series of motion instructions together to cause the arm to move along a continuous path while maintaining a specified speed throughout all the motion. The TMove instruction without CP always causes the arm to decelerate to a stop prior to reaching the point destination.

See Also
AccelS, CP, Find, !....! Parallel Processing, Point Assignment, SpeedS, TGo, Till, Tool

TMove Statement Example

> TMove XY(100, 0, 0, 0) 'Move 100 mm in the X direction (in the tool coordinate system)
Function TMoveTest

  Speed 50
  Accel 50, 50
  SpeedS 100
  AccelS 1000, 1000
  Power High

  Tool 0
  P1 = XY(300, 300, -20, 0)
  P2 = XY(300, 300, -20, 0) /L

  Go P1
  Print Here
  TMove XY(0, 0, -30, 0)
  Print Here

  Go P2
  Print Here
  TMove XY(0, 0, -30, 0)
  Print Here

  Fend

[Output]
X:  300.000 Y: 300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y: 300.000 Z:  -50.000 U:  0.000 V:  0.000 W:  0.000 /R /0
X:  300.000 Y: 300.000 Z:  -20.000 U:  0.000 V:  0.000 W:  0.000 /L /0
X:  300.000 Y: 300.000 Z:  -50.000 U:  0.000 V:  0.000 W:  0.000 /L /0
Tmr Function

Tmr function which returns the amount of time in seconds which has elapsed since the timer was started.

Syntax

```
Tmr(timerNumber)
```

Parameters

`timerNumber` Integer expression representing which of the 64 timers to check the time of. (0 to 63)

Return Values

Elapsed time for the specified timer as a real number in seconds. Valid range is 0 to approx. 1.7E+31. Timer resolution is 0.001 seconds.

Description

Returns elapsed time in seconds since the timer specified was started. Unlike the ElapsedTime function, the Tmr function counts the time while the program is halted.

Timers are reset with TmReset.

```
Real overhead

TmReset 0
overHead = Tmr(0)
```

See Also

ElapsedTime Function, TmReset

Tmr Function Example

```
TmReset 0  ' Resets Timer 0
For i = 1 To 10  ' Performs operation 10 times
    GoSub Cycle
Next
Print Tmr(0) / 10  ' Calculates and display cycle time
```
TmReset Statement

Resets the timers used by the Tmr function.

Syntax

TmReset  timerNumber

Parameters

  timerNumber        Integer expression representing which of the 64 timers to reset. (0 to 63)

Description

Resets and starts the timer specified by timerNumber.

Use the Tmr function to retrieve the elapsed time for a specific timer.

See Also

  Tmr

TmReset Statement Example

  TmReset 0          ' Resets Timer 0
  For i = 1 To 10    ' Performs operation 10 times
    GoSub CYL
  Next
  Print Tmr(0)/10    ' Calculates and display cycle time
Toff Statement

Turns off execution line display on the LCD.

Syntax

    Toff

Description

Execution line will not be displayed on the LCD.

See Also

    Ton

Toff Statement Example

    Function main
        Ton MyTask
        ...
        Toff
    Fend
Ton Statement

Specifies a task which shows an execution line on the LCD.

Syntax

```
Ton taskIdentifier
Ton
```

Parameters

- `taskIdentifier`: Task name or integer expression representing the task number. Task name is a function name used in an Xqt statement or a function started from the Run window or Operator window.

  Task number range is:
  Normal tasks : 1 to 32

Description

Execution line of task 1 is displayed in initial status.
Ton statement displays the specified task execution line on the LCD.
When `taskIdentifier` is omitted, the task execution line with Ton statement execution is displayed on the LCD.

See Also

- Toff

Ton Statement Example

```
Function main
  Ton MyTask
  ...
  Toff
Fend
```
Tool Statement

Selects or displays the current tool.

**Syntax**

(1) Tool toolNumber  
(2) Tool

**Parameters**

*toolNumber*  
Optional. Integer expression from 0 to 15 representing which of 16 tool definitions to use with subsequent motion instructions.

**Return Values**

Displays current Tool when used without parameters.

**Description**

Tool selects the tool specified by the tool number (toolNum). When the tool number is “0”, no tool is selected and all motions are done with respect to the center of the end effector joint. However, when Tool entry 1, 2, or 3 is selected motion is done with respect to the end of the tool as defined with the tool definition.

**Note**

*Power Off and Its Effect on the Tool Selection*

Turning main power off does not change the tool coordinate system selection.

*Lifetime of Compact Flash*

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

**See Also**

TGo, TLSet, Tmove

**Tool Statement Example**

The example shown below shows a good test which can be done from the command window to help understand the difference between moving when a tool is defined and when no tool is defined.

```
>tlset 1,100,0,0,0  
\> Define tool coordinate system for Tool 1 (plus 100 mm in x direction from hand coordinate system)
>tool 1  
\>Selects Tool 1 as defined by TLSet
>tgop1
\>Positions the Tool 1 tip position at P1
>tool 0  
\>Tells robot to use no tool for future motion
>go p1  
\>Positions the center of the U-Joint at P1
```
**Tool Function**

Returns the current tool number.

**Syntax**

```
Tool
```

**Return Values**

Integer containing the current tool number.

**See Also**

Tool Statement

**Tool Function Example**

```
Integer savTool

savTool = Tool
Tool 2
Go P1
Tool savTool
```
Trap Statement (User defined trigger)

Defines interrupts and what should happen when they occur.
With the Trap statement, you can jump to labels or call functions when the event occurs.
Trap statement has 2 types as below:
- 4 Traps that interrupts by the user defined input status
- 7 Traps that interrupts by the system status
Trap with user defined trigger is explained here.

Syntax

\[
\text{Trap } trapNumber, \text{ eventCondition GoTo label} \\
\text{Trap } trapNumber, \text{ eventCondition Call funcname} \\
\text{Trap } trapNumber, \text{ eventCondition Xqt funcname} \\
\text{Trap } trapNumber
\]

Parameters

- \textit{trapNumber} Integer number from 1 to 4 representing which of 4 Trap numbers to use.
  (SPEL+ supports up to 4 active Trap interrupts at the same time.)
- \textit{eventCondition} Input status specified as a trigger
  \[[\text{Event}] \text{ comparative operator} (\text{ }, =, <>, >=, >, <, <=) \text{ [Integer expression]}\]
  The following functions and variables can be used in the \textit{Event}:
  Functions : Sw, In, InW, Oport, Out, OutW, MemSw, MemIn, MemInW, Ctr,
  GetRobotInsideBox, GetRobotInsidePlane, AIO_In, AIO_InW,
  AIO_Out, AIO_OutW
  Variables : Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort
global preserve variable, Global variable, module variable
  In addition, using the following operators you can specify multiple event conditions.
  Operator : And, Or, Xor
  Example : Trap 1, Sw(5) = On Call, TrapFunc
  Trap 1, Sw(5) = On And Till(6) = Off, Call TrapFunc
- \textit{label} The label where program execution is to be transferred when Trap condition is satisfied.
- \textit{funcName} The function that is executed when Call or Xqt when the Trap condition is satisfied.
The function with argument cannot be specified.

Description

A Trap executes interrupt processing which is specified by GoTo, Call, or Xqt when the specified condition
is satisfied.
The Trap condition must include at least one of the functions above.
When variables are included in the Trap condition, their values are computed when setting the Trap condition.
No use of variable is recommended. Otherwise, the condition may be an unintended condition.
Once the interrupt process is executed, its Trap setting is cleared. If the same interrupt process is necessary,
the Trap instruction must execute it again.

To cancel a Trap setting simply execute the Trap instruction with only the \textit{trapNumber} parameter. e.g. “Trap 3” cancels Trap #3.
When the Function that executed Trap GoTo ends (or exit), the Trap Goto will be canceled automatically.
When the declared task ends, Trap Call will be canceled.
Trap Xqt will be canceled when all tasks have stopped.
**If GoTo is specified**
In the task set to Trap, the command being executed will be processed as described below. Then, control branches to the specified label.
- Any arm motion will pause immediately
- Waiting status by the Wait or Input commands will discontinue
- All other commands will complete execution before control branches

**If Call is specified**
After executing the same process as GoTo described above, then control branches to the specified line number or label.
Once the function ends, program execution returns to the next statement after the statement where program interruption occurred. Call statements cannot be used in the Trap processing function.
When an error occurs in the trap process function, error handling with OnErr will be invalid and an error will occur.

**If Xqt is specified**
Program control executes the specified function as an interrupt processing task. In this case, the task which executes the Trap command will not wait for the Trap function to finish and will continue to execute.
You cannot execute a task with an Xqt statement from an interrupt processing task.

**Notes**

**For EPSON RC+4.x user**
The Trap Call function of EPSON RC+ 4.x or before is replaced with Trap Xqt in EPSON RC+ 7.0.
The Trap GoSub function of EPSON RC+ 4.x or before is removed in EPSON RC+ 7.0. Instead, use Trap Call.

**To use a variables in the event condition expression**
- Available variables are Integer type (Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)
- Array variables are not available
- Local variables are not available
- If a variable value cannot satisfy the event condition for more than 0.01 seconds, the system cannot retrieve the change in variables.
- Up to 64 can wait for variables in one system (including the ones used in the event condition expressions such as Wait). If it is over 64, an error occurs during the project build.
- If you specify Byref to a waiting variable on any function call, an error occurs.
- When a variable is included in the right side member of the event condition expression, the value is calculated when setting the Trap condition. We recommend not using variables in an integer expression to avoid making unintended conditions.

**See Also**
Call, GoTo, Xqt
Trap Statement Example

Example 1: Error process defined by User
Sw(0) Input is regarded as an error input defined by user.

```plaintext
Function Main
  Trap 1, Sw(0) = On GoTo EHandle  ' Defines Trap
  .
  .
EHandle:
  On 31  'Signal tower lights
  OpenCom #1
  Print #1, "Error is issued"
  CloseCom #1
Fend
```

Example 2: Usage like multi-tasking

```plaintext
Function Main
  Trap 2, MemSw(0) = On Or MemSw(1) = On Call Feeder
  .
  .
Fend

Function Feeder
  Select TRUE
  Case MemSw(0) = On
    MemOff 0
    On 2
  Case MemSw(1) = On
    MemOff 1
    On 3
  Send

  ' Re-arm the trap for next cycle
  Trap 2, MemSw(0) = On Or MemSw(1) = On Call Feeder
Fend
```

Example 3: Using global variable as event condition

```plaintext
Global Integer gi

Function main
  Trap 1, gi = 5 GoTo THandle
  Xqt sub
  Wait 100
  Exit Function

THandle:
  Print "IN Trap ", gi
Fend

Function sub
  For gi = 0 To 10
    Print gi
    Wait 0.5
  Next
Fend
```
Trap Statement (System status trigger)

Defines interrupts and what should happen when they occur.
With the Trap statement, you can jump to labels or call functions when the event occurs.
Trap statement has 2 types as below:
- 4 Traps that interrupts by the user defined input status
- 7 Traps that interrupts by the system status
Trap with system status triggers is explained here.

Syntax

```
Trap {Emergency | Error | Pause | SGOpen | SGClose | Abort | Finish } Xqt funcname
```

Parameters

- **Emergency**: In the emergency stop status, executes the specified function.
- **Error**: In the error status, executes the specified function.
- **Pause**: In the pause status, executes the specified function.
- **SGOpen**: When safeguard is open, executes the specified function.
- **SGClose**: When safeguard is closed, executes the specified function.
- **Abort**: All tasks except the background tasks stops (such as when a statement corresponding to the Abort All is executed or Pause button is pressed) by the user or system, executes the specified function.
- **Finish**: All tasks except the background tasks are completed, executes the specified function. It cannot be executed in the condition which executes the Trap Abort.
- **funcname**: Function of interrupt processing task for which Xqt is executed when the system status is completed. Functions with argument cannot be specified. However, three parameters can be specified if “Error” is specified for the parameter.

Note

Trap *** Call function of EPSON RC+4.x or before is replaced to Trap *** Xqt in EPSON RC+ 7.0.

Description

When the system status completes, the specified interrupt processing task is executed.
Even if you execute an interrupt processing task, the Trap settings cannot be cleared.
To clear the Trap setting, omit the `funcname` and execute the Trap statement.
Example: Trap Emergency clears Trap Emergency
After all normal tasks are completed and the controller is in the Ready status, all Trap settings are cleared.
You cannot execute more tasks using the Xqt from an interrupt processing.

**CAUTION**

- **Forced flag**
  You can turn On/Off the I/O outputs even in the Emergency Stop status, Safeguard Open status, Teach mode, or error status by specifying the Forced flag to the I/O output statement such as On and Off statements.
  DO NOT connect the external devices which can move machines such as actuators with the I/O outputs which specifies the Forced flag. It is extremely dangerous and it can lead the external devices to move in the Emergency Stop status, Safeguard Open status, Teach mode, or error status.
  I/O outputs which specifies the Forced flag is supposed to be connected with the external device such as LED as the status display which cannot move machines.
Trap Statement (System status trigger)

**If Emergency is specified**
When the Emergency Stop is activated, the specified function is executed in the NoEmgAbort task attribute.
The commands executable from the interrupt processing tasks can execute the NoEmgAbort task.
When the interrupt processing of Emergency Stop is completed, finish the task promptly. Otherwise, the controller cannot be in the Ready status. You cannot reset the Emergency Stop automatically by executing the Reset command from the interrupt processing task.
When the task executes I/O On/Off from the interrupt processing task, uncheck the [Outputs off during emergency stop] check box in the [Controller]-[Preferences] page. If this check box is checked, the execution order of turn Off by the controller and turn On using the task are not guaranteed.

**If Error is specified**
When the Error is activated, the specified function is executed in the NoEmgAbort task attribute.
The commands executable from the interrupt processing tasks can execute the NoEmgAbort task.
When the interrupt processing of Emergency Stop is completed, finish the task promptly. Otherwise, the controller cannot be in the Ready status.
The three omittable parameters (errNumber, robotNumber, jointNumber) can be specified to the user function. If you want to use these parameters, add three byval integer parameters to the trap function.
If a motion error occurs, errNumber, robotNumber, and jointNumber are set.
If an error other than the motion error occurs, '0' will be set to robotNumber, and jointNumber.

**If Pause is specified**
When the Pause is activated, the specified function is executed in the NoEmgAbort task attribute.

**If SGOpen is specified**
When the Safeguard is open, the specified function is executed in the NoEmgAbort task attribute.

**If SGClose is specified**
When the safeguard is closed and latched, the specified function is executed in the NoEmgAbort task attribute.
If you execute the Cont statement from the interrupt processing tasks, an error occurs.

**If Abort is specified**
All tasks except background tasks stop (such as when a statement corresponding to the Abort All is executed or Pause button is pressed) by the user or system, executes the specified function in the NoPause attribute.
When the interrupt processing of Pause is completed, finish the task promptly. Otherwise, the controller cannot be in the Ready status. Although a task executed with the Trap Abort has an error, the Trap Error processing task is not executed.
If the Shutdown or Restart statements are aborted, processing tasks of neither the Trap Abort nor Trap Finish is executed.

**If Finish is specified**
All tasks except the background tasks stops (such as when a statement corresponding to the Abort All is executed or Pause button is pressed) by the user or system, executes the specified function in the NoPause attribute. It cannot be executed in the condition which executes the Trap Abort processing task.
When the shutdown and interrupt processing are completed, finish the tasks promptly. Otherwise, the controller cannot be in the Ready status.

**See Also**
Era, Erl, Err, Ert, ErrMsg$, OnErr, Reset, Restart, SysErr, Xqt
Trap Statement Example

Function main

:  Trap Error Xqt suberr

:  Fend

Function suberr

  Print "Error =", Err
  On ErrorSwitch
  Fend

Function main

  Trap Error Xqt trapError

FEnd

Function trapError(errNum As Integer, robotNum As Integer, jointNum As Integer)
  Print “error number = “, errNum
  Print “robot number = “, robotNum
  Print “joint number = “, jointNum
  If Ert = 0 Then
    Print “system error”
  Else
    Print “task error”
    Print “function = “, Erf$(Ert)
    Print “line number = “, Erl(Ert)
  EndIf
FEnd
**Trim$ Function**

Returns a string equal to specified string without leading or trailing spaces.

**Syntax**

`Trim$(string)`

**Parameters**

- `string` String expression.

**Return Values**

Specified string with leading and trailing spaces removed.

**See Also**

`LTrim$`, `RTrim$`

**Trim$ Function Example**

```
str$ = "  data  
str$ = Trim$(str$)  ' str$ = "data"
```
TW Function

Returns the status of the Wait, WaitNet, and WaitSig commands.

Syntax
TW

Return Values
Returns False if Wait condition is satisfied within the time interval.
Returns True if the time interval has elapsed.

Description
The Timer Wait function TW returns the status of the preceding Wait condition with time interval with a False (Wait condition was satisfied) or a True (time interval has elapsed).

See Also
TMOut, Wait

TW Function Example

```
Wait Sw(0) = On, 5    ' Waits up to 5 seconds for input bit 0 On
If TW = True Then
    Print “Time Up”    ' Displays “Time UP” after 5 seconds
EndIf
```
UBound Function

Returns the largest available subscript for the indicated dimension of an array.

Syntax

UBound (arrayName [, dimension])

Parameters

arrayName

Name of the array variable; follows standard variable naming conventions.

dimension

Optional. Integer expression indicating which dimension's upper bound is returned. Use 1 for the first dimension, 2 for the second, and 3 for the third. If dimension is omitted, 1 is assumed.

See Also

Redim

UBound Function Example

Integer i, a(10)

For i=0 to UBound(a)
    a(i) = i
Next
UByte Statement

Declares variables of UByte type. (unsigned variable type, size: 2 bytes).

**Syntax**

```
UByte varName [((subscripts))] [, varName [((subscripts))]...]
```

**Parameters**

- `varName` Variable name which the user wants to declare as UByte type.
- `subscripts` Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows:
  
  `(ubound1, [ubound2], [ubound3])`

  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

<table>
<thead>
<tr>
<th>Type</th>
<th>Maximum Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local variable</td>
<td>2,000</td>
</tr>
<tr>
<td>Global Preserve variable</td>
<td>4,000</td>
</tr>
<tr>
<td>Global variable and module variable</td>
<td>100,000</td>
</tr>
</tbody>
</table>

**Description**

UByte is used to declare variables as UByte type. Variables of UByte type can contain values from 0 to 255. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

**See Also**

- Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Real, Short, String, UInt32, UInt64, UShort

**UByte Statement Example**

The following example shows a simple program that declares some variables as UByte type and assigns values to the variables.

The program monitors whether the top bit of “test_ok” is 1 or 0. The result will be displayed on the screen. (Since the value 15 is assigned to the variable, the bit with higher “test_ok” value is always set in this example.)

```
Function Test
    UByte A(10)      'Single dimension array of UByte type
    UByte B(10, 10)  'Two dimension array of UByte type
    UByte C(5, 5, 5) 'Three dimension array of UByte type
    UByte test_ok
    test_ok = 15
    Print "Initial Value of test_ok = ", test_ok
    test_ok = (test_ok And 8)
    If test_ok <> 8 Then
        Print "test_ok high bit is ON"
    Else
        Print "test_ok high bit is OFF"
    EndIf
EndFunction
```
UCase$ Function

Returns a string that has been converted to uppercase.

Syntax

\[
\text{UCase$ (string)}
\]

Parameters

\[
\text{string} \quad \text{String expression.}
\]

Return Values

The converted uppercase string.

See Also

LCase$, LTrim$, Trim$, RTrim$

UCase$ Function Example

\[
\begin{align*}
\text{str$} & = "Data" \\
\text{str$} & = \text{UCase$ (str$)} \quad \text{str$} = "DATA"
\end{align*}
\]
UInt32 Statement

Declares variables of UInt32 type. (unsigned 4-byte integer variable).

Syntax

\[ \text{UInt32 } \text{varName} \left[ \text{[(subscripts)]} \right] \left[ , \text{varName} \left[ \text{[(subscripts)]} \right] \right] \]

Parameters

- \text{varName} : Variable name which the user wants to declare.
- \text{subscripts} : Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows
  \[ (\text{ubound1}, [\text{ubound2}], [\text{ubound3}]) \]
  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.
  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.
  When specifying the upper bound value, make sure the number of total elements is within the range shown below:
  - Local variable 2,000
  - Global Preserve variable 4,000
  - Global variable and module variable 100,000

Description

UInt32 is used to declare variables as integer type. Variables of integer type can contain values from 0 to 4294967295. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt64, UShort

UInt32 Statement Example

The following example shows a simple program that declares some variables as integer type using UInt32.

```plaintext
Function uint32test
    UInt32 A(10)       'Single dimension array of UInt32 type
    UInt32 B(10, 10)   'Two dimension array of UInt32 type
    UInt32 C(5, 5, 5)  'Three dimension array of UInt32 type
    UInt32 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
Next i
Fend
```
UInt64 Statement

Declares variables of Uint64 type. (unsigned 8-byte integer variable).

Syntax

\[
\text{UInt64 } \text{varName} [(\text{subscripts})] [, \text{varName} [(\text{subscripts})]...]
\]

Parameters

- **varName**
  - Variable name which the user wants to declare.

- **subscripts**
  - Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows.

\[
(\text{ubound1}, [\text{ubound2}], [\text{ubound3}])
\]

- ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

- The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

- When specifying the upper bound value, make sure the number of total elements is within the range shown below:

\[
\begin{align*}
\text{Local variable} & : 2,000 \\
\text{Global Preserve variable} & : 4,000 \\
\text{Global variable and module variable} & : 100,000
\end{align*}
\]

Description

UInt64 is used to declare variables as integer type. Variables of integer type can contain values from 0 to 18446744073709551615. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UShort

UInt64 Statement Example

The following example shows a simple program that declares some variables as integer type using UInt64.

```
Function uint64test
    UInt64 A(10)         ' Single dimension array of UInt64 type
    UInt64 B(10, 10)     ' Two dimension array of UInt64 type
    UInt64 C(5, 5, 5)    ' Three dimension array of UInt64 type
    UInt64 var1, arrayvar(10)
    Integer i
    Print "Please enter an Integer Number"
    Input var1
    Print "The Integer variable var1 = ", var1
    For i = 1 To 5
        Print "Please enter an Integer Number"
        Input arrayvar(i)
        Print "Value Entered was ", arrayvar(i)
    Next i
    Fend
```

UOpen Statement

Opens a file for read / write access.

**Syntax**

```
UOpen fileName As #fileNumber
.
Close #fileNumber
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fileName</code></td>
<td>String expression that specifies path and file name. If path is omitted, the file in the current directory is specified. See ChDisk for the details.</td>
</tr>
<tr>
<td><code>fileNumber</code></td>
<td>Integer expression representing values from 30 to 63.</td>
</tr>
</tbody>
</table>

**Description**

Opens the specified file by the specified file number. This statement is used for writing and loading data in the specified file.

**Note**

A network path is available.

If the specified file does not exist on disk, the file will be created and the data will be written into it. If the specified file already exists on disk, the data will be written and read starting from the beginning of the existing data.

The read/write position (pointer) of the file can be changed using the Seek command. When switching between read and write access, you must use Seek to reposition the file pointer.

`fileNumber` identifies the file while it is open and cannot be used to refer to a different file until the current file is closed. `fileNumber` is used by other file operations such as Print#, Read, Write, Seek, and Close.

Close closes the file and releases the file number.

It is recommended that you use the FreeFile function to obtain the file number so that more than one task are not using the same number.

**See Also**

Close, Print #, Input#, AOpen, BOpen, ROpen, WOpen, FreeFile, Seek

**UOpen Statement Example**

```
Integer fileNum, i, j

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
UOpen "TEST.DAT" As #fileNum
Seek #fileNum, 10
Input #fileNum, j
Print "data = ", j
Close #fileNum
```
UpdateDB Statement

Updates the data in the table which is retrieved in the opened data base.

Syntax

```
UpdateDB #DBNumber, item, value
```

Parameters

- **DBNumber**
  - Integer expression (501 to 508) representing the data base number specified by OpenDB.
- **item**
  - Item name of the table to update.
- **value**
  - Value to be updated.

Description

Updates the data in the table which is retrieved in the opened data base with the specified value. Before updating the data, it is required to issue SelectDB and select the record to be updated.

Note

- Connection of PC with installed RC+ is required.

See Also

OpenDB, CloseDB, SelectDB, DeleteDB

UpdateDB Statement Example

Example using SQL database

Following is an example to register the data to the table “Employees” in the sample database “Northwind” of SQL server 2000, and update the items in the registered data.

```plaintext
Integer count, i, eid
String Lastname$, Firstname$, Title$

OpenDB #501, SQL, "(LOCAL)", "Northwind"
count = SelectDB(#501, "Employees", "TitleOfCourtesy = 'Mr.'")
Print #501, "Epson", "Taro", "Engineer", "Mr."
count = SelectDB(#501, "Employees", "LastName = 'Epson' and FirstName = 'Taro'")
Input #501, eid, Lastname$, Firstname$, Title$
Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
UpdateDB #501, "Title", "Chief Engineer"
count = SelectDB(#501, "Employees", "LastName = 'Epson' and FirstName = 'Taro'")
Input #501, eid, Lastname$, Firstname$, Title$
Print eid, ",", Lastname$, ",", Firstname$, ",", Title$
CloseDB #501
```
UShort Statement

Declares variables of UShort type. (unsigned 2-byte integer variable).

Syntax

```
UShort varName [[(subscripts)]] [, varName [(subscripts)]]...
```

Parameters

- **varName**: Variable name which the user wants to declare.
- **subscripts**: Optional. Dimensions of an array variable; up to 3 dimensions may be declared. The subscripts syntax is as follows
  
  `(ubound1, [ubound2], [ubound3])`

  ubound1, ubound2, ubound3 each specify the maximum upper bound for the associated dimension.

  The elements in each dimension of an array are numbered from 0 and the available number of array elements is the upper bound value + 1.

  When specifying the upper bound value, make sure the number of total elements is within the range shown below:

  - Local variable: 2,000
  - Global Preserve variable: 4,000
  - Global variable and module variable: 100,000

Description

UShort is used to declare variables as integer type. Integer variables can contain values from 0 to 65535. Local variables should be declared at the top of a function. Global and module variables must be declared outside of functions.

See Also

- Boolean, Byte, Double, Global, Int32, Int64, Integer, Long, Real, Short, String, UByte, UInt32, UInt64

UShort Statement Example

The following example shows a simple program that declares some variables as integer type using UShort.

```
Function ushorttest
  UShort A(10)    'Single dimension array of UShort type
  UShort B(10, 10) 'Two dimension array of UShort type
  UShort C(5, 5, 5) 'Three dimension array of UShort type
  UShort var1, arrayvar(10)
  Integer i
  Print "Please enter an Integer Number"
  Input var1
  Print "The Integer variable var1 = ", var1
  For i = 1 To 5
    Print "Please enter an Integer Number"
    Input arrayvar(i)
    Print "Value Entered was ", arrayvar(i)
  Next i
  Fend
```
Val Function

Converts a character string that consists of numbers into their numerical value and returns that value.

Syntax

Val(string)

Parameters

string String expression which contains only numeric characters. The string may also contain a prefix: &H (hexadecimal), &O (octal), or &B (binary).

Return Values

Returns an integer or floating point result depending upon the input string. If the input string has a decimal point character than the number is converted into a floating point number. Otherwise the return value is an integer.

Description

Val converts a character string of numbers into a numeric value. The result may be an integer or floating point number. If the string passed to the Val instruction contains a decimal point then the return value will be a floating point number. Otherwise it will be an integer.

See Also

Abs, Asc, Chr$, Int, Left$, Len, Mid$, Mod, Right$, Sgn, Space$, Str$

Val Function Example

The example shown below shows a program which converts several different strings to numbers and then prints them to the screen.

Function ValDemo

String realstr$, intstr$
Real realvar, realsqr
Integer intvar, intsqr

realstr$ = "2.5"
realvar = Val(realstr$)
realsqr = realvar * realvar
Print "The value of ", realstr$, " squared is: ", realsqr

intstr$ = "25"
intvar = Val(intstr$)
intsqr = intvar * intvar
Print "The value of ", intstr$, " squared is: ", intsqr
Fend

Here's another example from Command window.

> Print Val("25.999")
25.999
>
**VSD Statement**

Sets the variable speed CP motion function for SCARA robot.

**Syntax**

\[
\text{VSD \{ ON | Off \}}
\]

**Parameters**

On | Off  
---|---
On: Enables the variable speed CP motion function of SCARA robot.  
Off: Disables the variable speed CP motion function of SCARA robot.

**Description**

VSD is available for following commands.  
Move, Arc, Arc3
This command is available only for SCARA robots.  
For other than SCARA robots, use AvoidSingularity SING_VSD.

The variable speed CP motion function prevents the acceleration error and overspeed error from occurring when SCARA robot is executing CP motion. This function automatically controls the joint speed while keeping the trajectory. If the joint speed is controlled, the tool center point speed specified by SpeedS will not be kept. However, the original speed setting will be returned when the joint speed gets below the limit.  
If constant velocity is prioritized, set AccelS, DecelS, and SpeedS smaller and eliminate the error occurrence.

If the acceleration and overspeed errors occur even when the VSD statement is used, set AccelS, DecelS, and SpeedS smaller.

If the VSD parameter is changed, the current setting is effective until the next controller startup.  
VSD is set off when the startup of the controller.

**See Also**

VSD Function

**VSD Statement Example**

\[
\text{VSD On} \quad \text{Enable the variable speed CP motion and execute the motion}  
\quad \text{Move P1}  
\quad \text{Move P2}  
\quad \text{VSD Off}
\]
VSD Function

Returns the setting of the variable speed CP motion function for SCARA robot.

Syntax

VSD

Return Values

On = Enables the variable speed CP motion function
Off = Disables the variable speed CP motion function

See Also

VSD

VSD Function Example

If VSD = Off Then
    Print "Variable Speed Drive is off"
EndIf
VxCalib Statement

This command is only for use with external vision systems and cannot be used with Vision Guide.
Creates calibration data for an external vision system.

Syntax
(1) \texttt{VxCalib CalNo}
(2) \texttt{VxCalib CalNo, CamOrient, P(pixel\_st : pixel\_ed), P(robot\_st : robot\_ed) [, TwoRefPoints]}
(3) \texttt{VxCalib CalNo, CamOrient, P(pixel\_st : pixel\_ed), P(robot\_st : robot\_ed), P(ref0) [, P(ref180)]}

Parameters
\begin{itemize}
  \item \textit{CalNo} Integer expression that specifies the calibration data number. The range is from 0 to 15; up to 16 calibrations may be defined.
  \item \textit{CamOrient} Integer expression that specifies the camera mounting direction using the following values:
    \begin{itemize}
      \item 1 to 3: Available only for syntax (2).
      \item 4 to 7: Available only for syntax (3).
      \item 1: Standalone
      \item 2: Fixed downward
      \item 3: Fixed upward
      \item 4: Mobile on Joint #2
      \item 5: Mobile on Joint #4
      \item 6: Mobile on Joint #5
      \item 7: Mobile on Joint #6
    \end{itemize}
  \item \textit{P(pixel\_st : pixel\_ed)} Specifies the Pixel coordinates (X, Y only) using the continuous point data.
  \item \textit{P(robot\_st : robot\_ed)} Specifies the robot coordinates using the continuous point data.
    The point data varies with mounting directions of the camera specified by CamOrient.
    \begin{itemize}
      \item If CamOrient = 1 to 3:
        The robot coordinates must be set to the current TOOL and ARM values.
      \item If CamOrient = 4 to 7:
        The robot coordinates must be set as TOOL: 0, ARM: 0.
    \end{itemize}
  \item \textit{TwoRefPoints} Available for syntax (1).
    True, when using two measuring points. False, when using one measuring point.
    Specifying two measuring points makes the calibration more accurate.
    Optional.
    Default: False
  \item \textit{P(ref0)} Available for syntax (3).
    Specifies the robot coordinates of the reference point using the point data.
  \item \textit{P(ref180)} Available for syntax (3).
    Specifies the robot coordinates of the second reference point using the point data.
    Specifying two reference points makes the calibration more accurate.
    Optional.
\end{itemize}

Description
The VxCalib command calculates the vision calibration data for the specified calibration number using the specified camera orientation, pixel coordinates, robot coordinates, and reference points (Mobile camera only) given by the parameter.

When you specify only \textit{CalNo}, the point data and other settings you defined are displayed (only from the Command Window).
The following figure shows the coordinates system of the pixel coordinates. (Units: pixel)

For the pixel coordinates and robot coordinates, set the top left position of the window as Point 1 and set the bottom right position as Point 9 according to the order in the table below.

It is classified into the four categories by the parameter CamOrient and TwoRefPoints.

1) CamOrient = 1 to 3 (Standalone, Fixed Downward, Fixed Upward), TwoRefPoints = False

<table>
<thead>
<tr>
<th>Data order</th>
<th>Position</th>
<th>Pixel coordinates</th>
<th>Robot coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Top left</td>
<td>Detection coordinates 1</td>
<td>Measuring point coordinates 1</td>
</tr>
<tr>
<td>2</td>
<td>Top center</td>
<td>Detection coordinates 2</td>
<td>Measuring point coordinates 2</td>
</tr>
<tr>
<td>3</td>
<td>Top right</td>
<td>Detection coordinates 3</td>
<td>Measuring point coordinates 3</td>
</tr>
<tr>
<td>4</td>
<td>Center right</td>
<td>Detection coordinates 4</td>
<td>Measuring point coordinates 4</td>
</tr>
<tr>
<td>5</td>
<td>Center</td>
<td>Detection coordinates 5</td>
<td>Measuring point coordinates 5</td>
</tr>
<tr>
<td>6</td>
<td>Center left</td>
<td>Detection coordinates 6</td>
<td>Measuring point coordinates 6</td>
</tr>
<tr>
<td>7</td>
<td>Bottom left</td>
<td>Detection coordinates 7</td>
<td>Measuring point coordinates 7</td>
</tr>
<tr>
<td>8</td>
<td>Bottom center</td>
<td>Detection coordinates 8</td>
<td>Measuring point coordinates 8</td>
</tr>
<tr>
<td>9</td>
<td>Bottom right</td>
<td>Detection coordinates 9</td>
<td>Measuring point coordinates 9</td>
</tr>
</tbody>
</table>

2) CamOrient = 2 (Fixed Downward), TwoRefPoints = True

Note: When the tool is exactly defined, TwoRefPoints is not necessary and should be set to False.

By setting TwoRefPoints to True, two measuring points are used for each calibration position, which makes the calibration more accurate. 18 robot points with U axis: 0 degree / 180 degrees are required.

After setting 1 to 9 measuring points coordinates, turn the U axis by 180 degrees and set the measuring point coordinates 10 to 18 where the hand (such as the rod) is positioned at the calibration target position.

<table>
<thead>
<tr>
<th>Data order</th>
<th>Position</th>
<th>Pixel coordinates</th>
<th>Robot coordinates</th>
<th>U axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Top left</td>
<td>- - -</td>
<td>Measuring point coordinates 10</td>
<td>0 degree</td>
</tr>
<tr>
<td>2</td>
<td>Top center</td>
<td>- - -</td>
<td>Measuring point coordinates 11</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Top right</td>
<td>- - -</td>
<td>Measuring point coordinates 12</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Center right</td>
<td>- - -</td>
<td>Measuring point coordinates 13</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Center</td>
<td>- - -</td>
<td>Measuring point coordinates 14</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Center left</td>
<td>- - -</td>
<td>Measuring point coordinates 15</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Bottom left</td>
<td>- - -</td>
<td>Measuring point coordinates 16</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Bottom center</td>
<td>- - -</td>
<td>Measuring point coordinates 17</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Bottom right</td>
<td>- - -</td>
<td>Measuring point coordinates 18</td>
<td></td>
</tr>
</tbody>
</table>
3) CamOrient = 3 (Fixed Upward), TwoRefPoints = True
Note: When the tool is exactly defined, TwoRefPoints is not necessary and should be set to False.

By setting TwoRefPoints to True, two detection points are used, which makes the calibration more accurate. For only the pixel coordinates, 18 points of U axis: 0 degree / 180 degrees are required. After setting 1 to 9 detection coordinates at the each measuring point coordinates at 0 degrees, set the detection coordinates for points 10 to 18 at 180 degrees.

<table>
<thead>
<tr>
<th>Data order</th>
<th>Position</th>
<th>Pixel coordinates</th>
<th>Robot coordinates</th>
<th>U axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Top left</td>
<td>Detection coordinates 1</td>
<td>Measuring point coordinates 1</td>
<td>0 degree</td>
</tr>
<tr>
<td>2</td>
<td>Top center</td>
<td>Detection coordinates 2</td>
<td>Measuring point coordinates 2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Top right</td>
<td>Detection coordinates 3</td>
<td>Measuring point coordinates 3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Center right</td>
<td>Detection coordinates 4</td>
<td>Measuring point coordinates 4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Center</td>
<td>Detection coordinates 5</td>
<td>Measuring point coordinates 5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Center left</td>
<td>Detection coordinates 6</td>
<td>Measuring point coordinates 6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Bottom left</td>
<td>Detection coordinates 7</td>
<td>Measuring point coordinates 7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Bottom center</td>
<td>Detection coordinates 8</td>
<td>Measuring point coordinates 8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Bottom right</td>
<td>Detection coordinates 9</td>
<td>Measuring point coordinates 9</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data order</th>
<th>Position</th>
<th>Pixel coordinates</th>
<th>Robot coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Top left</td>
<td>Detection coordinates 10</td>
<td>- - -</td>
</tr>
<tr>
<td>11</td>
<td>Top center</td>
<td>Detection coordinates 11</td>
<td>- - -</td>
</tr>
<tr>
<td>12</td>
<td>Top right</td>
<td>Detection coordinates 12</td>
<td>- - -</td>
</tr>
<tr>
<td>13</td>
<td>Center right</td>
<td>Detection coordinates 13</td>
<td>- - -</td>
</tr>
<tr>
<td>14</td>
<td>Center</td>
<td>Detection coordinates 14</td>
<td>- - -</td>
</tr>
<tr>
<td>15</td>
<td>Center left</td>
<td>Detection coordinates 15</td>
<td>- - -</td>
</tr>
<tr>
<td>16</td>
<td>Bottom left</td>
<td>Detection coordinates 16</td>
<td>- - -</td>
</tr>
<tr>
<td>17</td>
<td>Bottom center</td>
<td>Detection coordinates 17</td>
<td>- - -</td>
</tr>
<tr>
<td>18</td>
<td>Bottom right</td>
<td>Detection coordinates 18</td>
<td>- - -</td>
</tr>
</tbody>
</table>

4) CamOrient = 4 to 7

<table>
<thead>
<tr>
<th>Data order</th>
<th>Position</th>
<th>Pixel coordinates</th>
<th>Robot coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Top left</td>
<td>Detection coordinates 1</td>
<td>Measuring point coordinates 1</td>
</tr>
<tr>
<td>2</td>
<td>Top center</td>
<td>Detection coordinates 2</td>
<td>Measuring point coordinates 2</td>
</tr>
<tr>
<td>3</td>
<td>Top right</td>
<td>Detection coordinates 3</td>
<td>Measuring point coordinates 3</td>
</tr>
<tr>
<td>4</td>
<td>Center right</td>
<td>Detection coordinates 4</td>
<td>Measuring point coordinates 4</td>
</tr>
<tr>
<td>5</td>
<td>Center</td>
<td>Detection coordinates 5</td>
<td>Measuring point coordinates 5</td>
</tr>
<tr>
<td>6</td>
<td>Center left</td>
<td>Detection coordinates 6</td>
<td>Measuring point coordinates 6</td>
</tr>
<tr>
<td>7</td>
<td>Bottom left</td>
<td>Detection coordinates 7</td>
<td>Measuring point coordinates 7</td>
</tr>
<tr>
<td>8</td>
<td>Bottom center</td>
<td>Detection coordinates 8</td>
<td>Measuring point coordinates 8</td>
</tr>
<tr>
<td>9</td>
<td>Bottom right</td>
<td>Detection coordinates 9</td>
<td>Measuring point coordinates 9</td>
</tr>
</tbody>
</table>

Note
In addition to the tables above, specify the robot coordinates of the reference points. Using the two reference points makes the calibration more accurate. In this case, it needs two points of U axis: 0 degree / 180 degrees.

After setting the first reference points coordinates, turn the U axis by 180 degrees and set the second reference points coordinates where the hand (such as the rod) is positioned at the calibration target position. When the tool is exactly defined, the two reference points are not necessary.

See Also
VxTrans Function, VxCalInfo Function, VxCalDelete, VxCalSave, VxCalLoad
VxCalib Statement

VxCalib Statement Example

Function MobileJ2

    Integer i
    Double d(8)

Robot 1
LoadPoints "MobileJ2.pts"

VxCalib 0, 4, P(21:29), P(1:9), P(0)

If (VxCalInfo(0, 1) = True) Then
    For i = 0 To 7
        d(i) = VxCalInfo(0, i + 2)
    Next i
    Print "Calibration result:"
    Print d(0), d(1), d(2), d(3), d(4), d(5), d(6), d(7)
    P52 = VxTrans(0, P51, P50)
    Print "Coordinates conversion result:"
    Print P52
    SavePoints "MobileJ2.pts"
    VxCalSave "MobileJ2.caa"
Else
    Print "Calibration failed"
EndIf

Fend
This command is only for use with external vision systems and cannot be used with Vision Guide. Deletes the calibration data for an external vision system calibration.

Syntax

VxCalDelete CalNo

Parameters

CalNo Integer expression that specifies the calibration data number. The range is from 0 to 15; up to 16 calibrations may be defined.

Description

Deletes the calibration data defined by the specified calibration number.

See Also

VxCalib, VxTrans Function, VxCallInfo Function, VxCalSave, VxCalLoad

VxCalDelete Statement Example

VxCalDelete "MobileJ2.caa"
VxCalLoad Statement

This command is only for use with external vision systems and cannot be used with Vision Guide. Loads the calibration data for an external vision system calibration from a file.

Syntax
VxCalLoad FileName

Parameters
FileName Specifies the file name from which the calibration data is loaded using a string expression. The file extension is "caa". If omitted, "caa" is automatically added. For extensions other than "caa", they are automatically changed to "caa".

Description
Loads the calibration data from the specified file in the current project.

See Also
VxCalib, VxTrans Function, VxCalInfo Function, VxCalDelete, VxCalSave

VxCalLoad Statement Example

VxCalLoad "MobileJ2.caa"
VxCalInfo Function

This command is only for use with external vision systems and cannot be used with Vision Guide.
Returns the calibration completion status and the calibration data.

Syntax
VxCalInfo (CalNo, CalData)

Parameters
CalNo Integer expression that specifies the calibration data number. The range is from 0 to 15; up to 16 calibrations may be defined.
CalData Specifies the calibration data type to acquire using the integer values in the table below.

<table>
<thead>
<tr>
<th>CalData</th>
<th>Calibration Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CalComplete</td>
</tr>
<tr>
<td>2</td>
<td>X Avg Error [mm]</td>
</tr>
<tr>
<td>3</td>
<td>X Max error [mm]</td>
</tr>
<tr>
<td>4</td>
<td>X mm per pixel [mm]</td>
</tr>
<tr>
<td>5</td>
<td>X tilt</td>
</tr>
<tr>
<td>6</td>
<td>Y Avg error [mm]</td>
</tr>
<tr>
<td>7</td>
<td>Y Max error [mm]</td>
</tr>
<tr>
<td>8</td>
<td>Y mm per pixel [mm]</td>
</tr>
<tr>
<td>9</td>
<td>Y tilt</td>
</tr>
</tbody>
</table>

Return Values
Returns the specified calibration data. For CalData = 1, the data type is Boolean. For all other data, the data type is Double.

Description
You can check which calibration has defined calibration data. Also, you can retrieve the calibration data values.

See Also
VxCalib, VxTrans Function, VxCalDelete, VxCalSave, VxCalLoad

VxCalInfo Function Example

Print VxCalInfo (0, 1)
VxCalSave Statement

This command is only for use with external vision systems and cannot be used with Vision Guide. Saves the calibration data for an external vision system calibration to a file.

Syntax
VxCalSave FileName

Parameters
FileName Specifies the file name from which the calibration data is loaded using a string expression. The extension is “.caa”. If omitted, “.caa” is automatically added. For extensions other than “.caa”, they are automatically changed to “.caa”.

Description
Saves the calibration data with the specified file name. The file is saved in the current project. If the file name is already existed, the calibration data is overwritten.

See Also
VxCalib, VxTrans Function, VxCalInfo Function, VxCalDelete, VxCalLoad

VxCalSave Statement Example

VxCalSave "MobileJ2.caa"
VxTrans Function

This command is only for use with external vision systems and cannot be used with Vision Guide.
Converting pixel coordinates to robot coordinates and returns the converted point data.

Syntax

VxTrans (CalNo, P(pixel) [, P(camRobot)]) As Point

Parameters

CalNo  
Integer expression that specifies the calibration data number.
The range is from 0 to 15; up to 16 calibrations may be defined.

P(pixel)  
Specifies the vision pixel coordinates (X, Y, U only) using point data.

P(camRobot)  
Optional. For a mobile camera, this is the position where the robot was located when the
image was acquired. If not specified, then the current robot position is used.
The point should be in TOOL: 0 and ARM: 0.

Return Values

Returns the calculated robot coordinates using the point data.

Description

This command converts pixel coordinates to robot coordinates using the calibration data of the specified
 calibration number.

When using a mobile camera, specify P(camRobot) if the robot has been moved from the position where
the image was acquired. Ensure that P(camRobot) is in TOOL: 0 and ARM: 0. The Joint #4 and Joint #6
angles of the set robot coordinates are used for the calculation.

See Also

VxCali, VxCaliInfo Function, VxCaliDelete, VxCaliSave, VxCaliLoad

VxTrans Function Example

P52 = VxTrans(0, P51, P50)
Wait Statement

Causes the program to Wait for a specified amount of time or until the specified input condition (using MemSw or Sw) is met. (Oport may also be used in the place of Sw to check hardware outputs.) Also waits for the values of global variables to change.

Syntax

(1) \textbf{Wait} \textit{time}
(2) \textbf{Wait} \textit{inputcondition}
(3) \textbf{Wait} \textit{inputcondition, time}

Parameters

\textit{time} Real expression between 0 and 2,147,483 which represents the amount of time to wait when using the Wait instruction to wait based on time. Time is specified in seconds. The smallest increment is 0.01 seconds.

\textit{inputcondition} The following syntax can be used to specify the input condition:

\begin{itemize}
  \item \textbf{[Event]}  Comparative operator ( =, <>, >=, >, <, <= ) \textbf{[Integer expression]}
\end{itemize}

The following functions and variables can be used in the \textit{Event}.

\textbf{Functions} :  Sw, In, InW, Oport, Out, OutW, MemSw, MemIn, MemInW, Ctr, GetRobotInsideBox, GetRobotInsidePlane, MCalComplete, Motor, LOF, ErrorOn, SafetyOn, EstopOn, TeachOn, Cnv_QueLen, WindowsStatus, AtHome, LatchState, WorkQue_Len, PauseOn, AIO_In, AIO_InW, AIO_Out, AIO_OutW

\textbf{Operators} :  Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort global preserve variables, global variables, module variables

In addition, using the following operators you can specify multiple input conditions.

\textbf{Operator} :  And, Or, Xor, Mask

Description

(1) \textbf{Wait with Time Interval}

When used as a timer, the Wait instruction causes the program to pause for the amount of time specified and then continues program execution.

(2) \textbf{Wait for Event Conditions without Time Interval}

When used as a conditional Wait interlock, the Wait instruction causes the program to wait until specified conditions are satisfied. If after TMOOut time interval has elapsed and the Wait conditions have not yet been satisfied, an error occurs. The user can check multiple conditions with a single Wait instruction by using the And, Mask, Or, or Xor instructions. (Please review the example section for Wait.)

(3) \textbf{Wait with Event Condition and Time Interval}

Specifies Wait condition and time interval. After either Wait condition is satisfied, or the time interval has elapsed, program control transfers to the next command. Use Tw to verify if the Wait condition was satisfied or if the time interval elapsed.
### Notes

**Specifying a Timeout for Use with Wait**

When the Wait instruction is used without a time interval, a timeout can be specified which sets a time limit to wait for the specified condition. This timeout is set through using the TMOOut instruction. Please refer to this instruction for more information. (The default setting for TMOOut is “0” which means no timeout.)

**Waiting for variable with Wait**

- Available variables are Integer type (Byte, Int32, Integer, Long, Short, UByte, UInt32, UShort)
- Array variables are not available
- Local variables are not available
- If variables value cannot satisfy the event condition for more than 0.01 seconds, the change in variables may not be retrieved.
- Up to 64 can wait for variables in one system (including ones used in the event condition expressions such as Till). If it is over 64, an error occurs during the project build.
- If you specify Byref to a waiting variable on any function call, an error occurs.
- When a variable is included in the right side member of the event condition expression, the value is calculated when setting the Trap condition. We recommend not using variables in an integer expression to avoid making unintended conditions.

**When Using PC COM port (1001 to 1008)**

- You cannot use Lof Function for Wait instruction.

**When the program is paused while Wait is executing**

The Wait instruction does not stop even when the program is paused while the Wait instruction is executing. The Wait instruction ends when an event condition is satisfied or the specified time has passed. If the time is set by the Wait parameter, the passed time is reset and the program waits for the specified time when the program is restarted by selecting Run Window Continue.

### See Also

AtHome, Cnv_QueLen, Ctr, ErrorOn, EstopOn, GetRobotInsideBox, GetRobotInsidePlane, In, InW, LatchState, LOF, Mask, MCalComplete, MemIn, MemInW, MemSw, Motor, Oport, Out, OutW, PauseOn, SaftyOn, Sw, TeachOn, TMOut, WindowsStatus, Tw, WorkQue_Len

### Wait Statement Example

The example shown below shows 2 tasks each with the ability to initiate motion instructions. However, a locking mechanism is used between the 2 tasks to ensure that each task gains control of the robot motion instructions only after the other task is finished using them. This allows 2 tasks to execute motion statements as required and in an orderly predictable fashion. MemSw is used in combination with the Wait instruction to wait until the memory I/O #1 is the proper value before it is safe to move again.

```plaintext
Function main
    Integer I
    MemOff 1
    Xqt !2, task2
    For i = 1 to 100
        Wait MemSw(1) = Off
        Go P(i)
        MemOn 1
    Next I
Fend

Function task2
    Integer i
    For i = 101 to 200
        Wait MemSw(1) = On
        Go P(i)
        MemOff 1
    Next i
```

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
Fend

' Waits until input 0 turns on
Wait Sw(0) = On

' Waits 60.5 secs and then continue execution
Wait 60.5

' Waits until input 0 is off and input 1 is on
Wait Sw(0) = Off And Sw(1) = On

' Waits until memory bit 0 is on or memory bit 1 is on
Wait MemSw(0) = On Or MemSw(1) = On

' Waits one second, then turn output 1 on
Wait 1; On 1

' Waits for the lower 3 bits of input port 0 to equal 1
Wait In(0) Mask 7 = 1

' Waits until the global Integer type variable giCounter is over 10
Wait giCounter > 10

' Waits ten seconds, until the global Long type variable glCheck is 30000
Wait glCheck = 30000, 10
WaitNet Statement

Wait for TCP/IP port connection to be established.

Syntax

```
WaitNet #portNumber [, timeOut]
```

Parameters

- **portNumber**: Integer expression for TCP/IP port number to connect. Range is 201 to 216
- **timeOut**: Optional. Maximum time to wait for connection.

See Also

OpenNet, CloseNet

WaitNet Statement Example

For this example, two controllers have their TCP/IP settings configured as follows:

**Controller #1**:
Port: #201
Host Name: 192.168.0.2
TCP/IP Port: 1000

```
Function tcpip
  OpenNet #201 As Server
  WaitNet #201
  Print #201, "Data from host 1"
Fend
```

**Controller #2**:
Port: #201
Host Name: 192.168.0.1
TCP/IP Port: 1000

```
Function tcpip
  String data$
  OpenNet #201 As Client
  WaitNet #201
  Input #201, data$
  Print "received ", data$, " from host 1"
  Fend
```
Waits for robot to decelerate and stop at position before executing the next statement while path motion is active.

**Syntax**

```
WaitPos
```

**Description**

Normally, when path motion is active (CP On or CP parameter specified), the motion command starts the next statement as deceleration starts.

Use the `WaitPos` command right before the motion to complete the deceleration motion and go on to the next motion.

**See Also**

Wait, WaitSig, CP

**WaitPos Statement Example**

```
Off 1
CP On
Move P1
Move P2
WaitPos  ' waits for robot to decelerate
On 1
CP Off
```
**WaitSig Statement**

Waits for a signal from another task.

**Syntax**

\[
\text{WaitSig} \signalNumber [, \timeOut]
\]

**Parameters**

- \signalNumber\n  Integer expression representing signal number to receive. Range is from 0 to 63.
- \timeOut\n  Optional. Real expression representing the maximum time to wait.

**Description**

Use `WaitSig` to wait for a signal from another task. The signal will only be received after `WaitSig` has started. Previous signals are ignored.

**See Also**

`Wait`, `WaitPos`, `Signal`

**WaitSig Statement Example**

```plaintext
Function Main
  Xqt SubTask
  Wait 1
  Signal 1
  .
  Fend

Function SubTask
  WaitSig 1
  Print "signal received"
  .
  Fend
```
Weight Statement

Specifies or displays the weight setting for the robot arm.

Syntax

\[
\text{Weight } \text{payloadWeight} [, \text{distance} | \text{S} | \text{T} ]
\]

Parameters

- **payloadWeight**: The weight of the end effector to be carried in Kg unit.
- **distance**: The distance from the rotational center of the second arm to the center of the gravity of the end effector in mm unit. Valid only for SCARA robots (including RS series).
- **S**: Load weight against the additional S axis in kg to 2 decimal places.
- **T**: Load weight against the additional T axis in kg to 2 decimal places.

Return Values

Displays the current Weight settings when parameters are omitted.

Description

Specifies parameters for calculating Point to Point motion maximum acceleration. The Weight instruction specifies the weight of the end effector and the parts to be carried.

The Arm length (distance) specification is necessary only for SCARA robots (including RS series). It is the distance from the second arm rotation joint centerline to the hand/work piece combined center of gravity.

If the robot has the additional axis, the loads on the additional axis must be set with the S, T parameters.

If the equivalent value work piece weight calculated from specified parameters exceeds the maximum allowable payload, an error occurs.

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Potential Errors

Weight Exceeds Maximum

When the equivalent load weight calculated from the value entered exceeds the maximum load weight, an error will occur.

Potential Damage to the Manipulator Arm

Take note that specifying a Weight hand weight significantly less than the actual work piece weight can result in excessive acceleration and deceleration. These, in turn, may cause severe damage to the manipulator.

Note

Weight Values Are Not Changed by Turning Main Power Off

The Weight values are not changed by turning power off.

See Also

Accel, Inertia
Weight Statement Example

This Weight instruction on the Command window displays the current setting.

```
> weight
2.000, 200.000
>
```

Sets the hand weight (3 kg) with the Weight statement

```
Weight 3.0
```

Sets the load weight on the additional S axis (30 kg) with the Weight statement

```
Weight 30.0, S
```
Weight Function

Returns a Weight parameter.

Syntax
Weight(paramNumber)

Parameters
paramNumber Integer expression containing one of the values below:
1: Payload weight
2: Arm length
3: Load on the additional S axis
4: Load on the additional T axis

Return Values
Real number containing the parameter value.

See Also
Inertia, Weight

Weight Function Example

Print "The current Weight parameters are: ", Weight(1)
Where Statement

Displays current robot position data.

Syntax
Where [localNumber]

Parameters
localNumber Optional. Specifies the local coordinate system number. Default is Local 0.

See Also
Joint, PList, Pulse

Where Statement Example

The display type can be different depending on the robot type and existence of additional axes. The following example is for Scara robot without the additional axis.

>where
WORLD: X: 350.000 mm Y: 0.000 mm Z: 0.000 mm U: 0.000 deg V: 0.000 deg W: 0.000 deg
JOINT: 1: 0.000 deg 2: 0.000 deg 3: 0.000 mm 4: 0.000 deg
PULSE: 1: 0 pls 2: 0 pls 3: 0 pls 4: 0 pls

>local 1, 100,100,0,0

>where 1
WORLD: X: 250.000 mm Y:-100.000 mm Z: 0.000 mm U: 0.000 deg V: 0.000 deg W: 0.000 deg
JOINT: 1: 0.000 deg 2: 0.000 deg 3: 0.000 mm 4: 0.000 deg
PULSE: 1: 0 pls 2: 0 pls 3: 0 pls 4: 0 pls
WindowsStatus Function

Returns the Windows startup status.

Syntax

WindowsStatus

Return Values

Integer value representing the current Windows startup status. The Windows startup status is returned in a bit image and shows the following status.

<table>
<thead>
<tr>
<th>Function name</th>
<th>System reservation</th>
<th>RC+ enabled</th>
<th>PC enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit number</td>
<td>15 to 2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Details of available functions</td>
<td>Vision Guide (Frame grabber type)</td>
<td>RC+ API</td>
<td>PC file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fieldbus master</td>
<td>PC RS-232C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Data base access</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>DLL call</td>
</tr>
</tbody>
</table>

Description

This function is used to check the controller startup status when the controller configuration is set to “Independent mode”. When the controller configuration is set to “Cooperative mode”, programs cannot be started until both RC+ function and PC function turn available.

WindowsStatus Function Example

Print "The current PC Booting up Status is: ", WindowsStatus
**WOpen Statement**

Opens a file for writing.

**Syntax**

```
WOpen fileName As #fileNumber
.
Close #fileNumber
```

**Parameters**

- `fileName`: A string expression containing the path and file name. If path is omitted, the file in the current directory is specified. See ChDisk for the details.
- `fileNumber`: Integer expression from 30 to 63

**Description**

Opens the specified file using the specified `fileNumber`. This statement is used to open and write data to the specified file. (To append data, refer to the AOpen explanation.)

If the specified filename does not exist on the disk's current directory, WOpen creates the file and writes to it. If the specified filename exists, WOpen erases all of the data in the file and writes to it.

`fileNumber` identifies the file while it is open and cannot be used to refer to a different file until the current file is closed. `fileNumber` is used by other file operations such as Print#, Write, Seek, and Close.

Close closes the file and releases the file number.

It is recommended that you use the FreeFile function to obtain the file number so that more than one task are not using the same number.

**Notes**

- A network path is available.
- File write buffering
  - File writing is buffered. The buffered data can be written with Flush statement. Also, when closing a file with Close statement, the buffered data can be written.

**See Also**

- AOpen, BOpen, Close, Print#, ROpen, UOpen, FreeFile

**WOpen Statement Example**

```plaintext
Integer fileNum, i, j

fileNum = FreeFile
WOpen "TEST.DAT" As #fileNum
For i = 0 To 100
    Print #fileNum, i
Next i
Close #fileNum

fileNum = FreeFile
ROpen "TEST.DAT" As #fileNum
For i = 0 to 100
    Input #fileNum, j
    Print "data = ", j
Next i
Close #fileNum
```
WorkQue_Add Statement

Adds the work queue data (point data and user data) to the specified work queue.

Syntax

```
WorkQue_Add WorkQueNum, pointData [ , userData ]
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **pointData**: The point data to be added to the work queue.
- **userData**: Optional. Real expression used to register the user data along with the point data. If omitted, 0 (real number) is registered as the user data.

Description

`pointData` and `userData` are added to the end of the work queue. When the Sort type is specified by `WorkQue_Sort`, however, they are registered according to the specified Sort type.

When the double registration prevention distance is set by `WorkQue_Reject`, the distance from the previously registered point data is calculated. If the point data is less than that distance, the point data and the user data are not added to the work queue. In this case, an error does not occur.

The upper limit of the work queue data is 1000. The work queue data is deleted by `WorkQue_Remove` when it is finished being used.

See Also

- `WorkQue_AutoRemove`, `WorkQue_Len`, `WorkQue_Reject`, `WorkQue_Remove`, `WorkQue_Sort`

WorkQueAdd Statement Example

```
Integer x, y
Real u

P0 = XY(300, 300, 300, 90, 0, 180)
P1 = XY(200, 280, 150, 90, 0, 180)
P2 = XY(200, 330, 150, 90, 0, 180)
P3 = XY(-200, 280, 150, 90, 0, 180)

Pallet 1, P1, P2, P3, 10, 10
x = 1
y = 1
u = 5.3

WorkQue_Add 1, Pallet(1, x, y), u
```
WorkQue_AutoRemove Statement

Sets the auto delete function to the specified work queue.

Syntax

WorkQue_AutoRemove WorkQueNum , {True | False}

Parameters

WorkQueNum  Integer expression (1 to 16) that specifies the work queue number.
True | False True: Enables the auto delete function.
False: Disables the auto delete function.

Description

Sets the auto delete function to the work queue. When the auto delete is enabled, the point data and the user data are automatically deleted from the work queue when the point data is obtained from the work queue by WorkQue_Get.
When the auto delete is disabled, the point data and the user data are not deleted. To delete them, use WorkQue_Remove.
The user data obtained by WorkQue_UserData are not deleted automatically.

Auto delete function can be set to each work queue.

See Also

WorkQue_AutoRemove Function, WorkQue_Get

WorkQue_AutoRemove Statement Example

WorkQue_AutoRemove 1, True
WorkQue_AutoRemove Function

Returns the state of the auto delete function set to the work queue.

Syntax

WorkQue_AutoRemove (WorkQueNum)

Parameters

WorkQueNum Integer expression (1 to 16) that specifies the work queue number.

Return Values

True if the auto delete function of the specified work queue is enabled, otherwise False.

See Also

WorkQue_AutoRemove, WorkQue_Get

WorkQue_AutoRemove Function Example

Boolean autoremove

autoremove = WorkQue_AutoRemove(1)
WorkQue_Get Function

Returns the point data from the specified work queue.

Syntax

```
WorkQue_Get(WorkQueNum [, index ])
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **index**: Optional. Integer expression that represents the index of the queue data to acquire. (the beginning of the index number is 0)

Return Values

The point data is returned from the specified work queue.

Description

Use WorkQue_Get to acquire the point data from the work queue. If the index is omitted, the first data of the queue data is returned. If the index is specified, the point data of the specified index is returned.

When the auto delete function is enabled by WorkQue_AutoRemove, the point data and the user data are deleted by WorkQue_Get.
When the auto delete is disabled, the point data and the user data are not deleted. To delete them, use WorkQue_Remove.

See Also

WorkQue_AutoRemove, WorkQue.Len, WorkQue_Reject, WorkQue_Remove, WorkQue_Sort

WorkQue_Get Function Example

```
' Jump to the first part in the queue and track it
Jump WorkQue_Get (1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1
WorkQueRemove 1
```
WorkQue_Len Function

Returns the number of the valid work queue data registered to the specified work queue.

Syntax

```plaintext
WorkQue_Len(WorkQueNum)
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.

Return Values

The integer expression representing the number of registered valid work queue data.

Description

Returns the number of registered valid work queue data.

You can also use WorkQue_Len as an argument to the Wait statement.

See Also

WorkQue_Add, WorkQue_Get, WorkQue_Remove

WorkQue_Len Function Example

```plaintext
Do
  Do While WorkQue_Len(1) > 0
    WorkQue_Remove 1, 0
  Loop
  If WorkQue_Len(1) > 0 Then
    Jump WorkQue_Get(1, 0) C0
    On gripper
    Wait .1
    WorkQue_Remove 1, 0
    Jump place
    Off gripper
    Jump idlePos
  EndIf
Loop
```
WorkQue_List Statement

Displays the work queue data list (point data and user data) of the specified work queue.

Syntax

WorkQue_List WorkQueNum [, numOfItems ]

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **numOfItems**: Optional. Integer expression to specify how many items to display. If omitted, all items are displayed.

Note

This command will only work in the command window.

See Also

WorkQue_Add, WorkQue_Get, WorkQue_Remove

WorkQue_List Statement Example

From the command window:

```
> WorkQue_List 1
Queue 0    = XY(     1.000,    1.000,    0.000,    0.000 )  /R /0 (   0.000)
Queue 1    = XY(     3.000,    1.000,    0.000,    0.000 )  /R /0 (   2.000)
Queue 2    = XY(     4.000,    1.000,    0.000,    0.000 )  /R /0 (   3.000)
Queue 3    = XY(     5.000,    1.000,    0.000,    0.000 )  /R /0 (   4.000)
Queue 4    = XY(     6.000,    1.000,    0.000,    0.000 )  /R /0 (   5.000)
```
WorkQue_Reject Statement

Sets and displays the minimum distance for double registration prevention to the specified work queue.

Syntax

```
WorkQue_Reject WorkQueNum [, rejectDistance ]
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **rejectDistance**: Optional when being executed from the command window. Real expression specifying the minimum distance between parts allowed in the queue in millimeters. If omitted, the current rejectDistance is displayed.

Description

Use WorkQue_Reject to specify the minimum distance between parts to prevent double registration of the point data. The work queue cannot be registered when the point data less than the minimum distance is registered by WorkQue_Add. WorkQue_Reject helps the system filter out double registration. The default is 0 mm.

WorkQue_Reject should be executed before adding the work queue data (point data and user data) by WorkQue_Add.

Double registration prevention can be set for each work queue.

See Also

WorkQue_Add, WorkQue_Reject Function

WorkQue_Reject Statement Example

```
WorkQue_Reject 1, 2.5
```
WorkQue_Reject Function

Returns the distance of the double registration prevention set to the specified work queue.

Syntax

```
WorkQue_Reject (WorkQueNum)
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.

Return Values

Real value in millimeters

See Also

- WorkQue_Add, WorkQue_Reject

WorkQue_Reject Function Example

```
Real rejectDist

RejectDist = WorkQue_Reject(1)
```
WorkQue_Remove Statement

Deletes the work queue data (point data and user data) from the specified work queue.

Syntax

```
WorkQue_Remove WorkQueNum [, index | All]
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **index**: Optional. Integer expression that represents the index of the queue data to delete. (the beginning of the index number is 0).
  
  Specify All when deleting all the queue data from the work queue.

Description

Use `WorkQue_Remove` to remove one or more items from a work queue data. Typically, you remove items from the queue after you are finished with the data.

See Also

- `WorkQue_Add`

WorkQue_Remove Statement Example

```
Jump WorkQue_Get(1)
On gripper
Wait .1
Jump place
Off gripper
Wait .1

' Remove the data from the WorkQueue
WorkQue_Remove 1
```


**WorkQue_Sort Statement**

Sets and displays the Sort type of the specified work queue.

**Syntax**

```
WorkQue_Sort WorkQueNum [, SortMethod ]
```

**Parameters**

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **SortMethod**: Specify the Sort method with an integer expression or with the following constant. This can be omitted if executed from the command window. If omitted, the current Sort method is displayed.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUE_SORT_NONE</td>
<td>0</td>
<td>No sorting (registration order to work queue)</td>
</tr>
<tr>
<td>QUE_SORT_POS_X</td>
<td>1</td>
<td>X coordinate ascending order</td>
</tr>
<tr>
<td>QUE_SORT_INV_X</td>
<td>2</td>
<td>X coordinate descending order</td>
</tr>
<tr>
<td>QUE_SORT_POS_Y</td>
<td>3</td>
<td>Y coordinate ascending order</td>
</tr>
<tr>
<td>QUE_SORT_INV_Y</td>
<td>4</td>
<td>Y coordinate descending order</td>
</tr>
<tr>
<td>QUE_SORT_POS_USER</td>
<td>5</td>
<td>User data (real value) ascending order</td>
</tr>
<tr>
<td>QUE_SORT_INV_USER</td>
<td>6</td>
<td>User data (real value) descending order</td>
</tr>
</tbody>
</table>

**Description**

Sets the Sort method to the work queue. When the point data and the user data are added by WorkQue_Add, they are registered to the work queue according to the specified Sort method.

When the user data is set again by WorkQue_UserData, the order of the work queues is changed according to the specified Sort method.

WorkQue_Sort should be executed before adding the work queue data (point data and user data) to the work queue data by WorkQue_Add.

WorkQue_Sort should be executed before setting the user data again by WorkQue_UserData.

Sort method can be set for each work queue.

**See Also**

- WorkQue_Add, WorkQue_UserData

**WorkQue_Sort Statement Example**

```
WorkQue_Sort 1, QUE_SORT_POS_X
```
WorkQue_Sort Function

Returns the Sort method of the specified work queue.

Syntax

\[
\text{WorkQue\_Sort (\textit{WorkQueNum})}
\]

Parameters

\[
\textit{WorkQueNum} \quad \text{Integer expression (1 to 16) that specifies the work queue number.}
\]

Return Values

An integer expression representing the Sort method set to the work queue.

4  = No sorting (registration order to work queue)
5  = X coordinate ascending order
6  = X coordinate descending order
7  = Y coordinate ascending order
8  = Y coordinate descending order
9  = User data (real value) ascending order
10 = User data (real value) descending order

See Also

WorkQue_Add, WorkQue_Sort, WorkQue_UserData

WorkQue_Sort Function Example

\[
\text{Integer quesort}
\]

\[
\text{quesort} = \text{WorkQue\_Sort}(1)
\]
WorkQue_UserData Statement

Resets and displays the user data (real number) registered to the specified work queue.

**Syntax**

```
WorkQue_UserData WorkQueNum [, index] [, userData]
```

**Parameters**

- **WorkQueNum**  
  Integer expression (1 to 16) that specifies the work queue number.

- **index**  
  Integer expression that represents the index of the work queue data. (the beginning of the index number is 0). Optional when executing from the command window.

- **userData**  
  Integer expression that represents the user data to be set again. This can be omitted when executed from the command window. If omitted, the current user data (real expression) is displayed.

**Description**

Resets and displays the user data currently registered to the work queue.

If the Sort method is specified by WorkQue_Sort, the order of the work queue data is changed according to the specified Sort method.

- QUE_SORT_POS_USER : User data (real expression) ascending order
- QUE_SORT_INV_USER : User data (real expression) descending order

**See Also**

- WorkQue_UserData Function
- WorkQue_UserData Example

```
WorkQue_UserData 1, 1, angle
```
WorkQue_UserData Function

Returns the user data (real value) registered to the specified work queue.

Syntax

```
WorkQue_UserData (WorkQueNum [, index])
```

Parameters

- **WorkQueNum**: Integer expression (1 to 16) that specifies the work queue number.
- **index**: Optional. Integer expression that represents the index of the work queue data. (the first index number is 0).

Return Values

Real value.

See Also

WorkQue_UserData

WorkQue_UserData Function Example

```
' Remove from queue
angle = WorkQue_UserData(1)  ' default to queue index of 0
Jump WorkQue_Get(1) :U(angle)
WorkQue_Remove 1
```
Wrist Statement

Sets the wrist orientation of a point.

Syntax

(1) Wrist point [, Flip | NoFlip]
(2) Wrist

Parameters

point P number or P(expr) or point label.
Flip | NoFlip Representing wrist orientation.

Return Values

When both parameters are omitted, the wrist orientation is displayed for the current robot position. If Flip | NoFlip is omitted, the wrist orientation for the specified point is displayed.

See Also

Elbow, Hand, J4Flag, J6Flag, Wrist Function

Wrist Statement Example

Wrist P0, Flip
Wrist P(mypoint), NoFlip

P1 = 320.000, 400.000, 350.000, 140.000, 0.000, 150.000

Wrist P1, NoFlip
Go P1

Wrist P1, Flip
Go P1
Wrist Function

Returns the wrist orientation of a point.

**Syntax**

```
Wrist [(point)]
```

**Parameters**

- **point**: Optional. \( P_{number} \) or \( P(expr) \) or point label or point expression. If `point` is omitted, then the wrist orientation of the current robot position is returned.

**Return Values**

1. NoFlip (/NF)
2. Flip (/F)

**See Also**

Elbow, Hand, J4Flag, J6Flag, Wrist Statement

**Wrist Function Example**

```
Print Wrist(pick)
Print Wrist(P1)
Print Wrist
Print Wrist(P1 + P2)
```
Write Statement

Writes characters to a file or communication port without end of line terminator.

Syntax

Write #portNumber, string

Parameters

- **portNumber**
  - ID number that specifies the file or communications port.
  - File number can be specified in ROpen, WOpen, AOpen statements.
  - Communication port number can be specified in OpenCom (RS-232C) or OpenNet (TCP/IP) statements.

- **string**
  - String expression that will be written to the file.

Description

Write is different from Print in that it does not add an end of line terminator.

Note

File write buffering

File writing is buffered. The buffered data can be written with Flush statement. Also, when closing a file with Close statement, the buffered data can be written.

See Also

Print, Read

Write Statement Example

```
OpenCom #1
For i = 1 to 10
    Write #1, data$(i)
Next i
CloseCom #1
```
WriteBin Statement

Writes binary data to a file or communications port.

Syntax

```
WriteBin #portNumber, data
WriteBin #portNumber, array(), count
```

Parameters

- **portNumber**: ID number that specifies the file or communications port.
  
  File number can be specified in BOpen statements.
  
  Communication port number can be specified in OpenCom (RS-232C) or OpenNet (TCP/IP) statements.

- **data**: Integer expression containing the data to be written.

- **array()**: Name of a byte, integer, or long array variable that contains the data bytes to be written.
  
  Specify a one dimension array variable.

- **count**: Specifies the number of bytes to be written.
  
  The specified count has to be less than or equal to the number of array elements and also smaller than 256 bytes.
  
  If the communication port (TCP/IP) is the subject, the count has to be less than or equal to the number of array and also smaller than 1024 bytes.

See Also

ReadBin, Write

WriteBin Statement Example

```
Integer i, data(100)

OpenCom #1
For i = 0 To 100
    WriteBin #1, i
Next I
WriteBin #1, data(), 100
CloseCom #1
```
Xor Operator

Performs the bitwise Xor operation (exclusive OR) on two expressions.

**Syntax**

```plaintext
result = expr1 Xor expr2
```

**Parameters**

- `expr1, expr2` A numeric value, or a variable name.
- `result` An integer.

**Result**

Returns a result of bitwise Xor operation.

**Description**

The Xor operator performs the bitwise Xor operation on the values of the operands. Each bit of the result is the Xored value of the corresponding bits of the two operands.

<table>
<thead>
<tr>
<th>If bit in expr1 is</th>
<th>And bit in expr2 is</th>
<th>The result is</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

**See Also**

And, LShift, Not, Or, Rshift

**Xor Operator Example**

```plaintext
>print 2 Xor 6
4
>
```
Xqt Statement

Initiates execution of a task from within another task.

Syntax
Xqt [taskNumber,] funcName [(argList)] [, Normal | NoPause | NoEmgAbort ]

Parameters

- **taskNumber**: Optional. The task number for the task to be executed. The range of the task number is 1 to 32. For background tasks, specifies integer value from 65 to 80.

- **funcName**: The name of the function to be executed.

- **argList**: Optional. List of arguments that are passed to the function procedure when it is called. Multiple arguments are separated by commas.

- **taskType**: Optional. Usually omitted. For background tasks, specifying a task type means nothing.

  - **Normal**: Executes a normal task.
  - **NoPause**: Executes a task that does not pause at Pause statement or Pause input signal occurrence or Safety Door Open.
  - **NoEmgAbort**: Executes a task that continues processing at Emergency Stop or error occurrence.

Description
Xqt starts the specified function and returns immediately.

Normally, the **taskNumber** parameter is not required. When **taskNumber** is omitted, SPEL+ automatically assigns a task number to the function, so you don't have to keep track of which task numbers are in use.

Notes

- **Task Type**
  - Specify **NoPause** or **NoEmgAbort** as a task type to execute a task that monitors the whole controller. However, be sure to use these special tasks based on the understanding of the task motion using SPEL+ or restriction of special tasks.
  - For details of special tasks, refer to the section **Special Tasks** in the EPSON RC+ User's Guide.

- **Background task**
  - When executing Xqt in a background task, the generated task is also the background task.
  - To execute the main function from a background task, use the StartMain statement.
### Unavailable Commands in NoEmgAbort Task and background task

The following commands cannot be executed in NoEmgAbort task and background task.

<table>
<thead>
<tr>
<th>A</th>
<th>Accel</th>
<th>G</th>
<th>Go</th>
<th>S</th>
<th>Sense</th>
<th>V</th>
<th>VLoad</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccelR</td>
<td>H</td>
<td>Home</td>
<td>SFree</td>
<td></td>
<td></td>
<td></td>
<td>VRun</td>
</tr>
<tr>
<td>AccelS</td>
<td></td>
<td>HomeClr</td>
<td>SingularityAngle</td>
<td></td>
<td></td>
<td></td>
<td>VSave</td>
</tr>
<tr>
<td>AIO_TrackingStart</td>
<td></td>
<td>HomeSet</td>
<td>SingularitySpeed</td>
<td></td>
<td></td>
<td></td>
<td>VSaveImage</td>
</tr>
<tr>
<td>AIO_TrackingEnd</td>
<td></td>
<td>Hordr</td>
<td>SLock</td>
<td></td>
<td></td>
<td></td>
<td>VSet</td>
</tr>
<tr>
<td>Arc</td>
<td>I</td>
<td>Inertia</td>
<td>SoftCP</td>
<td></td>
<td></td>
<td></td>
<td>VSaveModel</td>
</tr>
<tr>
<td>Arc3</td>
<td>J</td>
<td>JTran</td>
<td>Speed</td>
<td></td>
<td></td>
<td></td>
<td>VSet</td>
</tr>
<tr>
<td>Arch</td>
<td></td>
<td>Jump</td>
<td>SpeedFactor</td>
<td></td>
<td></td>
<td></td>
<td>VShowModel</td>
</tr>
<tr>
<td>Arm</td>
<td></td>
<td>Jump3</td>
<td>SpeedR</td>
<td></td>
<td></td>
<td></td>
<td>VStatsReset</td>
</tr>
<tr>
<td>ArmSet</td>
<td></td>
<td>Jump3CP</td>
<td>SpeedS</td>
<td></td>
<td></td>
<td></td>
<td>VStatsResetAll</td>
</tr>
<tr>
<td>ArmClr</td>
<td></td>
<td>JRange</td>
<td>SyncRobots</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AutoLJM</td>
<td>L</td>
<td>LimitTorque</td>
<td>T</td>
<td>TC</td>
<td></td>
<td></td>
<td>VStatsSave</td>
</tr>
<tr>
<td>AvoidSingularity</td>
<td></td>
<td>LimZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>VTeach</td>
</tr>
<tr>
<td>Base</td>
<td></td>
<td>LimZMargin</td>
<td>Till</td>
<td></td>
<td></td>
<td></td>
<td>VTrain</td>
</tr>
<tr>
<td>BGo</td>
<td></td>
<td>Local</td>
<td>TLSet</td>
<td></td>
<td></td>
<td></td>
<td>Weight</td>
</tr>
<tr>
<td>BMove</td>
<td></td>
<td>LocalClr</td>
<td>TLClr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Box</td>
<td>M</td>
<td>MCal</td>
<td>TMove</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BoxClr</td>
<td></td>
<td>MCordr</td>
<td>Tool</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brake</td>
<td></td>
<td>Motor</td>
<td>Trap</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Cnv_AbortTrack</td>
<td>Move</td>
<td>V</td>
<td>VCal</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_Accel</td>
<td>O</td>
<td>OLAccel</td>
<td>VcalPoints</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_DownStream</td>
<td>P</td>
<td>Pass</td>
<td>VCls</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_Fine</td>
<td></td>
<td>Pg_LSpeed</td>
<td>VCreateCalibration</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_Mode</td>
<td></td>
<td>Pg_Scan</td>
<td>VCreateObject</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_QueAdd</td>
<td></td>
<td>Plane</td>
<td>VCreateSequence</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_QueMove</td>
<td></td>
<td>PlaneClr</td>
<td>VDefArm</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_QueReject</td>
<td></td>
<td>Power</td>
<td>VDefGetMotionRange</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_QueUserData</td>
<td></td>
<td>PTPBoost</td>
<td>VDefLocal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_Trigger</td>
<td></td>
<td>Pulse</td>
<td>VDefSetMotionRange</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_UpStream</td>
<td>Q</td>
<td>QP</td>
<td>VDefTool</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CP</td>
<td>QPDecelR</td>
<td>VDeleteCalibration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Curve</td>
<td>QPDecelS</td>
<td>VDeleteObject</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVMove</td>
<td>R</td>
<td>Range</td>
<td>VDeleteSequence</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>ECP</td>
<td>Reset *1</td>
<td>VGet</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECPClr</td>
<td></td>
<td>Restart *2</td>
<td>VGoCenter</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECPSet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>Find</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1 Reset Error can be executed
*2 Executable from the Trap Error processing task
*3 Executable from the background tasks

**DO NOT use XQT command repeatedly in Loop statements.**

Do not use XQT command repeatedly in Loop statements such as Do…Loop.

The controller may freeze up. If you use Loop statements repeatedly, make sure to add Wait command (Wait 0.1).

**See Also**

Function/Fend, Halt, Resume, Quit, Startmain, Trap
**Xqt Statement**

**Xqt Statement Example**

```plaintext
Function main
  Xqt flash ' Start flash function as task 2
  Xqt Cycle(5) ' Start Cycle function as task 3
  Do
    Wait 3 ' Execute task 2 for 3 seconds
    Halt flash ' Suspend the task
    Wait 3
    Resume flash ' Resume the task
  Loop
Fend

Function Cycle(count As Integer)
  Integer i
  For i = 1 To count
    Jump pick
    On vac
    Wait .2
    Jump place
    Off vac
    Wait .2
  Next i
Fend

Function flash
  Do
    On 1
    Wait 0.2
    Off 1
    Wait 0.2
  Loop
Fend
```
XY Function

Returns a point from individual coordinates that can be used in a point expression.

Syntax

\[ XY(x, y, z, u[, v, w]) \]

Parameters

- \( x \)  
  Real expression representing the X coordinate.
- \( y \)  
  Real expression representing the Y coordinate.
- \( z \)  
  Real expression representing the Z coordinate.
- \( u \)  
  Real expression representing the U coordinate.
- \( v \)  
  Optional for 6-Axis robots (including N series). Real expression representing the V coordinate.
- \( w \)  
  Optional for 6-Axis robots (including N series). Real expression representing the W coordinate.

Return Values

A point constructed from the specified coordinates.

Description

When you don’t use the additional ST axis, there are nothing in particular to be care of.

You can move the manipulator to the specified coordinate with XY function like below:

\[ \text{Go } XY(60,30,-50,45) \]

When you use the additional ST axis, you need to be careful.

XY function returns the only robot point data, not including the additional axis.

If you use XY function lick this: Go \( XY(60,30,-50,45) \), the manipulator will move to the specified coordinate but the additional axis will not move. If you want to move the additional axis as well, specify like this: Go \( XY(60,30,-50,45) : ST(10,20) \).

For the details of additional axis, refer to EPSON RC+ Users Guide: 20. Additional Axis.

See Also

JA, Point Expression, ST Function

XY Function Example

\[ P10 = XY(60, 30, -50, 45) + P20 \]

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
XYLim Statement

Sets or displays the permissible XY motion range limits for the robot.

Syntax

```plaintext
XYLim  minX, maxX, minY, maxY [, minZ [, maxZ]]
```

Parameters

- **minX**: The minimum X coordinate position to which the manipulator may travel. (The manipulator may not move to a position with the X Coordinate less than minX.)
- **maxX**: The maximum X coordinate position to which the manipulator may travel. (The manipulator may not move to a position with the X Coordinate greater than maxX.)
- **minY**: The minimum Y coordinate position to which the manipulator may travel. (The manipulator may not move to a position with the Y Coordinate less than minY.)
- **maxY**: The maximum Y coordinate position to which the manipulator may travel. (The manipulator may not move to a position with the Y Coordinate greater than maxY.)
- **minZ** (Optional): The minimum Z coordinate position to which the manipulator may travel. (The manipulator may not move to a position with the Z Coordinate less than minZ.)
- **maxZ** (Optional): The maximum Z coordinate position to which the manipulator may travel. (The manipulator may not move to a position with the Z Coordinate greater than maxZ.)

Return Values

Displays current XYLim values when used without parameters.

Description

XYLim is used to define XY motion range limits. Many robot systems allow users to define joint limits but the SPEL+ language allows both joint limits and motion limits to be defined. In effect this allows users to create a work envelope for their application. (Keep in mind that joint range limits are also definable with SPEL.)

The motion range established with XYLim values applies to motion command target positions only, and not to motion paths from starting position to target position. Therefore, the arm may move outside the XYLim range during motion. (i.e. The XYLim range does not affect Pulse.)

Robot parameter data is stored in compact flash in controller. Therefore, writing to command flash occurs when executing this command. Frequent writing to compact flash affect to lifetime of compact flash. We recommend to use this command minimally.

Notes

**Turning Off Motion Range Checking**

There are many applications which don't require Motion Range limit checking and for that reason there is a simple method to turn this limit checking off. To turn motion range limit checking off, define the Motion Range Limit values for minX, maxX, minY, and maxY to be "0". For example XYLim 0, 0, 0, 0.

**Default Motion Range Limit Values**

The default values for the XYLim instruction are "0, 0, 0, 0". (Motion Range Limit Checking is turned off.)
Tip

Point & Click Setup for XYLim

EPSON RC+ has a point and click dialog box for defining the motion range limits. The simplest method to set the XYLim values is by using the XYZ Limits page on the Robot Manager.

See Also

Range

XYLim Statement Example

This simple example from the command window sets and then displays the current XYLim setting:

```
> xylim -200, 300, 0, 500

> XYLim

-200.000, 300.000, 0.000, 500.000
```
XYLim Function

Returns point data for either upper or lower limit of XYLim region.

Syntax

```
XYLim(limit)
```

Parameters

```
limit
```

Integer expression that specifies which limit to return.
1: Lower limit.
2: Upper limit.

Return Values

When “1” is specified for reference data, returns X axis lower limit position specified in XYLim as X of point data, Y axis lower limit position as Y, and Z axis lower limit position as Z.
When “2” is specified for reference data, returns X axis upper limit position specified in XYLim as X of point data, Y axis upper limit position as Y, and Z axis upper limit position as Z.

See Also

XYLim Statement

XYLim Function Example

```pascal
P1 = XYLim(1)
P2 = XYLim(2)
```
XYLimClr Statement

Clears the XYLim definition.

**Syntax**

`XYLimClr`

**See Also**

`XYLim`, `XYLimDef`

**XYLimClr Function Example**

This example uses the XYLimClr function in a program:

```c
Function ClearXYLim
    If XYLimDef = True Then
        XYLimClr
    EndIf
EndFunction
```
XYLimDef Function

Returns whether XYLim has been defined or not.

Syntax

XYLimDef

Return Values

True if XYLim has been defined, otherwise False.

See Also

XYLim, XYLimClr

XYLimDef Function Example

This example uses the XYLimDef function in a program:

```plaintext
Function ClearXYLim

  If XYLimDef = True Then
    XYLimClr
  EndIf

EndFend
```
## SPEL+ Error Messages

To get help for any SPEL+ error, place the cursor on the error message in the run or command windows and press the F1 key.

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Controller control program started.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Termination due to low voltage of the power supply.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Controller control program has completed.</td>
<td>Stores this log when the controller is rebooted from EPSON RC+ or TP1.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Preserve variables save area has been cleaned.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Function Main started.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Function Main started. Later same logs are skipped.</td>
<td>Skip the log &quot;Function Main started.&quot; to prevent system history space run out.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Serial number has been saved.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>System backup has been executed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>System restore has been executed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Robot parameters have been initialized.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Offset pulse value between the encoder origin and the home sensor (HOFs) is changed.</td>
<td>-</td>
<td>Value after change</td>
<td>Value before change</td>
</tr>
<tr>
<td>17</td>
<td>Message saving mode activated.</td>
<td>Uncommon event.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Conversion of Robot Parameter file has been executed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>DU firmware has been installed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Enable setting in Teach mode has been saved.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Enable setting in Teach mode has been changed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>EStop has been executed.</td>
<td>-</td>
<td>Robot number executing motion command</td>
<td>Controller status</td>
</tr>
<tr>
<td>24</td>
<td>Safeguard has opened.</td>
<td>-</td>
<td>Robot number executing motion command</td>
<td>Controller status</td>
</tr>
<tr>
<td>25</td>
<td>Robot setting has changed.</td>
<td>-</td>
<td>Robot number</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>Alarm setting has changed.</td>
<td>-</td>
<td>Alarm number</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>The battery alarm for the controller was reset.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>The battery alarm for the robot was reset.</td>
<td>-</td>
<td>Robot number</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>The grease alarm for the robot was reset.</td>
<td>-</td>
<td>Robot number</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>Device connected to Controller.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>Display device has changed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>Working mode has changed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>104</td>
<td>Cooperative mode has changed.</td>
<td>-</td>
<td>0: Independent</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1: Cooperative</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>Controller firmware has been installed.</td>
<td>-</td>
<td>1: Setup</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2: Initialize</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3: Upgrade</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4: Recover</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>IP address has been restored.</td>
<td>May store this log when the controller firmware is installed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>Controller rebooted</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>RC+ connected to the Controller.</td>
<td>-</td>
<td>1: Ethernet</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2: USB</td>
<td></td>
</tr>
<tr>
<td>121</td>
<td>TP connected to the Controller.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>123</td>
<td>RC+ disconnected from the Controller.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>TP disconnected from the Controller.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>126</td>
<td>Working mode changed to AUTO.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>127</td>
<td>Working mode changed to Program.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>Working mode changed to Teach.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>Remote Ethernet connected to the Controller</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>Remote Ethernet disconnected to the Controller</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>131</td>
<td>Remote Com connected to the Controller</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Logout status</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0: Normal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1: Abnormal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(Time-out)</td>
<td></td>
</tr>
<tr>
<td>133</td>
<td>Working mode changed to Test.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400</td>
<td>The battery alarm for the controller occurred. Replace the battery and</td>
<td>Replace the battery and reset the alarm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>reset the alarm.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>The battery alarm for the robot occurred. Replace the battery and</td>
<td>Replace the battery and reset the alarm.</td>
<td></td>
<td>Robot number</td>
</tr>
<tr>
<td></td>
<td>reset the alarm.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>402</td>
<td>The grease alarm occurred. Grease the robot and reset the alarm.</td>
<td>Grease the robot and reset the alarm.</td>
<td></td>
<td>Robot number</td>
</tr>
<tr>
<td>410</td>
<td>The battery alarm for the controller occurred. Replace the battery and</td>
<td>Replace the battery. After replacing the battery, reset the alarm in</td>
<td>1000 times of</td>
<td>1000 times</td>
</tr>
<tr>
<td></td>
<td>reset the alarm.</td>
<td></td>
<td>EPSON RC+ 7.0-[Tools]-</td>
<td>boundary value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Controller]-[Maintenance].</td>
<td></td>
</tr>
<tr>
<td>411</td>
<td>The battery alarm for the robot occurred. Replace the battery and</td>
<td>Replace the battery. After replacing the battery, reset the alarm in</td>
<td>1000 times of</td>
<td>1000 times</td>
</tr>
<tr>
<td></td>
<td>reset the alarm.</td>
<td></td>
<td>EPSON RC+ 7.0-[Tools]-</td>
<td>boundary value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Controller]-[Maintenance].</td>
<td></td>
</tr>
<tr>
<td>412</td>
<td>The belt alarm occurred. Replace the belt and reset the alarm.</td>
<td>Replace the timing belt. After replacing the timing belts, reset the</td>
<td>1000 times of</td>
<td>1000 times</td>
</tr>
<tr>
<td></td>
<td></td>
<td>alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>EPSON RC+ 7.0-[Tools]-</td>
<td>boundary value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Controller]-[Maintenance].</td>
<td></td>
</tr>
<tr>
<td>413</td>
<td>The grease alarm for the ball screw spline occurred. Grease the ball</td>
<td>Grease up the ball screw spline. After greasing up, reset the alarm in</td>
<td>1000 times of</td>
<td>1000 times</td>
</tr>
<tr>
<td></td>
<td>screw spline occurred. Grease the ball screw spline units and reset the</td>
<td></td>
<td>EPSON RC+ 7.0-[Tools]-</td>
<td>boundary value</td>
</tr>
<tr>
<td></td>
<td>alarm.</td>
<td></td>
<td>[Controller]-[Maintenance].</td>
<td></td>
</tr>
<tr>
<td>414</td>
<td>The motor alarm occurred. Replace the motor.</td>
<td>Replace the motor. After replacing the motor, reset the alarm in EPSON</td>
<td>1000 times of</td>
<td>1000 times</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>EPSON RC+ 7.0-[Tools]-</td>
<td>boundary value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Controller]-[Maintenance].</td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>415</td>
<td>The gear alarm occurred. Replace the gear units and reset the alarm.</td>
<td>Replace the gear units. After replacing the gear units, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>416</td>
<td>The ball screw spline alarm occurred. Replace the ball screw spline and reset the alarm.</td>
<td>Replace the ball screw spline. After replacing the ball screw spline, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>420</td>
<td>The battery alarm for the controller occurred. Replace the battery and reset the alarm.</td>
<td>Replace the battery. After replacing the battery, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>421</td>
<td>The battery alarm for the robot occurred. Replace the battery and reset the alarm.</td>
<td>Replace the battery. After replacing the ball screw spline, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>422</td>
<td>The belt alarm occurred. Replace the belt and reset the alarm.</td>
<td>Replace the timing belt. After replacing the timing belts, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>423</td>
<td>The grease alarm for the ball screw spline occurred. Grease the ball screw spline and reset the alarm.</td>
<td>Grease up the ball screw spline. After greasing up, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>424</td>
<td>The motor alarm occurred. Replace the motor and reset the alarm.</td>
<td>Replace the motor. After replacing the motor, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>425</td>
<td>The gear alarm occurred. Replace the gear units and reset the alarm.</td>
<td>Replace the gear units. After replacing the gear units, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>426</td>
<td>The ball screw spline alarm occurred. Replace the ball screw spline and reset the alarm.</td>
<td>Replace the ball screw spline. After replacing the ball screw spline, reset the alarm in EPSON RC+ 7.0-[Tools]-[Controller]-[Maintenance].</td>
<td>1000 times of consumptio n rate</td>
<td>1000 times of boundary value</td>
</tr>
<tr>
<td>501</td>
<td>Trace history is active.</td>
<td>Effects system performance if trace history is active.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>502</td>
<td>Memory has been initialized.</td>
<td>When this error occurs, the value of the Global Preserve variable will be initialized. Replace the CPU board battery. Replace the CPU board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>503</td>
<td>Found Hard disk error. You should replace the hard disk ASAP.</td>
<td>This is a warning of the hard disk failure. Replace the hard disk as soon as possible.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>504</td>
<td>An Error occurred on a Background Task.</td>
<td>Make sure there are no problems in the system and continue the operation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>505</td>
<td>Controller was rebooted.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>507</td>
<td>The controller is started by using the previous initial setting file since the initial setting file is corrupted. Check the settings.</td>
<td>Setting changes from the previous start may not be saved. Please check the settings.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>511</td>
<td>Battery voltage of the CPU board backup is lower than the allowed voltage. Replace the CPU board battery.</td>
<td>Replace the CPU board battery immediately. Keep the power to the controller ON as far as possible until you replace the battery.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>512</td>
<td>5V input voltage for the CPU board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by a 5V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>513</td>
<td>24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by a 24V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>514</td>
<td>Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop the controller as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>515</td>
<td>Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>516</td>
<td>Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>517</td>
<td>Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop the controller as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>518</td>
<td>54V input voltage for the CPU board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by a 54V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>519</td>
<td>3.3V input voltage for the CPU board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by a 3.3V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>520</td>
<td>DC input voltage for the CPU board is lower than or bigger than the allowed voltage.</td>
<td>If normal voltage is not generated by a DC power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>521</td>
<td>DU1 3.3V input voltage for the board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by 3.3V of Drive Unit 1 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>522</td>
<td>DU1 5V input voltage for the board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by 5V of Drive Unit 1 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>523</td>
<td>DU1 24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 24V of Drive Unit 1 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>524</td>
<td>DU1 Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop Drive Unit 1 as soon as possible and check whether the ambient temperature of Drive Unit 1 is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>525</td>
<td>DU1 Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter of Drive Unit 1 is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>526</td>
<td>DU1 Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter of Drive Unit 1 is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>531</td>
<td>DU2 3.3V input voltage for the board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by 3.3V of Drive Unit 2 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>532</td>
<td>DU2 5V input voltage for the board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by 5V of Drive Unit 2 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>533</td>
<td>DU2 24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 24V of Drive Unit 2 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>534</td>
<td>DU2 Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop Drive Unit 2 as soon as possible and check whether the ambient temperature of Drive Unit 2 is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>535</td>
<td>DU2 Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter of Drive Unit 2 is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>536</td>
<td>DU2 Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter of Drive Unit 2 is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>541</td>
<td>DU3 3.3V input voltage for the board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by 3.3V of Drive Unit 3 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>542</td>
<td>DU3 5V input voltage for the board is lower than the allowed voltage.</td>
<td>If normal voltage is not generated by 5V of Drive Unit 3 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>543</td>
<td>DU3 24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 24V of Drive Unit 3 power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>544</td>
<td>DU3 Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop Drive Unit 3 as soon as possible and check whether the ambient temperature of Drive Unit 3 is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>545</td>
<td>DU3 Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter of Drive Unit 3 is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>546</td>
<td>DU3 Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter of Drive Unit 3 is not clogged up. If the warning is not cleared after the controller is rebooted, replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>550</td>
<td>Communication with the Compact Vision is disconnected. Check the network wiring.</td>
<td>Check the connection between the controller and the compact vision.</td>
<td>Camera No.</td>
<td></td>
</tr>
<tr>
<td>551</td>
<td>Compact Vision CPU fan RPM has decreased. Clean the fan filter and / or replace the fan.</td>
<td>Check whether the fan filter of the compact vision is not clogged up. If the warning is not cleared after the controller and the compact vision are rebooted, replace the CPU fan.</td>
<td>Camera No.</td>
<td>Current value</td>
</tr>
<tr>
<td>552</td>
<td>Compact Vision CPU fan RPM has decreased. Clean the fan filter and / or replace the fan.</td>
<td>Replace the CPU fan of the compact vision.</td>
<td>Camera No.</td>
<td>Current value</td>
</tr>
<tr>
<td>553</td>
<td>Compact Vision chassis fan RPM has decreased. Replace the fan.</td>
<td>Check whether the fan filter of the compact vision is not clogged up. If the warning is not cleared after the controller and the compact vision are rebooted, replace the system fan.</td>
<td>Camera No.</td>
<td>Current value</td>
</tr>
<tr>
<td>554</td>
<td>Compact Vision chassis fan RPM has decreased. Replace the fan.</td>
<td>Replace the system fan of the compact vision.</td>
<td>Camera No.</td>
<td>Current value</td>
</tr>
<tr>
<td>555</td>
<td>Compact Vision CPU temperature is too high. Check the installation environment (ventilation, ambient temperature, etc.)</td>
<td>Check whether the fan filter of the compact vision is not clogged up. If the warning is not cleared after the controller and the compact vision are rebooted, check the installation environment (surrounding space, ambient temperature) of the compact vision.</td>
<td>Camera No.</td>
<td>1000 times of current value</td>
</tr>
<tr>
<td>556</td>
<td>Compact Vision CPU temperature is too high. Check the installation environment (ventilation, ambient temperature, etc.)</td>
<td>Check whether the fan filter of the compact vision is not clogged up. If the warning is not cleared after the controller and the compact vision are rebooted, check the installation environment (surrounding space, ambient temperature) of the compact vision.</td>
<td>Camera No.</td>
<td>1000 times of current value</td>
</tr>
<tr>
<td>557</td>
<td>Compact Vision backup battery voltage is low. Replace the battery.</td>
<td>Replace the backup battery of the compact vision.</td>
<td>Camera No.</td>
<td>1000 times of current value</td>
</tr>
<tr>
<td>558</td>
<td>Compact Vision backup battery voltage is low. Replace the battery.</td>
<td>Replace the backup battery of the compact vision.</td>
<td>Camera No.</td>
<td>1000 times of current value</td>
</tr>
<tr>
<td>559</td>
<td>Compact Vision process was terminated abnormally. Restart the Compact Vision unit.</td>
<td>If the warning is not cleared after the controller and the compact vision are rebooted, initialize the compact vision.</td>
<td>Camera No.</td>
<td></td>
</tr>
<tr>
<td>560</td>
<td>Compact Vision available memory is low. Restart the Compact Vision unit.</td>
<td>If the warning is not cleared after the controller and the compact vision are rebooted, initialize the compact vision.</td>
<td>Camera No.</td>
<td>Current value</td>
</tr>
<tr>
<td>561</td>
<td>Compact Vision available disk space is low. Reduce the number of objects that use models (Geometric, Correlation, DefectFinder, etc.)</td>
<td>Check the vision sequence if it has unnecessary models which can be reduced. Consider to use the USB memory.</td>
<td>Camera No.</td>
<td>Current value</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>562</td>
<td>A critical hardware error occurred in the Compact Vision unit. Check the hardware condition such as internal wiring.</td>
<td>If the warning is not cleared after the controller and the compact vision are rebooted, initialize the compact vision.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>563</td>
<td>A critical hardware error occurred in the Compact Vision unit. Check the hardware condition such as internal wiring.</td>
<td>If the warning is not cleared after the controller and the compact vision are rebooted, replace the LED/SW board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>569</td>
<td>Communication with the Compact Vision recovered.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>597</td>
<td>The PTP motion to avoid the singularity point has completed.</td>
<td>PTP motion for the singularity avoidance was completed. Clicking the same jog button will operate the robot in the normal jog motion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>598</td>
<td>Robot stopped due to collision detection</td>
<td>Move the manipulator to the direction avoiding collision</td>
<td></td>
<td></td>
</tr>
<tr>
<td>599</td>
<td>Jogging attempted near singularity point.</td>
<td>The robot could not jog in the CP motion (default). Clicking the same jog button will operate the robot in the PTO motion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>700</td>
<td>Motor driver type does not match the current robot model. Check the robot model. Replace the motor driver.</td>
<td>Check the robot model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>736</td>
<td>Encoder has been reset. Reboot the controller.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>737</td>
<td>Low voltage from the encoder battery. Replace the battery.</td>
<td>Turn OFF the controller and replace the battery. For the battery replacement procedure, refer to Maintenance in the Manipulator manual.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>752</td>
<td>Servo alarm D.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>1001</td>
<td>Operation Failure. Command parameter is invalid.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1002</td>
<td>Requested data cannot be accessed. The data is not set up or the range is invalid.</td>
<td>Check whether the target I/O, variables, and tasks exist.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1003</td>
<td>The password is invalid</td>
<td>Enter the correct password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1004</td>
<td>Cannot execute with unsupported version.</td>
<td>Use the correct version file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1005</td>
<td>Cannot execute with invalid serial number.</td>
<td>Use the backup data for the same controller to restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1006</td>
<td>Cannot execute with invalid Robot model.</td>
<td>Use the backup data for the same controller to restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1007</td>
<td>Cannot execute with invalid Controller.</td>
<td>Use the supported installer.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010</td>
<td>Remote setup error. Cannot assign R-IO input number to remote input.</td>
<td>Specify the input number excluding the R-IO input number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011</td>
<td>Remote setup error. Cannot assign a bit number which does not exist to a remote I/O signal. Check the fieldbus slave size.</td>
<td>Check the fieldbus slave size.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1012</td>
<td>Remote setup error. Cannot assign a bit number which does not exist to a remote I/O signal. Check the fieldbus master size.</td>
<td>Check the fieldbus master size.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1013</td>
<td>Fieldbus slave failure. Cannot change the size because it currently includes a remote I/O signal.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1014</td>
<td>Fieldbus master failure. Cannot change the size because it currently includes a remote I/O signal.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1015</td>
<td>Remote setup error. Cannot assign Hand-IO input/output number to remote input.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1020</td>
<td>Cannot execute in recovery mode.</td>
<td>Boot the controller as normal.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1021</td>
<td>Cannot execute due to controller initialization failure.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1022</td>
<td>Cannot execute without the project being open.</td>
<td>Open a project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1023</td>
<td>Cannot execute while the project is open.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1024</td>
<td>Cannot activate from remote.</td>
<td>Enable the remote input.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1025</td>
<td>Execution in Teach mode is prohibited.</td>
<td>Change to the AUTO mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1026</td>
<td>Cannot execute in Teach mode except from TP.</td>
<td>Change to the AUTO mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1027</td>
<td>Cannot execute in Auto mode.</td>
<td>Change to the Program mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1028</td>
<td>Cannot execute in Auto mode except from the main console.</td>
<td>Change to the Program mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1029</td>
<td>Cannot execute from OP.</td>
<td>Enable the OP input.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1030</td>
<td>Does not allow Operation mode to be changed.</td>
<td>Change to the Auto mode with a console in the Program mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1031</td>
<td>Cannot execute while tasks are executing.</td>
<td>Stop the task and then execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1032</td>
<td>Cannot execute while the maximum number of tasks are executing.</td>
<td>Stop the task and then execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1033</td>
<td>Cannot execute during asynchronous motion command.</td>
<td>Execute after the motion ends.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1034</td>
<td>Asynchronous command stopped during operation.</td>
<td>The asynchronous command already stopped when the controller received a stop command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1035</td>
<td>Cannot execute when Remote I/O enabled except from the remote device.</td>
<td>The command cannot be executed by the console except the remote I/O when AutoMode output of the remote I/O is ON.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1036</td>
<td>Cannot execute when Remote Ethernet enabled except from the remote Ethernet device.</td>
<td>The command cannot be executed by the console except the remote Ethernet when Auto flag of the remote Ethernet is ON.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1037</td>
<td>Execution is prohibited.</td>
<td>Prohibited command was executed while executing the program. Stop the program, and then execute the command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1038</td>
<td>Cannot execute during Emergency Stop status.</td>
<td>Cancel the Emergency Stop status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1039</td>
<td>Cannot execute while the safeguard is open.</td>
<td>Close the safeguard.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1040</td>
<td>Cannot execute during error condition.</td>
<td>Cancel the error condition.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1041</td>
<td>Cannot execute when the remote pause input is ON.</td>
<td>Change the remote pause input to OFF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1042</td>
<td>Input waiting condition is the only available condition to input.</td>
<td>The controller received an input while it was not in the Input waiting condition.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1043</td>
<td>Cannot execute during file transfer.</td>
<td>Execute after the file transmission.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1044</td>
<td>Cannot cancel the command executed from other devices.</td>
<td>Cancel the motion command from the device the command was issued from.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1045</td>
<td>Cannot execute after low voltage was detected.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1046</td>
<td>Other devices are in program mode.</td>
<td>Check connection of other devices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1047</td>
<td>Password is too long.</td>
<td>Enter the password that is less than 16 characters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1048</td>
<td>Export Controller Status failed.</td>
<td>1. Retry using the same USB memory.</td>
<td>2. Retry using another USB memory.</td>
<td>3. Retry after rebooting the controller.</td>
</tr>
<tr>
<td>1049</td>
<td>Export Controller Status busy.</td>
<td>Execute the command after completing the controller status backup.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1050</td>
<td>Execution in Test mode is prohibited</td>
<td>Execute in other modes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>1054</td>
<td>Cannot execute in TEST mode except from TP.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1055</td>
<td>Cannot execute the Background Task.</td>
<td>Confirm that no background task is running.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>1056</td>
<td>Cannot execute from OP.</td>
<td>Enable the TP3 input.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1057</td>
<td>Cannot execute when TP3 enabled except from the TP3.</td>
<td>The command cannot be executed from other consoles when TP3 is enabled.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1058</td>
<td>Cannot execute excluding T2 mode.</td>
<td>Switch to &lt;Teach/T2&gt; key.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1059</td>
<td>Cannot change to T2 mode.</td>
<td>T2 mode cannot be used on RC700-A Controllers complying with UL</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>standards</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1100</td>
<td>File failure. Cannot access the file.</td>
<td>1. Reboot the controller.</td>
<td>2. Reinstall the firmware.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Replace the CF.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1102</td>
<td>File failure. Read and write failure of the registry</td>
<td>1. Reboot the controller.</td>
<td>2. Replace the CF.</td>
<td></td>
</tr>
<tr>
<td>1103</td>
<td>File is not found.</td>
<td>Check whether the file exists.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1104</td>
<td>Project file was not found.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1105</td>
<td>Object file was not found.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1106</td>
<td>Point files were not found.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1107</td>
<td>The program is using a feature that is not supported by the current</td>
<td>Check the compiler version in the EPSON RC+ 7.0-[Project]-[Properties]-[Compiler].</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>controller firmware version.</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1108</td>
<td>One or more source files are updated. Please build the project.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1109</td>
<td>Not enough storage capacity.</td>
<td>Increase free space of the USB memory</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1110</td>
<td>File is not found.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1111</td>
<td>Conveyor file was not found.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1112</td>
<td>Force files were not found.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1114</td>
<td>Cannot create the project in the controller.</td>
<td>There is a possibility that the folder is full or the project name is duplicated. Check it by the application selection.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1120</td>
<td>File failure. Setting file is corrupt.</td>
<td>Restore the controller configuration.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1121</td>
<td>File failure. Project file is corrupt.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1122</td>
<td>File failure. Point file is corrupt.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1123</td>
<td>File failure. I/O label file is corrupt.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1124</td>
<td>File failure. User error file is corrupt.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1126</td>
<td>File failure. Software option information is corrupt.</td>
<td>1. Reboot the controller.</td>
<td>2. Reinstall the firmware.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. Reconfigure the option.</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>1127</td>
<td>File failure. Vision file is corrupt.</td>
<td>Rebuild the project.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1128</td>
<td>File failure. Backup information file is corrupt.</td>
<td>The specified backup information cannot be restored. Acquire the backup information again, and then restore the file.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>1130</td>
<td>Error message failure. No item is found in the error history.</td>
<td>No error history exists. Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1131</td>
<td>Cannot access the USB memory.</td>
<td>Insert the USB memory properly. When this error still occurs after the USB memory is inserted properly, the memory may be unrecognizable to controller. Insert another memory to check the operation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1132</td>
<td>File failure. Failed to copy the file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1133</td>
<td>File failure. Failed to delete the file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1134</td>
<td>File failure. GUI Builder file is corrupt.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1138</td>
<td>File failure. Force Guide file is corrupt.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1140</td>
<td>File failure. Failed to open the object file.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1141</td>
<td>File failure. Failed to open the project file.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1142</td>
<td>File failure. Failed to read the project file.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1143</td>
<td>File failure. Failed to open the condition save file.</td>
<td>1. Retry using the same USB memory. 2. Retry using another USB memory. 3. Retry after rebooting the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1144</td>
<td>File failure. Failed to write the condition save file.</td>
<td>1. Retry using the same USB memory. 2. Retry using another USB memory. 3. Retry after rebooting the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1145</td>
<td>File failure. Failed to open the conveyor file.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1146</td>
<td>File failure. Failed to read the conveyor file.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1150</td>
<td>File failure. Error history is invalid.</td>
<td>1. Reboot the controller. 2. Replace the CF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1151</td>
<td>File failure. Failed to map the error history.</td>
<td>1. Reboot the controller. 2. Replace the CF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1152</td>
<td>File failure. Failed to open the error history file.</td>
<td>1. Reboot the controller. 2. Replace the CF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1153</td>
<td>File failure. Failed to write the error history file.</td>
<td>1. Reboot the controller. 2. Replace the CF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1155</td>
<td>File failure. Failed to open the settings file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1156</td>
<td>File failure. Failed to save the settings file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1157</td>
<td>File failure. Failed to read the settings file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1158</td>
<td>File failure. Failed to write the settings file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1160</td>
<td>MCD failure. Failed to open the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1161</td>
<td>MCD failure. Failed to read the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1163</td>
<td>MCD failure. Failed to save the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1165</td>
<td>MPD failure. Failed to open the MPD file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1166</td>
<td>MPD failure. Failed to read the MPD file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1168</td>
<td>MPD failure. Failed to save the MPD file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1170</td>
<td>MPL failure. Failed to open the MPL file.</td>
<td>1. Reboot the controller. 2. Reinstall the firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1181</td>
<td>PRM failure. Failed to replace the PRM file.</td>
<td>1. Reboot the controller. 2. Reconfigure the robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1185</td>
<td>File failure. Failed to open the backup information file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1186</td>
<td>File failure. Failed to read the backup information file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1187</td>
<td>File failure. Failed to write the backup information file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1188</td>
<td>File failure. Failed to save the backup information file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1189</td>
<td>The backup data was created by an old version.</td>
<td>Cannot restore the controller configuration in the specified procedure for using old backup data. Check the backup data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1190</td>
<td>The backup data was created by a newer version.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1191</td>
<td>There is no project in the backup data.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1192</td>
<td>Cannot execute with invalid robot number.</td>
<td>Check that the Backup data is same as current robot number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1193</td>
<td>Cannot execute with invalid robot information.</td>
<td>Check that the Backup data is same as current robot number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1194</td>
<td>Cannot execute with invalid drive unit number.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1195</td>
<td>File failure. Failed to map the health history.</td>
<td>Reboot the controller.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1196</td>
<td>File failure. Failed to open the health history file.</td>
<td>Reboot the controller.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1197</td>
<td>File failure. Failed to write the health history file.</td>
<td>Reboot the controller.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1198</td>
<td>Restore failure. The project is invalid. Restore the project together if restoring Vision hardware configuration.</td>
<td>Restore the project together if restoring Vision hardware configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1200</td>
<td>Compile failure. Check the compile message.</td>
<td>This error occurs during compilation from TP. Correct where the error occurred.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>1201</td>
<td>Link failure. Check the link message.</td>
<td>This error occurs during compilation from TP. Correct where the error occurred.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1250</td>
<td>User Outputs failure. The Name is empty.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1251</td>
<td>User Outputs failure. The Condition is empty.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1252</td>
<td>User Outputs failure. Robot number is out of the available range.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1260</td>
<td>Alarm Setting failure. Robot does not exist.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1261</td>
<td>Alarm Setting failure. Failed to get the expiration date.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1262</td>
<td>Alarm Setting failure. Failed to set the alarm.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1263</td>
<td>Alarm Setting failure. Specified alarm number is out of the allowable range.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1264</td>
<td>Alarm Setting failure. Specified alarm number is not enabled.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1290</td>
<td>Force monitor number is out of the allowable range. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1291</td>
<td>The force monitor is already used on another device. Close the force monitor on another device, and then run the force monitor.</td>
<td>Check whether another PC is using the force monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1292</td>
<td>Failed to set/load information of FG sequence and object.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1400</td>
<td>Force Guide file error.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1401</td>
<td>Invalid Force Guide file path.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1402</td>
<td>Failed to open Force Guide file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1403</td>
<td>Do not open Force Guide file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1404</td>
<td>Invalid Force Guide sequence number.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1405</td>
<td>Invalid Force Guide object number.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1406</td>
<td>Invalid Force Guide sequence property.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1407</td>
<td>Invalid Force Guide object property.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1408</td>
<td>Invalid Force Guide object type.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1409</td>
<td>Invalid parameter.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1410</td>
<td>Invalid Force Guide file version.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1413</td>
<td>Specified Force Guide object as ConditionObject is disabled. Confirm the settings.</td>
<td>Confirm the settings of Force Guide object.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1420</td>
<td>Failed to convert program. Execute rebuild.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1421</td>
<td>Initialization failure. Cannot allocate memory.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1422</td>
<td>Finalization failure. Failed to finalization of controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1423</td>
<td>Invalid conversion file path.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1424</td>
<td>Invalid Prg file path.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1425</td>
<td>Invalid Command file path.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1426</td>
<td>Invalid conversion file.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1427</td>
<td>Invalid Command file. Update from Ver.7.4.0.2 or earlier to Ver7.4.0.2 or later may fail. Reinstall the firmware.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1428</td>
<td>Failed to execute conversion of program. Execute rebuild. Rebuild the project.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1429</td>
<td>Failed to write Prg file.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>Communication error.</td>
<td>Execute the command again after a while. Check the connection between the EPSON RC+7.0 and controller.</td>
<td>1: Communication timeout 2: USB cable disconnection 3: USB reception failure 4: USB communication shutdown</td>
<td></td>
</tr>
<tr>
<td>1501</td>
<td>Command did not complete in time.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1502</td>
<td>Communication disconnection between RC+ and Controller. Re-establish communication. Check the connection between the EPSON RC+7.0 and controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1503</td>
<td>Disconnection while executing a task. Check the connection between the console device and controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1504</td>
<td>Communication disconnection between Remote Ethernet and Controller. Re-establish communication. Check the connection between the Remote Ethernet device and controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1505</td>
<td>Communication disconnection between Remote RS232 and Controller. Re-establish communication. Check the connection between the Remote RS232 device and controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1506</td>
<td>Communication disconnection between TP3 and Controller. Re-establish communication. Check the connection between TP3 and controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1510</td>
<td>Out of IP Address range. Check the IP address setting of the controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1511</td>
<td>Reserved IP Address. The IP address is reserved. Set the other IP address.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1512</td>
<td>Reserved IP Gateway. The gateway address is reserved. Set the other gateway address.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1513</td>
<td>The setting conditions of global address do not match. Set the password.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>1514</td>
<td>The connection parameters or password are invalid. Check the parameters and the password.</td>
<td>Check the parameters and the password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1521</td>
<td>Vision communication. Failed to initialize Ethernet.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1522</td>
<td>Vision communication. Failed to terminate Ethernet.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1523</td>
<td>Vision communication. Failed to create the socket handle.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1524</td>
<td>Vision communication. Failed to connect.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1526</td>
<td>Vision communication. Failed to send to the server.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1527</td>
<td>Vision communication. Failed to read from the server.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1528</td>
<td>Vision communication. Failed to set option.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1529</td>
<td>Vision communication. Ethernet has not been initialized yet.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1530</td>
<td>Vision communication. Connection is not completed.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1531</td>
<td>Vision communication. All sockets are used.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1532</td>
<td>Vision communication. Sending time-out.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1533</td>
<td>Vision communication. Receiving time-out.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1534</td>
<td>Vision communication. Communication error.</td>
<td>Check the connection between the camera and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1550</td>
<td>Communication failure. Ethernet initialization error.</td>
<td>Reboot the controller. Check the connection of the Ethernet cable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1551</td>
<td>Communication failure. USB initialization error.</td>
<td>Reboot the controller. Check the connection of the USB cable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1552</td>
<td>Communication failure. Controller internal communication error.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1553</td>
<td>Communication failure. Invalid data is detected.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1555</td>
<td>Ethernet transmission error.</td>
<td>Check the connection between the EPSON RC+7.0 and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1556</td>
<td>Ethernet reception error.</td>
<td>Check the connection between the EPSON RC+7.0 and controller. If the router is used between the PC and controller, confirm that the DHCP function is disabled.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1557</td>
<td>USB transmission error.</td>
<td>Check the connection between the EPSON RC+7.0 and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1558</td>
<td>USB reception error.</td>
<td>Check the connection between the EPSON RC+7.0 and controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1559</td>
<td>Communication failure. Failed to allocate memory</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1580</td>
<td>Parser communication failure. Communication error.</td>
<td>1. Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Upgrade the firmware.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
</table>
| 1581 | Parser communication failure. Time-out occurred during communication. | 1. Reboot the controller.  
2. Reinstall the firmware. | | |
| 1582 | Parser communication failure. Transmission error. | Reboot the controller.  
Rebuild the project. | | |
| 1583 | Parser communication failure. Initialization error. | Reboot the controller. | | |
| 1584 | Parser communication failure. Connection error. | Reboot the controller. | | |
| 1585 | Parser communication failure. Parameter is invalid. | Reboot the controller.  
Rebuild the project. | | |
| 1586 | Parser communication failure. Busy | - | | |
| 1587 | Parser communication failure. Invalid data is detected. | Upgrade the firmware. | | |
| 1700 | Initialization failure. Failed to initialize TP. | - | | |
| 1701 | Initialization failure. Failed to initialize TP. | - | | |
| 1702 | Initialization failure. Failed to initialize TP. | - | | |
| 1703 | File failure. Failed to read the screen data file. | - | | |
| 1704 | Failed to read the setting file. | - | | |
| 1706 | Failed to open the TP port. | - | | |
| 1708 | Failed to read the key table for TP. | - | | |
| 1709 | Failed to change the language. | - | | |
| 1710 | Failed to display the screen. | - | | |
| 1800 | The controller is already connected to RC+. | Only one RC+ 7.0 can be connected to the controller. | | |
| 1802 | The command was attempted without being connected to a controller. | Connect to the controller. | | |
| 1803 | Failed to read or write the file on the PC. | - | | |
| 1804 | Initialization failure. Failed to allocate memory on the PC. | - | | |
| 1805 | Connection failure. Check the controller startup and connection of the communication cable. | - | | |
| 1806 | Timeout during connection via Ethernet. | - | | |
| 1807 | Timeout during connection via USB. | - | | |
| 1808 | USB driver is not installed. | Failed to install EPSON RC+ 7.0.  
Install EPSON RC+ 7.0 again. | | |
| 1809 | Initialization failure. Failed to initialize PC daemon. | Reboot the System. | | |
| 1810 | PC daemon error. Uncommon error. | 1. Reboot the EPSON RC+7.0.  
2. Reboot the PC. | | |
<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1812</td>
<td>Connection failure. The connected controller is not supported in EPSON RC+ 7.0. Please use EPSON RC+ 5.0.</td>
<td>Connected controller is RC180 or RC90 compatible with EPSON RC+ 5.0. Check the connection between the PC and controller.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1852</td>
<td>System error. Uncommon error.</td>
<td>1. Reboot the EPSON RC+ 7.0.</td>
<td>2. Reboot the PC.</td>
<td>3. Reinstall the EPSON RC+ 7.0.</td>
</tr>
<tr>
<td>1861</td>
<td>Initialization failure. Failed to initialize SimulatorMNG.</td>
<td>1. Reboot the EPSON RC+ 7.0.</td>
<td>2. Reboot the PC.</td>
<td>3. Reinstall the EPSON RC+ 7.0.</td>
</tr>
<tr>
<td>1862</td>
<td>Initialization failure. Failed to initialize WBProxy.</td>
<td>1. Reboot the EPSON RC+ 7.0.</td>
<td>2. Reboot the PC.</td>
<td>3. Reinstall the EPSON RC+ 7.0.</td>
</tr>
<tr>
<td>1863</td>
<td>The parameter is invalid.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1864</td>
<td>Initialization failure. Virtual controller does not exist.</td>
<td>Installation of the EPSON RC+ 7.0 failed. Reinstall the software.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1865</td>
<td>Initialization failure. Failed to start virtual controller.</td>
<td>1. Retry after a while.</td>
<td>2. Reboot the PC.</td>
<td>-</td>
</tr>
<tr>
<td>1867</td>
<td>Cannot execute because it is not dry run mode.</td>
<td>Dry run mode is invalid. Enable the dry run.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1868</td>
<td>Initialization failure. Directory cannot be found.</td>
<td>Installation of the EPSON RC+ 7.0 failed. Reinstall the software.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1872</td>
<td>Connection failure. Files for simulator that used real controller cannot be found.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1873</td>
<td>Connection failure. Files for simulator that used virtual controller cannot be found.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1874</td>
<td>Virtual Controller cannot be added.</td>
<td>Installation of the EPSON RC+ 7.0 failed. Reinstall the software.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1875</td>
<td>Simulator Object failure. Cannot load data for the simulator object.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1876</td>
<td>Simulator Object failure. Cannot read data for the simulator object.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1877</td>
<td>Simulator Object failure. Cannot remove data from the simulator object.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1878</td>
<td>Simulator Object failure. Cannot update data for the simulator object.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1879</td>
<td>Other virtual controllers are starting.</td>
<td>Other virtual controllers may be used in the EPSON RC+ 5.0. Or, the virtual controller may be already used in another EPSON RC+ 7.0.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1880</td>
<td>Cannot execute during controller reset.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1901</td>
<td>Unsupported. Unsupported command was attempted.</td>
<td>Update the firmware.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1902</td>
<td>Unsupported. Unsupported parameter was specified.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1903</td>
<td>System error.</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>1910</td>
<td>System error. Failed to write the reboot file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>2000</td>
<td>Unsupported. Unsupported command was attempted.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2001</td>
<td>Unsupported. Unsupported motion command was attempted.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td>Unsupported. Unsupported Function argument was specified.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>Unsupported. Unsupported Function return value was specified.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Unsupported. Unsupported condition was specified.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2006</td>
<td>Unsupported. Unsupported I/O command was specified.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2007</td>
<td>Unsupported condition was specified.</td>
<td>Cannot jog in the CP motion (default).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2008</td>
<td>Unsupported. Unknown error number.</td>
<td>Clicking the same jog button will operate the robot in the PTP motion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td>Object file error. Build the project. Out of internal code range.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2011</td>
<td>Object file error. Build the project. Function argument error.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>Object file error. Build the project. Command argument error.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>Object file error. Build the project. Cannot process the code.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td>Object file error. Build the project. Cannot process the variable type code.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2015</td>
<td>Object file error. Build the project. Cannot process the string type code.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td>Object file error. Build the project. Cannot process the variable category code.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>Object file error. Build the project. Cannot process because of improper code.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2018</td>
<td>Object file error. Build the project. Failed to calculate the variable size.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2019</td>
<td>Object file error. Cannot process the variable wait. Build the project.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2020</td>
<td>Stack table number exceeded.</td>
<td>Check whether no function is called infinitely. Reduce the Call function depth.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>2021</td>
<td>Stack area size exceeded. Stack error. Function call or local variable is out of range.</td>
<td>If using many local variables, especially String type, replace them to global variables.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2022</td>
<td>Stack failure. Required data not found on the stack.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2023</td>
<td>Stack failure. Unexpected tag found on the stack.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2024</td>
<td>Stack area size exceeded. Local variable is out of range.</td>
<td>Change the size of the Local variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2031</td>
<td>System failure. Robot number is beyond the maximum count.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2032</td>
<td>System failure. Task number compliance error.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2033</td>
<td>System failure. Too many errors.</td>
<td>Remedy the errors occurring frequently.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2040</td>
<td>Thread failure. Failed to create the thread.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2041</td>
<td>Thread failure. Thread creation timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2042</td>
<td>Thread failure. Thread termination timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2043</td>
<td>Thread failure. Thread termination timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2044</td>
<td>Thread failure. Daemon process timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2045</td>
<td>Thread failure. Task continuance wait timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2046</td>
<td>Thread failure. Task stop wait timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2047</td>
<td>Thread failure. Task startup wait timeout.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2050</td>
<td>Object file operation failure. Object file size is beyond the allowable size.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2051</td>
<td>Object file operation failure. Cannot delete the object file during execution.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2052</td>
<td>Object file operation failure. Cannot allocate the memory for the object file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2053</td>
<td>Object file operation failure. Object file cannot be accessed while it is updating.</td>
<td>Perform the same processing after a while. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2054</td>
<td>Object file operation failure. Function ID failure. Rebuild the project.</td>
<td>Synchronize the files of the project. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2055</td>
<td>Object file operation failure. Local variable ID failure. Rebuild the project.</td>
<td>Synchronize the files of the project. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2056</td>
<td>Object file operation failure. Global variable ID failure. Rebuild the project.</td>
<td>Synchronize the files of the project. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>-------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>2057</td>
<td>Object file operation failure. Global Preserve variable ID failure.</td>
<td>Synchronize the files of the project.</td>
<td></td>
<td>Rebuild the project.</td>
</tr>
<tr>
<td></td>
<td>Failed to calculate the variable size.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2058</td>
<td>Object file operation failure.</td>
<td>Synchronize the files of the project.</td>
<td></td>
<td>Rebuild the project.</td>
</tr>
<tr>
<td></td>
<td>Failed to calculate the variable size.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2059</td>
<td>Exceed the global variable area.</td>
<td>Reduce the number of Global preserve variables to be used.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cannot assign the Global variable area because it failed to allocate memory.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2070</td>
<td>SRAM failure. SRAM is not mapped.</td>
<td>Replace the CPU board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2071</td>
<td>SRAM failure. Cannot delete when Global Preserve variable is in use.</td>
<td>Perform the same processing after a while. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2072</td>
<td>Exceed the backup variable area.</td>
<td>Reduce the number of Global preserve variables to be used.</td>
<td>Maximum size</td>
<td>The size you attempted to use</td>
</tr>
<tr>
<td></td>
<td>Cannot assign the Global Preserve variable area because it failed to allocate memory.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2073</td>
<td>SRAM failure. Failed to clear the Global Preserve variable area.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2074</td>
<td>SRAM failure. Failed to clean up the Global Preserve variable save area.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2100</td>
<td>Initialization failure. Failed to open the initialization file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2101</td>
<td>Initialization failure. Duplicated initialization.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2102</td>
<td>Initialization failure. Failed to initialize MNG.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2103</td>
<td>Initialization failure. Failed to create an event.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2104</td>
<td>Initialization failure. Failed to setup a priority.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2105</td>
<td>Initialization failure. Failed to setup the stack size.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2106</td>
<td>Initialization failure. Failed to setup an interrupt process.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2107</td>
<td>Initialization failure. Failed to start an interrupt process.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2108</td>
<td>Initialization failure. Failed to stop an interrupt process.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2109</td>
<td>Initialization failure. Failed to terminate MNG.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2110</td>
<td>Initialization failure. Failed to allocate memory.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2111</td>
<td>Initialization failure. Failed to initialize motion.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2112</td>
<td>Initialization failure. Failed to terminate motion.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2113</td>
<td>Initialization failure. Failed to map SRAM.</td>
<td>Replace the CPU board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2114</td>
<td>Initialization failure. Failed to register SRAM.</td>
<td>Replace the CPU board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2115</td>
<td>Initialization failure. Fieldbus board is beyond the maximum count.</td>
<td>Check the number of fieldbus boards.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2116</td>
<td>Initialization failure. Failed to initialize fieldbus.</td>
<td>Reboot the controller.</td>
<td>Check the fieldbus board.</td>
<td>Replace the fieldbus board.</td>
</tr>
<tr>
<td>2117</td>
<td>Initialization failure. Failed to terminate fieldbus.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2118</td>
<td>Initialization failure. Failed to open motion.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2119</td>
<td>Initialization failure. Failed to initialize conveyor tracking.</td>
<td>Make sure the settings of conveyor and encoder are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2120</td>
<td>Initialization failure. Failed to allocate the system area.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2121</td>
<td>Initialization failure. Failed to allocate the object file area.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2122</td>
<td>Initialization failure. Failed to allocate the robot area.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2123</td>
<td>Initialization failure. Failed to create event.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2124</td>
<td>Initialization failure. An unsupported Fieldbus module is installed.</td>
<td>Install the Fieldbus module purchased from SEC.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2126</td>
<td>The settings are initialized since the initial setting file is corrupted. Restored the system.</td>
<td>Since the initial setting file has significantly corrupted, the initial setting file of factory-default is used.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2130</td>
<td>MCD failure. Failed to open the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2131</td>
<td>MCD failure. Failed to map the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2132</td>
<td>PRM failure. PRM file cannot be found.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2133</td>
<td>PRM failure. Failed to map the PRM file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2134</td>
<td>PRM failure. PRM file contents error.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2135</td>
<td>PRM failure. Failed to convert the PRM file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2136</td>
<td>PRM failure. Failed to convert the PRM file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2137</td>
<td>PRM failure. Failed to convert the PRM file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2140</td>
<td>DU Initialization Error. Cannot use drive units.</td>
<td>Communication with drive units is not available for the virtual controllers. Return the configuration file to original setting if it was changed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2141</td>
<td>DU Initialization Error. Failed to initialize drive units.</td>
<td>Check the connection with drive units.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2142</td>
<td>DU Initialization Error. Failed to initialize drive units.</td>
<td>Check the connection with drive units.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2143</td>
<td>DU Initialization Error. Timeout during initialization of drive units.</td>
<td>Check the connection with drive units.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2144</td>
<td>DU Initialization Error. No data to download to drive units.</td>
<td>Reboot the control unit and drive units.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2145</td>
<td>DU Initialization Error. Failed to start communication with drive units.</td>
<td>Reboot the control unit and drive units.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2146</td>
<td>DU Initialization Error. Timeout when starting communication with drive units.</td>
<td>Reboot the control unit and drive units.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2147</td>
<td>DU Initialization Error. Failed to update the drive units software.</td>
<td>Review the software update setting. Check the connection with the Drive Unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2148</td>
<td>DU Initialization Error. Failed to update the drive units software.</td>
<td>Check the file name. Check the update file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2149</td>
<td>DU Initialization Error. Failed to update the drive units software.</td>
<td>Check the Drive Unit power and connection. Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2150</td>
<td>Operation failure. Task number cannot be found.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2151</td>
<td>Operation failure. Executing the task.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2152</td>
<td>Operation failure. Object code size failure.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2154</td>
<td>Operation failure. Executing jog.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2155</td>
<td>Operation failure. Cannot execute the jog function.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2156</td>
<td>Operation failure. Jog data is not configured.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2157</td>
<td>Operation failure. Failed to change the jog parameter.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2158</td>
<td>Operation failure. Failed to allocate the area for the break point.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2159</td>
<td>Operation failure. Break point number is beyond the allowable setup count.</td>
<td>Reduce the break points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2160</td>
<td>Operation failure. Failed to allocate the function ID.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2161</td>
<td>Operation failure. Failed to allocate the local variable address.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2162</td>
<td>Operation failure. Not enough buffer to store the local variable.</td>
<td>Review the size of the Local variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2163</td>
<td>Operation failure. Value change is available only when the task is halted.</td>
<td>Halt the task by the break point.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2164</td>
<td>Operation failure. Failed to allocate the global variable address.</td>
<td>Review the size of the global variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2165</td>
<td>Operation failure. Not enough buffer to store the global variable.</td>
<td>Review the size of the global variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2166</td>
<td>Operation failure. Failed to obtain the Global Preserve variable address.</td>
<td>Review the size of the global preserve variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2167</td>
<td>Operation failure. Not enough buffer to store the Global Preserve variable.</td>
<td>Review the size of the global preserve variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2168</td>
<td>Operation failure. SRAM is not mapped.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2169</td>
<td>Operation failure. Cannot clear the Global Preserve variable when loading the object file.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2170</td>
<td>Operation failure. Not enough buffer to store the string.</td>
<td>Check the size of the string variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2171</td>
<td>Operation failure. Cannot start the task after low voltage was detected.</td>
<td>Check the controller power. Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2172</td>
<td>Operation failure. Duplicated remote I/O configuration.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2173</td>
<td>Remote setup error. Cannot assign non-existing input number to remote function.</td>
<td>Check the I/O input number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2174</td>
<td>Remote setup error. Cannot assign non-existing output number to remote function.</td>
<td>Check the I/O output number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2175</td>
<td>Operation failure. Remote function is not configured.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2176</td>
<td>Operation failure. Event wait error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2179</td>
<td>Remote setup error. Cannot assign same input number to some remote functions.</td>
<td>Check the remote setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2180</td>
<td>Remote setup error. Cannot assign same output number to some remote functions.</td>
<td>Check the remote setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2181</td>
<td>Operation failure. Task number has not been reserved for RC+ API.</td>
<td>Set the number of RC+API tasks.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2190</td>
<td>Cannot calculate because it was queue data.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2191</td>
<td>Cannot execute AbortMotion because robot is not running from a task.</td>
<td>If you don’t operate the robot from a program, you cannot use AbortMotion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2192</td>
<td>Cannot execute AbortMotion because robot task is already finished.</td>
<td>Task is completed. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2193</td>
<td>Cannot execute Recover without motion because AbortMotion was not executed.</td>
<td>Execute AbortMotion in advance to execute Recover WithoutMove.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2194</td>
<td>Conveyor setting error.</td>
<td>Make sure the settings of conveyor and encoder are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2195</td>
<td>Conveyor setting error.</td>
<td>Make sure the settings of conveyor and encoder are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2196</td>
<td>Conveyor number is out of range.</td>
<td>Make sure the settings of conveyor and encoder are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2197</td>
<td>Command parameter prohibited for conveyor tracking motion was used.</td>
<td>Delete LJM.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2200</td>
<td>Robot in use. Cannot execute the motion command when other tasks are using the robot.</td>
<td>The motion command for the robot cannot be simultaneously executed from more than one task. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2201</td>
<td>Robot does not exist.</td>
<td>Check whether the robot setting is performed properly. Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2202</td>
<td>Motion control module status failure. Unknown error was returned.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2203</td>
<td>Cannot clear local number ' 0 '.</td>
<td>The Local number 0 cannot be cleared. Review the program.</td>
<td>The Arm number you attempted to clear</td>
<td></td>
</tr>
<tr>
<td>2204</td>
<td>Cannot clear an arm while in use.</td>
<td>The Arm cannot be cleared while it is in use. Check whether the Arm is not used.</td>
<td></td>
<td>The Arm number you attempted to clear</td>
</tr>
<tr>
<td>2205</td>
<td>Cannot clear arm number ' 0 '.</td>
<td>The Arm number 0 cannot be cleared. Review the program.</td>
<td></td>
<td>The Tool number you attempted to clear</td>
</tr>
<tr>
<td>2206</td>
<td>Cannot clear a tool while in use.</td>
<td>The Tool cannot be cleared while it is in use. Check whether the Tool is not used.</td>
<td></td>
<td>The Tool number you attempted to clear</td>
</tr>
<tr>
<td>2207</td>
<td>Cannot clear tool number ' 0 '.</td>
<td>The Tool number 0 cannot be cleared. Review the program.</td>
<td></td>
<td>The ECP number you attempted to clear</td>
</tr>
<tr>
<td>2208</td>
<td>Cannot clear ECP ' 0 '.</td>
<td>The ECP number 0 cannot be cleared. Review the program.</td>
<td></td>
<td>The ECP number you attempted to clear</td>
</tr>
<tr>
<td>2209</td>
<td>Cannot clear an ECP while in use.</td>
<td>The ECP cannot be cleared while it is in use. Check whether the ECP is not used.</td>
<td></td>
<td>The ECP number you attempted to clear</td>
</tr>
<tr>
<td>2210</td>
<td>Cannot specify ' 0 ' as the local number.</td>
<td>The command processing the Local cannot specify the Local number 0. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2216</td>
<td>Box number is out of range.</td>
<td>Available Box numbers are from 1 to 15. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2217</td>
<td>Box number is not defined.</td>
<td>Specified Box is not defined. Review the Box number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2218</td>
<td>Plane number is out of range.</td>
<td>Available Box numbers are from 1 to 15. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2219</td>
<td>Plane number is not defined.</td>
<td>Specified Plane is not defined. Review the Plane number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2220</td>
<td>PRM failure. No PRM file data is found.</td>
<td>Reboot the controller. Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>2221</td>
<td>PRM failure. Failed to flash the PRM file.</td>
<td>Reboot the controller. Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2222</td>
<td>Local number is not defined.</td>
<td>Check the Local setting. Review the program.</td>
<td>The specified Local number</td>
<td></td>
</tr>
<tr>
<td>2223</td>
<td>Local number is out of range.</td>
<td>Available Local number is from 1 to 15. Review the program.</td>
<td>The specified Local number</td>
<td></td>
</tr>
<tr>
<td>2224</td>
<td>Unsupported. MCOFS is not defined.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2225</td>
<td>CalPls is not defined.</td>
<td>Check the CalPls setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2226</td>
<td>Arm number is out of range.</td>
<td>Available Arm number is from 0 to 3. Depending on commands, the Arm number 0 is not available. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2227</td>
<td>Arm number is not defined.</td>
<td>Check the Arm setting. Review the program.</td>
<td>The specified Arm number</td>
<td></td>
</tr>
<tr>
<td>2228</td>
<td>Pulse for the home position is not defined.</td>
<td>Check the HomeSet setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2229</td>
<td>Tool number is out of range.</td>
<td>Available Tool number is from 0 to 3. Depending on commands, the Tool number 0 is not available. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2230</td>
<td>Tool number is not defined.</td>
<td>Check the Tool setting. Review the program.</td>
<td>The specified Tool number</td>
<td></td>
</tr>
<tr>
<td>2231</td>
<td>ECP number is out of range.</td>
<td>Available Tool number is from 0 to 15. Depending on commands, the Tool number 0 is not available. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2232</td>
<td>ECP number is not defined.</td>
<td>Check the ECP setting. Review the program.</td>
<td>The specified ECP number</td>
<td></td>
</tr>
<tr>
<td>2233</td>
<td>Axis to reset the encoder was not specified.</td>
<td>Be sure to specify the axis for encoder reset.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2234</td>
<td>Cannot reset the encoder with motor in the on state.</td>
<td>Turn the motor power OFF before reset.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2235</td>
<td>XYLIM is not defined.</td>
<td>Check the XYLim setting. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2236</td>
<td>PRM failure. Failed to set up the PRM file contents to the motion control status module.</td>
<td>Reboot the controller. Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2237</td>
<td>Pallet number is out of range.</td>
<td>Available Pallet numbers are from 0 to 15. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2238</td>
<td>Pallet is not defined.</td>
<td>Check the Pallet setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2240</td>
<td>Array subscript is out of user defined range. Cannot access or update beyond array bounds.</td>
<td>Check the array subscript. Review the program.</td>
<td></td>
<td>The specified subscript</td>
</tr>
<tr>
<td>2241</td>
<td>Dimensions of array do not match the declaration.</td>
<td>Check the array's dimensions. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2242</td>
<td>Zero '0' was used as a divisor.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2243</td>
<td>Variable overflow. Specified variable was beyond the maximum allowed value.</td>
<td>Check the variable type and calculation result. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2244</td>
<td>Variable underflow. Specified variable was below the minimum allowed value.</td>
<td>Check the variable type and calculation result. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
<td>------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>2245</td>
<td>Cannot execute this command with a floating point number.</td>
<td>This command cannot be executed for Real or Double type. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2246</td>
<td>Cannot calculate the specified value using the Tan function.</td>
<td>Check the specified value. Review the program.</td>
<td>The specified value</td>
<td></td>
</tr>
<tr>
<td>2247</td>
<td>Specified array subscript is less than ' 0 '.</td>
<td>Check the specified value. Review the program.</td>
<td>The specified value</td>
<td></td>
</tr>
<tr>
<td>2248</td>
<td>Array failure. Redim can only be executed for an array variable.</td>
<td>You attempted to Redim the variable that is not array. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2249</td>
<td>Array failure. Cannot specify Preserve for other than a single dimension array.</td>
<td>Other than a single dimension array was specified as Preserve for Redim. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2250</td>
<td>Array failure. Failed to calculate the size of the variable area.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2251</td>
<td>Cannot allocate enough memory for Redim statement.</td>
<td>Reduce the number of subscripts to be specified for Redim. Perform Redim modestly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2252</td>
<td>Cannot allocate enough memory for ByRef.</td>
<td>Reduce the number of array's subscripts to be seen by ByRef.</td>
<td>The number of array subsripts</td>
<td></td>
</tr>
<tr>
<td>2253</td>
<td>Cannot compare characters with values.</td>
<td>Check whether the string type and the numeric data type are not compared. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2254</td>
<td>Specified data is beyond the array bounds. Cannot refer or update beyond the array bounds.</td>
<td>Check the number of array's subscripts and data. Review the program.</td>
<td>The number of array subscripts</td>
<td>The number of data to be referred or updated</td>
</tr>
<tr>
<td>2255</td>
<td>Variable overflow or underflow. Specified variable is out of value range.</td>
<td>The value that exceeds the range of Double type is specified. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2256</td>
<td>Specified array subscript is beyond the maximum allowed range.</td>
<td>Reduce the number of subscripts to be specified. For available subscripts, see the online help.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2257</td>
<td>Cannot specify Int64 variable or UInt64 variable.</td>
<td>Int64 variable or UInt64 variable cannot be specified. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2258</td>
<td>Task number is out of the available range.</td>
<td>For available task number, see the online help. Review the program.</td>
<td>The specified task number</td>
<td></td>
</tr>
<tr>
<td>2259</td>
<td>Specified task number does not exist.</td>
<td>Review the program.</td>
<td>The specified task number</td>
<td></td>
</tr>
<tr>
<td>2260</td>
<td>Robot number is out of the available range.</td>
<td>The available Robot number is 1. Review the program.</td>
<td>The specified robot number</td>
<td></td>
</tr>
<tr>
<td>2261</td>
<td>Output number is out of the available range. The Port No. or the Device No. is out of the available range.</td>
<td>For available output number, see the online help. Review the program.</td>
<td>The specified output number</td>
<td></td>
</tr>
<tr>
<td>2262</td>
<td>Command argument is out of the available range. Check the arguments. Added data 1: Passed value. Added data 2: argument order.</td>
<td>For available range of argument, see the online help. Review the program.</td>
<td>The Added value</td>
<td>What number argument?</td>
</tr>
<tr>
<td>2263</td>
<td>Joint number is out of the available range.</td>
<td>Available Joint number is from 1 to 6. Review the program.</td>
<td>The specified joint number</td>
<td></td>
</tr>
<tr>
<td>2264</td>
<td>Wait time is out of available range.</td>
<td>Available wait time is from 0 to 2147483. Review the program.</td>
<td>The specified wait time</td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>2267</td>
<td>Timer number is out of available range.</td>
<td>Available timer number is from 0 to 15. Review the program.</td>
<td>The specified timer number</td>
<td></td>
</tr>
<tr>
<td>2268</td>
<td>Trap number is out of available range.</td>
<td>Available trap number is from 1 to 4. Review the program.</td>
<td>The specified trap number</td>
<td></td>
</tr>
<tr>
<td>2269</td>
<td>Language ID is out of available range.</td>
<td>For available language ID, see the online help. Review the program.</td>
<td>The specified language ID</td>
<td></td>
</tr>
<tr>
<td>2270</td>
<td>Specified D parameter value for the parallel process is out of available range.</td>
<td>Available D parameter value is from 0 to 100. Review the program.</td>
<td>The specified D parameter value</td>
<td></td>
</tr>
<tr>
<td>2271</td>
<td>Arch number is out of available range.</td>
<td>Available arch number is from 0 to 7. Review the program.</td>
<td>The specified arch number</td>
<td></td>
</tr>
<tr>
<td>2272</td>
<td>Device No. is out of available range.</td>
<td>The specified number representing a control device or display device is out of available range. For available device number, see the online help. Review the program.</td>
<td>The specified device number</td>
<td></td>
</tr>
<tr>
<td>2273</td>
<td>Output data is out of available range.</td>
<td>Available output data value is from 0 to 255. Review the program.</td>
<td>Output data</td>
<td>What number byte data is out of range?</td>
</tr>
<tr>
<td>2274</td>
<td>Asin argument is out of available range. Range is from -1 to 1.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2275</td>
<td>Acos argument is out of available range. Range is from -1 to 1.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2276</td>
<td>Sqr argument is out of available range.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2277</td>
<td>Randomize argument is out of available range.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2278</td>
<td>Sin, Cos, Tan argument is out of available range.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2280</td>
<td>Timeout period set by the TMOOut statement expired before the wait condition was completed in the WAIT statement.</td>
<td>Investigate the cause of timeout. Check whether the set timeout period is proper.</td>
<td>Timeout period</td>
<td></td>
</tr>
<tr>
<td>2281</td>
<td>Timeout period set by TMOOut statement in WaitSig statement or SyncLock statement expired.</td>
<td>Investigate the cause of timeout. Check whether the set timeout period is proper.</td>
<td>Signal number Timeout period</td>
<td></td>
</tr>
<tr>
<td>2282</td>
<td>Timeout period set by TMOOut statement in WaitNet statement expired.</td>
<td>Investigate the cause of timeout. Check whether the set timeout period is proper.</td>
<td>Port number Timeout period</td>
<td></td>
</tr>
<tr>
<td>2283</td>
<td>Timeout. Timeout at display device setting.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2290</td>
<td>Cannot execute a motion command.</td>
<td>Cannot execute the motion command after using the user function in the motion command. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2291</td>
<td>Cannot execute the OnErr command.</td>
<td>Cannot execute OnErr in the motion command when using user function in the motion command. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2292</td>
<td>Cannot execute an I/O command while the safeguard is open. Need Forced.</td>
<td>I/O command cannot be executed while the safeguard is open. Review the program</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2293</td>
<td>Cannot execute an I/O command during emergency stop condition. Need Forced.</td>
<td>I/O command cannot be executed during emergency stop condition. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>2294</td>
<td>Cannot execute an I/O command when an error has been detected. Need Forced.</td>
<td>I/O command cannot be executed while an error occurs. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2295</td>
<td>Cannot execute this command from a NoEmgAbort Task and Background Task.</td>
<td>For details on in executable commands, refer to the online help. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2296</td>
<td>One or more source files are updated. Please build the project.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2297</td>
<td>Cannot execute an I/O command in TEACH mode without the Forced parameter.</td>
<td>I/O command cannot be executed in TEACH mode. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2298</td>
<td>Cannot continue execution in Trap SGCLOSE process.</td>
<td>You cannot execute Cont and Recover statements with processing task of Trap SGCLOSE.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2299</td>
<td>Cannot execute this command. Need the setting [enable the advance task control commands] from RC+ controller preference settings.</td>
<td>Enable the [enable the advance task control commands] from RC+ to execute the command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2300</td>
<td>Robot in use. Cannot execute the motion command when other task is using the robot.</td>
<td>The motion command for the robot cannot be simultaneously executed from more than one task. Review the program.</td>
<td>Task number that is using the robot</td>
<td></td>
</tr>
<tr>
<td>2301</td>
<td>Cannot execute the motion command when the Enable Switch is OFF.</td>
<td>Execute the motion command with the enable switch gripped.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2302</td>
<td>Cannot execute a Call statement in a Trap Call process.</td>
<td>Another function cannot be called from the function called by Trap Call. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2303</td>
<td>Cannot execute a Call statement in a parallel process.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2304</td>
<td>Cannot execute an Xqt statement in a parallel process.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2305</td>
<td>Cannot execute a Call statement from the command window.</td>
<td>Execute Call from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2306</td>
<td>Cannot execute an Xqt statement from the task started by Trap Xqt.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2307</td>
<td>Cannot execute this command while tasks are executing.</td>
<td>Check whether all tasks are completed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2308</td>
<td>Cannot turn on the motor because of a critical error.</td>
<td>Find the previously occurring error in the error history and resolve its cause. Then, reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2309</td>
<td>Cannot execute a motion command while the safeguard is open.</td>
<td>Check the safeguard status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2310</td>
<td>Cannot execute a motion command while waiting for continue.</td>
<td>Execute the Continue or Stop and then execute the motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2311</td>
<td>Cannot execute a motion command during the continue process.</td>
<td>Wait until the Continue is complete and then execute the motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2312</td>
<td>Cannot execute a task during emergency stop condition.</td>
<td>Check the emergency stop status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2313</td>
<td>Cannot continue execution immediately after opening the safeguard.</td>
<td>Wait 1.5 seconds after the safeguard is open, and then execute the Continue.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2314</td>
<td>Cannot continue execution while the safeguard is open.</td>
<td>Check the safeguard status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2315</td>
<td>Cannot execute Cont and Restart command in resume operation.</td>
<td>Wait until the Continue is completed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2316</td>
<td>Cannot continue execution after an error has been detected.</td>
<td>Check the error status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2317</td>
<td>Cannot execute the task when an error has been detected.</td>
<td>Reset the error by Reset and then execute the task.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2318</td>
<td>Cannot execute a motion command when an error has been detected.</td>
<td>Execute the motion command after resetting the error by Reset.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2319</td>
<td>Cannot execute an I/O command during emergency stop condition.</td>
<td>Check the emergency stop status.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2320</td>
<td>Function failure. Argument type does not match.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2321</td>
<td>Function failure. Return value does not match to the function.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2322</td>
<td>Function failure. ByRef type does not match.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2323</td>
<td>Function failure. Failed to process the ByRef parameter.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2324</td>
<td>Function failure. Dimension of the ByRef parameter does not match.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2325</td>
<td>Function failure. Cannot use ByRef in an Xqt statement.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2326</td>
<td>Cannot execute a Dll Call statement from the command window.</td>
<td>Execute Dll Call from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2327</td>
<td>Failed to execute a Dll Call.</td>
<td>Check the DLL. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2328</td>
<td>Cannot execute the task before connection with RC+.</td>
<td>You need to connect with RC+ before executing the task.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2329</td>
<td>Cannot execute an Eval statement in a Trap Call process.</td>
<td>Check the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2330</td>
<td>Trap failure. Cannot use the argument in Trap Call or Xqt statement.</td>
<td>Check the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2331</td>
<td>Trap failure. Failed to process Trap Goto statement.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2332</td>
<td>Trap failure. Failed to process Trap Goto statement.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2333</td>
<td>Trap failure. Trap is already in process.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2334</td>
<td>Cannot execute an Eval statement in a Trap Finish or a Trap Abort process.</td>
<td>Check the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2335</td>
<td>Cannot continue execution and Reset Error in TEACH mode.</td>
<td>Check the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2336</td>
<td>Cannot use Here statement with a parallel process.</td>
<td>Go Here .Z(0) ! D10; MemOn(1) ! is not executable. Change the program to: P999 = Here Go P999 Here .Z(0) ! D10; MemOn(1) !</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>---------</td>
<td>------------</td>
</tr>
<tr>
<td>2337</td>
<td>Cannot execute except from an event handler functions of GUI Builder.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2338</td>
<td>Cannot execute Xqt, data input, and output for TP in a TEST mode.</td>
<td>Cannot execute in TEST mode.</td>
<td>Review the program.</td>
<td></td>
</tr>
<tr>
<td>2339</td>
<td>Cannot execute in stand-alone mode.</td>
<td>Change the setting to “cooperative mode” and execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2340</td>
<td>Specified value in InBCD function is an invalid BCD value.</td>
<td>Review the program.</td>
<td>Tens digit</td>
<td>Units digit</td>
</tr>
<tr>
<td>2341</td>
<td>Specified value in the OpBCD statement is an invalid BCD value.</td>
<td>Review the program.</td>
<td>The specified value</td>
<td></td>
</tr>
<tr>
<td>2342</td>
<td>Cannot change the status for output bit configured as remote output.</td>
<td>Check the remote I/O setting.</td>
<td>I/O number</td>
<td>1: bit, 2: byte, 3: word</td>
</tr>
<tr>
<td>2343</td>
<td>Output time for asynchronous output commanded by On or Off statement is out of the available range.</td>
<td>Review the program.</td>
<td>The specified time</td>
<td></td>
</tr>
<tr>
<td>2344</td>
<td>I/O input/output bit number is out of available range or the board is not installed.</td>
<td>Review the program. Check whether the expansion I/O board and Fieldbus I/O board are correctly detected.</td>
<td>Bit number</td>
<td></td>
</tr>
<tr>
<td>2345</td>
<td>I/O input/output byte number is out of available range or the board is not installed.</td>
<td>Review the program. Check whether the expansion I/O board and Fieldbus I/O board are correctly detected.</td>
<td>Byte number</td>
<td></td>
</tr>
<tr>
<td>2346</td>
<td>I/O input/output word number is out of available range or the board is not installed.</td>
<td>Review the program. Check whether the expansion I/O board and Fieldbus I/O board are correctly detected.</td>
<td>Word number</td>
<td></td>
</tr>
<tr>
<td>2347</td>
<td>Memory I/O bit number is out of available range.</td>
<td>Review the program.</td>
<td>Bit number</td>
<td></td>
</tr>
<tr>
<td>2348</td>
<td>Memory I/O byte number is out of available range.</td>
<td>Review the program.</td>
<td>Byte number</td>
<td></td>
</tr>
<tr>
<td>2349</td>
<td>Memory I/O word number is out of available range.</td>
<td>Review the program.</td>
<td>Word number</td>
<td></td>
</tr>
<tr>
<td>2350</td>
<td>Command allowed only when virtual I/O mode is active.</td>
<td>The command can be executed only for virtual I/O mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2353</td>
<td>Specified command cannot be executed from the Command window.</td>
<td>Execute specified command from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2354</td>
<td>Cannot execute the I/O output command when the Enable Switch is OFF.</td>
<td>Execute the I/O output command with the enable switch gripped.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2360</td>
<td>File failure. Failed to open the configuration file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2361</td>
<td>File failure. Failed to close the configuration file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2362</td>
<td>File failure. Failed to open the key of the configuration file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2363</td>
<td>File failure. Failed to obtain a string from the configuration file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>-------------------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>2364</td>
<td>File failure. Failed to write in the configuration file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2365</td>
<td>File failure. Failed to update the configuration file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2370</td>
<td>The string combination exceeds the maximum string length.</td>
<td>The maximum string length is 255.</td>
<td>Review the program.</td>
<td>Combined string length</td>
</tr>
<tr>
<td>2371</td>
<td>String length is out of range.</td>
<td>The maximum string length is 255.</td>
<td>Review the program.</td>
<td>The specified length</td>
</tr>
<tr>
<td>2372</td>
<td>Invalid character is specified after the ampersand in the Val function.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2373</td>
<td>Illegal string specified for the Val function.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2374</td>
<td>String Failure. Invalid character code in the string.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2380</td>
<td>Cannot use ‘0’ for Step value in For...Next.</td>
<td>Check the Step value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2381</td>
<td>Relation between For...Next and GoSub is invalid. Going in or out of a For...Next using a Goto statement.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2382</td>
<td>Cannot execute Return while executing OnErr.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2383</td>
<td>Return was used without GoSub.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2384</td>
<td>Case or Send was used without Select.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2385</td>
<td>Cannot execute EResume while executing GoSub.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2386</td>
<td>EResume was used without OnErr.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2400</td>
<td>Curve failure. Failed to open the Curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2401</td>
<td>Curve failure. Failed to allocate the header data of the curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2402</td>
<td>Curve failure. Failed to write the curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2403</td>
<td>Curve failure. Failed to open the curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2404</td>
<td>Curve failure. Failed to update the curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2405</td>
<td>Curve failure. Failed to read the curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2406</td>
<td>Curve failure. Curve file is corrupt.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2407</td>
<td>Curve failure. Specified a file other than a curve file.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2408</td>
<td>Curve failure. Version of the curve file is invalid.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>2409</td>
<td>Curve failure. Robot number in the curve file is invalid.</td>
<td>Reboot the controller.</td>
<td>Create a Curve file again.</td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>2410</td>
<td>Curve failure. Cannot allocate enough memory for the CVMove statement.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2411</td>
<td>Specified point data in the Curve statement is beyond the maximum count.</td>
<td>The maximum number of points specified in the Curve statement is 200. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2412</td>
<td>Specified number of output commands in the Curve statement is beyond the maximum count.</td>
<td>The maximum number of output commands specified in the Curve statement is 16. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2413</td>
<td>Curve failure. Specified internal code is beyond the allowable size in Curve statement.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2414</td>
<td>Specified continue point data $P(\cdot)$ is beyond the maximum count.</td>
<td>The maximum number of points specified continuously is 200. Review the program.</td>
<td>Start point</td>
<td>End point</td>
</tr>
<tr>
<td>2415</td>
<td>Curve failure. Cannot create the curve file.</td>
<td>Reboot the controller. Create a Curve file again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2416</td>
<td>Curve file does not exist.</td>
<td>Check whether the specified Curve file name is correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2417</td>
<td>Curve failure. Output command is specified before the point data.</td>
<td>Check whether no output command is specified before the point data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2430</td>
<td>Error message failure. Error message file does not exist.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2431</td>
<td>Error message failure. Failed to open the error message file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2432</td>
<td>Error message failure. Failed to obtain the header data of the error message file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2433</td>
<td>Error message failure. Error message file is corrupted.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2434</td>
<td>Error message failure. Specified a file other than the error message file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2435</td>
<td>Error message failure. Version of the error message file is invalid.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2440</td>
<td>File Error. File number is already used.</td>
<td>Check the file number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2441</td>
<td>File Error. Failed to open the file.</td>
<td>Make sure the file exists and you specified the file correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2442</td>
<td>File Error. The file is not open.</td>
<td>Open the file in advance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2443</td>
<td>File Error. The file number is being used by another task.</td>
<td>Check the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2444</td>
<td>File Error. Failed to close the file.</td>
<td>Check the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2445</td>
<td>File Error. File seek failed.</td>
<td>Review the program. Check the pointer setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2446</td>
<td>File Error. All file numbers are being used.</td>
<td>Close unnecessary files.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2447</td>
<td>File Error. No read permission.</td>
<td>Use ROpen or UOpen that has read access to the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>2448</td>
<td>File Error. No write permission.</td>
<td>Use WOpen or UOpen that has write access to the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2449</td>
<td>File Error. No binary permission.</td>
<td>Use BOpen that has binary access to the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2450</td>
<td>File Error. Failed to access the file.</td>
<td>Check the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2451</td>
<td>File Error. Failed to write the file.</td>
<td>Check the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2452</td>
<td>File Error. Failed to read the file.</td>
<td>Check the file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2453</td>
<td>File Error. Cannot execute the command for current disk.</td>
<td>The specified command is not available in the current disk (ChDisk).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2454</td>
<td>File Error. Invalid disk.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2455</td>
<td>File Error. Invalid drive.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2456</td>
<td>File Error. Invalid folder.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2460</td>
<td>Database Error. The database number is already being used.</td>
<td>Review the program.</td>
<td>Specify the number of other database. Close the database.</td>
<td></td>
</tr>
<tr>
<td>2461</td>
<td>Database Error. The database is not open.</td>
<td>Review the program.</td>
<td>Open the database.</td>
<td></td>
</tr>
<tr>
<td>2462</td>
<td>Database Error. The database number is being used by another task.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2470</td>
<td>Windows Communication Error. Invalid status.</td>
<td>Reboot the Controller.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>2471</td>
<td>Windows Communication Error. Invalid answer.</td>
<td>Reboot the Controller.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>2474</td>
<td>Windows Communication Error. No request.</td>
<td>Reboot the Controller.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>2475</td>
<td>Windows Communication Error. Data buffer overflow.</td>
<td>Reduce the data volume.</td>
<td>Review the program.</td>
<td></td>
</tr>
<tr>
<td>2476</td>
<td>Windows Communication Error. Failed to wait for event.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2477</td>
<td>Windows Communication Error. Invalid folder.</td>
<td>Make sure the specified folder is correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2478</td>
<td>Windows Communication Error. Invalid error code.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2500</td>
<td>Specified event condition for Wait is beyond the maximum count.</td>
<td>The maximum number of event conditions is 8. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2501</td>
<td>Specified bit number in the Ctr function was not initialized with a CTReset statement.</td>
<td>Review the program.</td>
<td>The specified bit number</td>
<td></td>
</tr>
<tr>
<td>2502</td>
<td>Task number is beyond the maximum count to execute.</td>
<td>The available number of tasks that can be executed simultaneously is 32 for normal tasks, and 16 for background tasks. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>2503</td>
<td>Cannot execute Xqt when the specified task number is already executing.</td>
<td>Review the program.</td>
<td></td>
<td>The specified</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>task number</td>
</tr>
<tr>
<td>2504</td>
<td>Task failure. Specified manipulator is already executing a parallel</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>process.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2505</td>
<td>Not enough data for Input statement variable assignment.</td>
<td>Check the content of</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>communication data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2506</td>
<td>Specified variable for the Input statement is beyond the maximum</td>
<td>For OP, only one variable</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>can be specified. For other</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>devices, up to 32 variables</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2507</td>
<td>All counters are in use and cannot initialize a new counter with</td>
<td>The available number of the</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>counters that can be set</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>simultaneously is 16.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2508</td>
<td>OnErr failure. Failed to process the OnErr statement.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2509</td>
<td>OnErr failure. Failed to process the OnErr statement.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2510</td>
<td>Specified I/O label is not defined.</td>
<td>The specified I/O label is</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>not registered. Check the</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>I/O label file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2511</td>
<td>SyncUnlock statement is used without executing a previous SyncLock</td>
<td>Review the program.</td>
<td></td>
<td>Signal number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>statement. Review the</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2512</td>
<td>SyncLock statement was already executed.</td>
<td>The SyncLock statement</td>
<td>Signal number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>cannot be executed for the</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>second time in a row.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2513</td>
<td>Specified point label is not defined.</td>
<td>The specified point label</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is not registered. Check the</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>point file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2514</td>
<td>Failed to obtain the motor on time of the robot.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2515</td>
<td>Failed to configure the date or the time.</td>
<td>Check whether a date and</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>time is set correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2516</td>
<td>Failed to obtain the debug data or to initialize.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2517</td>
<td>Failed to convert into date or time.</td>
<td>Check the time set on</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2518</td>
<td>Larger number was specified for the start point data than the end</td>
<td>Specify a larger number for</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>point data than that for</td>
<td>Start point</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the start point data.</td>
<td>End point</td>
<td></td>
</tr>
<tr>
<td>2519</td>
<td>Invalid format syntax for FmtStr$.</td>
<td>Check the format.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2520</td>
<td>File name is too long.</td>
<td>Check whether the specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>point file name is correct.</td>
<td>The maximum</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>string length</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>of the file name</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>is 32.</td>
<td></td>
</tr>
<tr>
<td>2521</td>
<td>File path is too long.</td>
<td>Check whether the specified</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>point file name is correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2522</td>
<td>File name is invalid.</td>
<td>Make sure you don’t use</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>improper characters for</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2523</td>
<td>The continue process was already executed.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2524</td>
<td>Cannot execute Xqt when the specified trap number is already executing.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2525</td>
<td>Password is invalid.</td>
<td>Check whether a password is</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>set correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2526</td>
<td>No wait terms.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2527</td>
<td>Too many variables used for global variable wait.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2528</td>
<td>The global variable that was not able to be used for the wait command was specified.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2529</td>
<td>Cannot use ByRef if the variable is used for global variable wait.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2530</td>
<td>Too many point files.</td>
<td>Check the point file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2531</td>
<td>The point file is used by another robot.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2532</td>
<td>Cannot progress to the point position because there is undefined data.</td>
<td>Check the point data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2533</td>
<td>Error on INP or OUTP.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2534</td>
<td>No main function to start for Restart statement.</td>
<td>Without executing main function, Restart is called.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2535</td>
<td>Does not allow Enable setting in Teach mode to be changed.</td>
<td>Setup the authority.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2536</td>
<td>Failed to change Enable setting in Teach mode.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2537</td>
<td>Count of point data P(;) is not correct or format of parameter is not correct.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2539</td>
<td>Password is invalid.</td>
<td>Check the password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2540</td>
<td>Not connected to RC+.</td>
<td>Connect to the RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2541</td>
<td>Duplicate parameter.</td>
<td>Same robot number was specified. Check the parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2542</td>
<td>The specified work queue number is invalid.</td>
<td>Available work queue number s are from 1 to 16. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2543</td>
<td>Invalid sequence was specified.</td>
<td>Specified sequence name cannot be found. Review the sequence name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2544</td>
<td>Invalid object was specified.</td>
<td>Specified object name cannot be found. Review the object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2545</td>
<td>Invalid calibration was specified.</td>
<td>Specified calibration name cannot be found. Review the calibration name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2546</td>
<td>Cannot turn on the motor immediately after opening the safeguard.</td>
<td>Wait 1.5 seconds after the safeguard is open, and then execute the motor on.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2548</td>
<td>Too many force files. Delete the force files or use the existing force files.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2549</td>
<td>The force file which is not associated with the robot cannot be specified. Specify the correct force file.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2550</td>
<td>Specified command is not supported for joint type robot and cartesian type robot.</td>
<td>Specified robot is not supported. Check the robot configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2551</td>
<td>Failed to Get the health information.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2552</td>
<td>Does not allow setting in UL mode to be changed.</td>
<td>Setup the authority.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2553</td>
<td>Failed to change setting in UL mode.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>2556</td>
<td>An excessive loop was detected. Please reduce the number of looped tasks or set Wait</td>
<td>This error message is only displayed in T/VT series Manipulators. Do not perform any processing such as infinite loop or any other similar processing as much as possible. For more details, refer to Restrictions of Functions in Maintenance, T/VT series manual.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2557</td>
<td>An error occurred in Trap. Note 1: Detailed error information Following the detailed error information, take a relevant countermeasure.</td>
<td>An error occurred in Trap. Check the corresponding error code in the system history and take countermeasures.</td>
<td>Detailed error information</td>
<td></td>
</tr>
<tr>
<td>2558</td>
<td>Argument parameter is too long.</td>
<td>Confirm a parameter of the argument.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2559</td>
<td>Cannot execute when the motor is in the off state.</td>
<td>Change to the state to motor on and execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2560</td>
<td>The current robot number and the robot number of the force guide sequence property do not match. Please check the robot number.</td>
<td>Confirm the current robot number and the robot number of the force guide sequence.</td>
<td>Robot number</td>
<td></td>
</tr>
<tr>
<td>2561</td>
<td>The current robot type and the robot type of the force guide sequence property do not match. Reconfigure the RobotNumber property.</td>
<td>Confirm the current robot number and the robot number of the force guide sequence property. Reconfigure the RobotNumber property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2562</td>
<td>The current tool number and the robot tool of the force guide sequence property do not match. Please check the tool number.</td>
<td>Confirm the current tool number and the robot tool of the force guide sequence property.</td>
<td>Tool number</td>
<td></td>
</tr>
<tr>
<td>2563</td>
<td>The point file being loaded does not match the point file of the force guide sequence property. Please check the point file.</td>
<td>Confirm the loaded point file and the point file of the force guide sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2564</td>
<td>An instruction that cannot be executed during torque control was executed.</td>
<td>Turn OFF the torque control and execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2565</td>
<td>Prohibited command while tracking was executed.</td>
<td>Delete Prohibited commands from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2566</td>
<td>Cannot execute the FGRun command for same robot.</td>
<td>Cannot execute the FGRun command for same robot. End the FGRun command or execute it in other robot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2567</td>
<td>Cannot execute the FGGet command for the running force guide sequence.</td>
<td>Cannot execute the FGGet command for the running force guide sequence. Execute it after the force guide sequence ends.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2568</td>
<td>An instruction that cannot be executed by parallel processing was executed. Review the program.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2569</td>
<td>Cannot get the force guide sequence property.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2570</td>
<td>Sequence number is out of range. Please check the specified sequence number.</td>
<td>Sequence number is from 1 to 64. Confirm the specified sequence number.</td>
<td>Sequence number</td>
<td></td>
</tr>
<tr>
<td>2571</td>
<td>Object number is out of range. Please check the specified object number.</td>
<td>Object number is from 1 to 16. Confirm the specified object number.</td>
<td>Object number</td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2572</td>
<td>Cannot clear the result of the force guide.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2573</td>
<td>Cannot set the result of the force guide.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2574</td>
<td>Cannot get the result of the force guide.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2575</td>
<td>Storing the force guide sequence result in a variable failed.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2576</td>
<td>Force Sequence name that does not exist was specified.</td>
<td>Confirm the specified force sequence name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2577</td>
<td>Force Object name that does not exist was specified.</td>
<td>Confirm the specified force object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2578</td>
<td>Cannot execute the FGGet command for the unexecuted force guide sequence.</td>
<td>Confirm the specified force guide sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2600</td>
<td>Mass Property Object number is out of the allowable range.</td>
<td>The MassProperties numbers that can be specified are from 1 to 15.</td>
<td>Please review the program.</td>
<td></td>
</tr>
<tr>
<td>2601</td>
<td>Mass Property Object is not defined.</td>
<td>Please confirm the setting of MassProperties.</td>
<td>Please review the program.</td>
<td></td>
</tr>
<tr>
<td>2602</td>
<td>Cannot clear Mass Property Object while in use. Specify another Mass Property Object before clearing the previous object.</td>
<td>MP cannot be cleared while in use. Please confirm whether MP is in use.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2603</td>
<td>Cannot clear Mass Property Object number '0'</td>
<td>MP-number 0 cannot be cleared. Please review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2840</td>
<td>Failed in the confirmation of the DU connection count.</td>
<td>Check whether the Drive Unit is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2841</td>
<td>Failed in the acquisition of the DU connection count.</td>
<td>Check whether the Drive Unit is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2842</td>
<td>Failed in the confirmation of the DU connection information.</td>
<td>Check whether the Drive Unit is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2843</td>
<td>Failed in the acquisition of the DU connection information.</td>
<td>Check whether the Drive Unit is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2844</td>
<td>There is a missing number or repetition in the dip switch setting of DU.</td>
<td>Check the dip switches of the Drive Unit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2845</td>
<td>The drive unit (DU) used by the robot is not connected.</td>
<td>Check whether the Drive Unit is connected properly.</td>
<td></td>
<td>Delete the robot registration or connect the DU with the manipulator registered.</td>
</tr>
<tr>
<td>2846</td>
<td>Because the increase and decrease of the drive unit was recognized, the controller unit is rebooted.</td>
<td>The controller was rebooted due to change of connection with the Drive Unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2847</td>
<td>The dip switch setting of the Force Sensor I/F unit is improper.</td>
<td>It is necessary to change the dip switch setting.</td>
<td>Please inquire with us.</td>
<td></td>
</tr>
<tr>
<td>2848</td>
<td>The Force Sensor I/F unit to which the Force Sensor is registered is not connected. Check connection.</td>
<td>Please confirm whether it is possible to connect it with Force Sensor I/F unit correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>2849</td>
<td>Failed to initialize the Force Sensor I/F unit. Check connection.</td>
<td>Please confirm whether it is possible to connect it with Force Sensor I/F unit correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2850</td>
<td>Failed to initialize the Force Sensor I/F unit. Check connection.</td>
<td>Please confirm whether it is possible to connect it with Force Sensor I/F unit correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2851</td>
<td>The Force Sensor which is different from the registered sensor is connected. Check connection or review the setting.</td>
<td>The serial number of the sensor connected with the registered sensor is not corresponding. Please exchange it for a new sensor after confirming the connection, returning to the connected sensor, or invalidating the sensor. In case of intended replacement, configure the connection settings again in the sensor setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2852</td>
<td>The registered Force Sensor is not connected. Check connection.</td>
<td>Please confirm whether it is possible to connect it with the registered sensor correctly. Please invalidate the sensor when you do not connect the sensor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2853</td>
<td>Failed to update the Force Sensor I/F unit software. Review the update procedure.</td>
<td>Please review the soft update setting. Please confirm the connection with Force Sensor I/F unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2854</td>
<td>Failed to update the Force Sensor I/F unit software. Review the update procedure.</td>
<td>Please confirm the file name. Please confirm the update file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2855</td>
<td>Failed to update the Force Sensor I/F unit software. Review the update procedure.</td>
<td>Please confirm the power supply and the connection of Force Sensor I/F unit. Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2856</td>
<td>The Force Sensor I/F unit with an old version is connected. Update the Force Sensor I/F unit software.</td>
<td>The version of the connected Force Sensor I/F unit needs to be updated. Update the Force Sensor I/F unit. For update procedures, please inquiry with us.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2857</td>
<td>The robot registered to the Force Sensor I/F unit is not connected. Review the robot registration or the Force Sensor configuration.</td>
<td>The robot that relates to the sensor is not registered. Please review the registration of the robot or invalidate the robot connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2858</td>
<td>Failed to allocate memory for the force monitor.</td>
<td>Reboot the controller. Please inquire with us if a similar error occurs after rebooting it.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2859</td>
<td>Failed to allocate memory for the force log.</td>
<td>Reboot the controller. Please inquire with us if a similar error occurs after rebooting it.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2860</td>
<td>The force monitor object specified in the force log is in use. Specify another force monitor object.</td>
<td>The same FM number cannot be specified. Please specify a different FM number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2861</td>
<td>The maximum number of the force logs is executed. Review the log timing.</td>
<td>The greatest log number is used. Please confirm the number of logs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2862</td>
<td>Failed to allocate memory of force function.</td>
<td>Reboot the controller. Please inquire with us if a similar error occurs after rebooting it.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------</td>
</tr>
<tr>
<td>2863</td>
<td>Execution of force guide sequence, RecordStart, FCMStart and LogStart cannot be executed at the same time. Please review the program.</td>
<td>Execute after the LogStart property ends by LogEnd property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2864</td>
<td>Execution of force guide sequence, RecordStart, FCMStart and force monitor cannot be executed at the same time. Please quit either.</td>
<td>Execute after quitting the Force Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2865</td>
<td>Execution of force guide sequence, RecordStart, FCMStart and LogStart cannot be executed at the same time. Please review the program.</td>
<td>Execute the LogStart property after the RecordStart property ends by force guide sequence, force control monitor, or the RecordEnd property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2866</td>
<td>Execution of force guide sequence, RecordStart, FCMStart and force monitor cannot be executed at the same time. Please quit either.</td>
<td>Execute the force monitor after quitting the RecordStart property by force guide sequence, force control monitor, or the RecordEnd property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2867</td>
<td>The specified channel in use. Specify another channel.</td>
<td>The same channel cannot be specified. Specify a different channel to execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2868</td>
<td>The force monitor object being used is specified. Please specify another force monitor object.</td>
<td>The same FM number cannot be specified. Specify a different FM number to execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2869</td>
<td>The specified duration of measurement is smaller than the specified measurement interval. Check the parameter.</td>
<td>Specify the measurement time larger than the measurement interval to execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2870</td>
<td>The product of the specified duration of measurement and the specified measurement interval is out of allowable range. Check the parameter.</td>
<td>Check the measurement time and interval.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2871</td>
<td>Execution of force guide sequence, RecordStart, FCMStart, force monitor cannot be used more than three at the same time.</td>
<td>To execute newly, make sure to quit either of the two running items and execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2872</td>
<td>Force monitor cannot be launched twice.</td>
<td>To start force monitor newly, quit the running force monitor and start a new one.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2880</td>
<td>Failed to initialize the Force Sensor I/F board. Check connection.</td>
<td>Check connection of the controller and Force Sensor I/F board. Reboot the controller. Please inquire with us if a similar error occurs even after rebooting the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2881</td>
<td>Failed to initialize the Force Sensor I/F board. Check connection.</td>
<td>Check connection of the controller and Force Sensor I/F board. Reboot the controller. Please inquire with us if a similar error occurs even after rebooting the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>2882</td>
<td>Detected two boards: Force Sensor I/F board and RS-232C board.</td>
<td>Remove the Force Sensor I/F board or the second board of RS-232C board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If using the Force Sensor I/F board, RS-232C board is available up to</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>one board.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2883</td>
<td>Detected two boards: Force Sensor I/F board and RS-232C board with</td>
<td>Return the setting to the first board of RS-232C board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>the second board setting.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If using the Force Sensor I/F board, return the setting to the first</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>board of RS-232C board.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2884</td>
<td>Failed to initialize the Force Sensor I/F board.</td>
<td>Check connection of the controller and Force Sensor I/F board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check connection.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Please inquire with us if a similar error occurs even after rebooting the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2885</td>
<td>Sensor 3 and 4 of Force Sensors are enabling.</td>
<td>Disable the sensor 3 and 4 of the Force Sensor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>If using Force Sensor I/F board, disable the sensor 3 and 4 of Force</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sensors.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check connection of the Force Sensor.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Please inquire with us if a similar error occurs even after rebooting the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2887</td>
<td>Detected Force Sensor I/F board and Force Sensor I/F unit.</td>
<td>Unable to use the Force Sensor I/F board and Force Sensor I/F unit at the same time.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Remove either Force Sensor I/F board or Force Sensor I/F unit.</td>
<td>Remove the Force Sensor I/F board or Force Sensor I/F unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2888</td>
<td>Unsupported Force Sensor is set.</td>
<td>Check the configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check the configuration.</td>
<td>Firmware version may be old.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check whether the firmware version is supported and update it as necessary.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2900</td>
<td>Failed to open as server for the Ethernet port.</td>
<td>Check whether the Ethernet port is set properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check whether the Ethernet cable is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2901</td>
<td>Failed to open as client for the Ethernet port.</td>
<td>Check whether the Ethernet port is set properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check whether the Ethernet cable is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2902</td>
<td>Failed to read from the Ethernet port.</td>
<td>Check whether the port of communication recipient is not close.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2904</td>
<td>Invalid IP Address was specified.</td>
<td>Review the IP address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2905</td>
<td>Ethernet failure.</td>
<td>Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>No specification of Server/Client.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2906</td>
<td>Ethernet port was not configured.</td>
<td>Check whether the Ethernet port is set properly.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2907</td>
<td>Ethernet port was already in use by another task.</td>
<td>A single port cannot be used by more than one task.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2908</td>
<td>Cannot change the port parameters while the Ethernet port is open.</td>
<td>The port parameters cannot be changed while the port is open.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2909</td>
<td>Ethernet port is not open.</td>
<td>To use the Ethernet port, execute the OpenNet statement.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>2910</td>
<td>Timeout reading from an Ethernet port.</td>
<td>Check the communication.</td>
<td>Timeout value</td>
<td></td>
</tr>
<tr>
<td>2911</td>
<td>Failed to read from an Ethernet port.</td>
<td>Check the communication.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2912</td>
<td>Ethernet port was already open by another task.</td>
<td>A single port cannot be used by more than one task.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2913</td>
<td>Failed to write to the Ethernet port.</td>
<td>Check whether the Ethernet port is set properly.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2914</td>
<td>Ethernet port connection was not completed.</td>
<td>Check whether the port of communication recipient is open.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2915</td>
<td>Data received from the Ethernet port is beyond the limit of one line.</td>
<td>The maximum length of a line is 255 bytes.</td>
<td></td>
<td>The number of bytes in a received line</td>
</tr>
<tr>
<td>2916</td>
<td>Failed to process a dummy file of virtual Ethernet port</td>
<td>Check the content of the dummy file.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2920</td>
<td>RS-232C failure. RS-232C port process error.</td>
<td>Check whether the RS-232C board is correctly detected.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2921</td>
<td>Failed to read from the RS-232C port.</td>
<td>Check the parameter and communication.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2922</td>
<td>Failed to read from the RS-232C port. Overrun error.</td>
<td>Slow down data transfer or reduce data size.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2926</td>
<td>The RS-232C port hardware is not installed.</td>
<td>Check whether the RS-232C board is correctly detected.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2927</td>
<td>RS-232C port is already open by another task.</td>
<td>A single port cannot be used by more than one task.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2928</td>
<td>Cannot change the port parameters while the RS-232C port is open.</td>
<td>The port parameters cannot be changed while the port is open.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2929</td>
<td>RS-232C port is not open.</td>
<td>To use the RS-232C port, execute the OpenCom statement.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2930</td>
<td>Timeout reading from the RS-232C port.</td>
<td>Check the communication.</td>
<td>Timeout value</td>
<td></td>
</tr>
<tr>
<td>2931</td>
<td>Failed to read from the RS-232C port.</td>
<td>Check the communication.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2932</td>
<td>RS-232C port is already open by another task.</td>
<td>A single port cannot be used by more than one task.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2933</td>
<td>Failed to write to the RS-232C port.</td>
<td>Check the communication.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2934</td>
<td>RS-232C port connection not completed.</td>
<td>Check the RS-232C port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2935</td>
<td>Data received from the RS-232C port is beyond the limit of one line.</td>
<td>The maximum length of a line is 255 bytes.</td>
<td></td>
<td>The number of bytes in a received line</td>
</tr>
<tr>
<td>2936</td>
<td>Failed to process a dummy file of virtual RS-232C port</td>
<td>Check the content of the dummy file.</td>
<td></td>
<td>Port number</td>
</tr>
<tr>
<td>2937</td>
<td>Cannot execute while Remote RS-232C are using.</td>
<td>Specified port is currently used. Specify another port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2938</td>
<td>Cannot execute while ModBus are using.</td>
<td>Specified port is currently used. Specify another port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2950</td>
<td>Daemon failure. Failed to create the daemon thread.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2951</td>
<td>Daemon failure. Timeout while creating the daemon thread.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>2952</td>
<td>TEACH/AUTO switching key input signal failure was detected.</td>
<td>Set the TP key switch to TEACH or AUTO properly. Check whether the TP is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2953</td>
<td>ENABLE key input signal failure was detected.</td>
<td>Check whether the TP is connected properly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2954</td>
<td>Relay weld was detected.</td>
<td>Overcurrent probably occurred due to short-circuit failure. Investigate the cause of the problem and take necessary measures and then replace the DPB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2955</td>
<td>Temperature of regeneration resistor was higher than the specified temperature.</td>
<td>Check whether the filter is not clogged up and the fan does not stop. If there is no problem on the filter and fan, replace the regenerative module.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2970</td>
<td>MNG failure. Area allocate error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2971</td>
<td>MNG failure. Real time check error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2972</td>
<td>MNG failure. Standard priority error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2973</td>
<td>MNG failure. Boost priority error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2974</td>
<td>MNG failure. Down priority error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2975</td>
<td>MNG failure. Event wait error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2976</td>
<td>MNG failure. Map close error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2977</td>
<td>MNG failure. Area free error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2978</td>
<td>MNG failure. AddIOMem error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2979</td>
<td>MNG failure. AddInPort error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2980</td>
<td>MNG failure. AddOutPort error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2981</td>
<td>MNG failure. AddInMemPort error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2982</td>
<td>MNG failure. AddOutMemPort error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2983</td>
<td>MNG failure. IntervalOutBit error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2984</td>
<td>MNG failure. CtrReset error.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2997</td>
<td>Collision Detection</td>
<td>If you use the simulator, check if the object is placed in the direction of the robot motion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2998</td>
<td>AbortMotion attempted when robot was not moving</td>
<td>See Help for AbortMotion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2999</td>
<td>AbortMotion attempted when robot was moving</td>
<td>See Help for AbortMotion.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3000</td>
<td>OBJ file size is large. TP1 may not be able to build this project.</td>
<td>When it is necessary to build the project from TP1, consider to reduce the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3001</td>
<td>The number of variables which is using Wait command is near the maximum allowed.</td>
<td>The number of variables which is using Wait command is exceeding 56 (the maximum is 64). Check if there are unnecessary variables.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3002</td>
<td>DLL file cannot be found.</td>
<td>Check the DLL file exists in either of the following folders:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Project folder</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Windows system folder</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Configuration folder of environment variable PATH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3003</td>
<td>DLL function cannot be found.</td>
<td>Check the name of the specified function. Also check the DLL file if the specified function exists in the DLL.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3050</td>
<td>Main function is not defined.</td>
<td>Declare a Main function.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3051</td>
<td>Function does not exist.</td>
<td>Declare an unresolved function.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3052</td>
<td>Variable does not exist.</td>
<td>Declare an unresolved variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3100</td>
<td>Syntax error.</td>
<td>Correct the syntax error.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3101</td>
<td>Parameter count error.</td>
<td>The number of parameters is excess or deficiency. Correct the parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3102</td>
<td>File name length is beyond the maximum allowed.</td>
<td>Shorten the file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3103</td>
<td>Duplicate function definition.</td>
<td>Change the function name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3104</td>
<td>Duplicate variable definition <code>**</code>.</td>
<td>Change the variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3105</td>
<td>Global and Global Preserve variables cannot be defined inside a function block.</td>
<td>Declare the Global and Global Preserve variables outside the function block.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3106</td>
<td>An undefined function was specified.</td>
<td>Specify a valid function name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3107</td>
<td>Both While and Until for Do...Loop was specified.</td>
<td>The While/Until statement is specified for both Do statement and Loop statement. Delete either While/Until statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3108</td>
<td>Specified line number or label <code>**</code> does not exist.</td>
<td>Set the line label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3109</td>
<td>Overflow error.</td>
<td>The direct numerical specification overflows. Reduce the numeric value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3110</td>
<td>An undefined variable was specified <code>**</code>.</td>
<td>There is an undefined variable. Declare the variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3111</td>
<td>Specified variable is not an array variable.</td>
<td>Specify the array variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3112</td>
<td>Cannot change the dimensions of the array variable.</td>
<td>Dimension of the array cannot be changed in Redim statement during the run time. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3114</td>
<td>Specified Next variable does not match the specified For variable.</td>
<td>Correct the variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3115</td>
<td>Cannot use a point expression in the first argument.</td>
<td>Specify a single point for the point flag setting. Do not specify a point expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3116</td>
<td>Array number of dimensions does not match the declaration.</td>
<td>Check the number of array dimensions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>3117</td>
<td>File cannot be found.</td>
<td>The file that configures the project cannot be found. Check the project folder if the file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3118</td>
<td>Corresponding EndIf cannot be found.</td>
<td>The number of EndIf statements that correspond to If and Elself statements is not enough. Add the Endif statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3119</td>
<td>Corresponding Loop cannot be found.</td>
<td>The number of Loop statements that correspond to Do statements is not enough. Add the Loop statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3120</td>
<td>Corresponding Next cannot be found.</td>
<td>The number of Next statements that correspond to For statements is not enough. Add the Next statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3121</td>
<td>Corresponding Send cannot be found.</td>
<td>The number of Send statements that correspond to Select statements is not enough. Add the Send statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3123</td>
<td>On/Off statements are beyond the maximum count.</td>
<td>An upper limit (max. 16) is set on the number of On/Off statements in Curve statement. Check the upper limit and correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3124</td>
<td>Point number is beyond the maximum count.</td>
<td>An upper limit (“200” for open curves, “50” for closed curves) is set on the available number of points in Curve statement. Check the upper limit and correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3125</td>
<td>Corresponding If cannot be found.</td>
<td>The number of EndIf statements that correspond to If statements is too many. Delete the unnecessary Endif.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3126</td>
<td>Corresponding Do cannot be found.</td>
<td>The number of Loop statements that correspond to Do statements is too many. Delete the unnecessary Loop.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3127</td>
<td>Corresponding Select cannot be found.</td>
<td>The number of Send statements that correspond to Select statements is too many. Delete the unnecessary Send.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3128</td>
<td>Corresponding For cannot be found.</td>
<td>The number of Next statements that correspond to For statements is too many. Delete the unnecessary Next.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3129</td>
<td>'_' cannot be used as the first character of an identifier.</td>
<td>Change the first character of the identifier to an alphabetic character.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3130</td>
<td>Cannot specify ROT parameter.</td>
<td>ROT parameter cannot be specified in BGo, Go, TGo, Jump, and Jump3 statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3131</td>
<td>Cannot specify ECP parameter.</td>
<td>ECP parameter cannot be specified in BGo, Go, TGo, Jump, Jump3, and Arc statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3132</td>
<td>Cannot specify Arch parameter.</td>
<td>Arch parameter cannot be specified in BGo, Go, TGo, Arc, Arc3, BMove, Move, and TMove statements. Correct the program</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3133</td>
<td>Cannot specify LimZ parameter.</td>
<td>LimZ parameter cannot be specified in BGo, Go, TGo, Jump3, Arc, Arc3, BMove, Move, and TMove statements. Correct the program</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3134</td>
<td>Cannot specify Sense parameter.</td>
<td>Sense parameter cannot be specified in BGo, Go, TGo, Arc, Arc3, BMove, Move, and TMove statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3135</td>
<td>Invalid parameter is specified.</td>
<td>Invalid parameter is specified in Xqt, and Call statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3137</td>
<td>Cannot specify the array variable subscript.</td>
<td>The array variable subscript cannot be specified. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3138</td>
<td>ByRef was not specified on Function declaration.</td>
<td>Specify ByRef in the parameter list of function declaration that is called by Call statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3139</td>
<td>Cannot execute the Xqt statement for a function that needs a ByRef parameter.</td>
<td>The Xqt statement cannot be executed for a function needing a ByRef parameter. Delete the ByRef parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3140</td>
<td>Cannot execute the Redim statement for a ByRef variable.</td>
<td>The Redim statement cannot be executed for a variable specifying ByRef parameter. Delete the ByRef parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3141</td>
<td>OBJ file is corrupt.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3142</td>
<td>OBJ file size is beyond the available size after compiling.</td>
<td>The compilation result exceeds the limit value (max. 1 MB per file). Divide the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3143</td>
<td>Indent length is beyond the available size.</td>
<td>The available length of the identifier is max. 32 characters for labels and variable names, and 64 characters for function names. Reduce the number of characters so as not to exceed the available length. For details of the available length, refer to EPSON RC+ User’s Guide “6.4 Function and Variable Names (Naming restriction)”.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3144</td>
<td>‘***’ already used for a function name.</td>
<td>Correct the identifier ‘***’ or the function name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3145</td>
<td>‘***’ already used for a Global Preserve variable.</td>
<td>Correct the identifier ‘***’ or the Global Preserve variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3146</td>
<td>‘***’ already used for a Global variable.</td>
<td>Correct the identifier ‘***’ or the Global variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3147</td>
<td>‘***’ already used for a Module variable.</td>
<td>Correct the identifier ‘***’ or the Module variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3148</td>
<td>‘***’ already used for a Local variable.</td>
<td>Correct the identifier ‘***’ or the Local variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3149</td>
<td>‘***’ already used for an I/O label.</td>
<td>Correct the identifier ‘***’ or the I/O label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3150</td>
<td>‘***’ already used for a User Error label.</td>
<td>Correct the identifier ‘***’ or the User Error label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3151</td>
<td>Cannot use a function parameter.</td>
<td>Argument cannot be specified for the function that is executed by the Trap statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3152</td>
<td>Specified elements of the array variable are beyond the available size.</td>
<td>Limit value of the array elements depends on the type of variables. Refer to EPSON RC+7.0 User’s Guide “6.7.6 Array” and correct the number of array elements so as not to exceed the limit value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>3153</td>
<td>Parameter type mismatch.</td>
<td>Parameter type does not match in Call, Force_GetForces, and Xqt statements. Correct the parameter type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3154</td>
<td>‘**’ is not an Input Bit label.</td>
<td>Specify a valid input bit label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3155</td>
<td>‘**’ is not an Input Byte label.</td>
<td>Specify a valid input byte label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3156</td>
<td>‘**’ is not an Input Word label.</td>
<td>Specify a valid input word label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3157</td>
<td>‘**’ is not an Output Bit label.</td>
<td>Specify a valid output bit label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3158</td>
<td>‘**’ is not an Output Byte label.</td>
<td>Specify a valid output byte label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3159</td>
<td>‘**’ is not an Output Word label.</td>
<td>Specify a valid output word label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3160</td>
<td>‘**’ is not a Memory Bit label.</td>
<td>Specify a valid memory I/O bit label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3161</td>
<td>‘**’ is not a Memory Byte label.</td>
<td>Specify a valid memory I/O byte label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3162</td>
<td>‘**’ is not a Memory Word label.</td>
<td>Specify a valid memory I/O word label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3163</td>
<td>Too many function arguments.</td>
<td>The maximum number of the function parameter is 100. Reduce the number of parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3164</td>
<td>Cannot compare with Boolean value.</td>
<td>The size of Boolean values cannot be compared. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3165</td>
<td>Cannot use Boolean value in the expression.</td>
<td>Boolean value cannot be used in the expression. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3166</td>
<td>Cannot compare between Boolean and expression.</td>
<td>The size of Boolean value and the expression cannot be compared. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3167</td>
<td>Cannot store Boolean value to a numeric variable.</td>
<td>Boolean value cannot be used in the numeric variable. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3168</td>
<td>Cannot store numeric value to a Boolean variable.</td>
<td>The numeric value cannot be used in Boolean variable. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3169</td>
<td>Undefined I/O label was specified.</td>
<td>Define a new I/O label or specify the defined I/O label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3170</td>
<td>Invalid condition expression was specified.</td>
<td>String expression is specified for the right side of the condition expression in Do or Loop statement. Correct the condition expression so that the right side of the expression is Boolean value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3171</td>
<td>Cannot compare between numeric value and string.</td>
<td>The numeric value and string cannot be compared. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3172</td>
<td>Cannot use a keyword for a variable name.</td>
<td>Some SPEL+ keywords cannot be used as the variable names. Correct the variable name not to overlap with the keywords.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3173</td>
<td>‘**’ is already used for a line label.</td>
<td>Correct the identifier ‘**’ or the line label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3174</td>
<td>Duplicate line number or label (**).</td>
<td>The line labels with the same name cannot be specified in the same function. Delete the line label ‘**’, or define a new line label and correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3175</td>
<td>Undefined Point label was specified.</td>
<td>Define a new point label or specify the defined point label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3176</td>
<td>An undefined variable was specified.</td>
<td>Define a new variable or specify the defined variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3177</td>
<td>'***' already used for a Point label.</td>
<td>Correct the identifier '***' or the point label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3178</td>
<td>Cannot use the result number.</td>
<td>The result number cannot be specified when a vision object that does not return multiple results is used in VSet and VGet statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3179</td>
<td>String literal is beyond the available length.</td>
<td>The limit value of the string length is max. 255 characters. Reduce the string length so as not to exceed the limit value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3180</td>
<td>Cannot change a calibration property value with the VSet command.</td>
<td>Calibration property cannot be changed in VSet statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3181</td>
<td>Array variable should be used with ByVal.</td>
<td>ByVal cannot be specified for the array variable. Specify the ByVal parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3182</td>
<td>Subscription was not specified.</td>
<td>Specify a subscription.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3183</td>
<td>Parameter cannot be omitted.</td>
<td>Add a parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3184</td>
<td>SYNC parameter cannot use with tracking command.</td>
<td>SYNC parameter cannot be specified in tracking commands. Delete the SYNC parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3185</td>
<td>Cannot use Queue data.</td>
<td>Queue data cannot be specified in BGo, BMove, TGo, and TMove statements. Delete the queue data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3186</td>
<td>Combination between Queue and Point data does not match.</td>
<td>Combination of queue data and point data cannot be specified for coordinate specification of Arc, Arc3, Jump3, and Jump3CP statements. Use either queue data or the point data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3187</td>
<td>Invalid Point flag value was specified.</td>
<td>Correct the program so that the point flag value is within the range from 0 to 127.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3188</td>
<td>Call command cannot be used in parallel processing.</td>
<td>Call command cannot be used parallel processing. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3189</td>
<td>Local variables cannot be used with the Wait command.</td>
<td>Change of local variable cannot be waited by Wait statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3190</td>
<td>Array variables cannot be used with the Wait command.</td>
<td>Change of array variable cannot be waited by Wait statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3191</td>
<td>Real variables cannot be used with the Wait command.</td>
<td>Change of real variable cannot be waited by Wait statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3192</td>
<td>String variables cannot be used with the Wait command.</td>
<td>Change of string variable cannot be waited by Wait statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3193</td>
<td>Vision object name is missing.</td>
<td>Vision object name cannot be omitted in VTeach statement. Specify the object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3194</td>
<td>Cannot use Boolean value for the timeout value.</td>
<td>Boolean value cannot be used for the timeout value of Wait statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3196</td>
<td>Fend statement was not found.</td>
<td>The number of Fend statements that correspond to Function statements is not enough. Add the Fend statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3197</td>
<td>Numeric variable name cannot use &quot;$&quot;.</td>
<td>Numeric variable name cannot use &quot;$&quot;. Correct the variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3198</td>
<td>String variables must have &quot;$&quot; suffix.</td>
<td>String variables must have a &quot;$&quot; suffix. Add a &quot;$&quot; suffix to the variable name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3199</td>
<td>Invalid object was specified.</td>
<td>Invalid vision object is specified in Vision Guide commands such as VSet and VGet. Specify the valid vision object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3200</td>
<td>Value is missing.</td>
<td>Add a value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3201</td>
<td>Expected ',' '.</td>
<td>Add ',' '.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3202</td>
<td>Expected '(' ')'.</td>
<td>Add '(' ')'.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3203</td>
<td>Expected ') ')'.</td>
<td>Add ') ')'.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3204</td>
<td>Identifier is missing.</td>
<td>Specify an identifier.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3205</td>
<td>Point is not specified.</td>
<td>Specify a point.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3206</td>
<td>Event condition expression is missing.</td>
<td>Add an event condition expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3207</td>
<td>Formula is missing.</td>
<td>Add a formula.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3208</td>
<td>String formula is missing.</td>
<td>Add a string formula.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3209</td>
<td>Point formula is missing.</td>
<td>Add a point formula.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3210</td>
<td>Line label was not specified.</td>
<td>Check if the specified line label exists in the program. Add a valid line label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3211</td>
<td>Variable was not specified.</td>
<td>Specify a variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3212</td>
<td>Corresponding Fend cannot be found.</td>
<td>The number of Fend statements that correspond to Function statements is not enough. Add the Fend statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3213</td>
<td>Expected ' : '.</td>
<td>Add ' : '</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3214</td>
<td>True/False was not specified.</td>
<td>True/False was not specified in the property of Vision Guide/GUI Builder or substitution of logical expression which requires Boolean value setting. Specify True or False.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3215</td>
<td>On/Off was not specified.</td>
<td>On or Off must be specified for the remote output logic setting of Motor, Brake, AutoLJM, SetSw, and Box statements. Specify On or Off.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3216</td>
<td>High/Low was not specified.</td>
<td>High or Low must be specified for the power mode setting of Power statement. Specify High or Low.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3217</td>
<td>Input bit label was not specified.</td>
<td>Input bit label is not specified in SetSW, CTReset statement, Sw, and Ctr function. Specify a valid input bit label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3218</td>
<td>Input byte label was not specified.</td>
<td>Input byte label is not specified in SetIn statement, In, and InBCD function. Specify a valid input byte label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3219</td>
<td>Input word label was not specified.</td>
<td>Input word label is not specified in SetInW statement, InReal, and InW function. Specify a valid input word label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3220</td>
<td>Output bit label was not specified.</td>
<td>Output bit label is not specified in On, Off statement, and Oport function. Specify a valid output bit label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3221</td>
<td>Output byte label was not specified.</td>
<td>Output byte label is not specified in Out, OpBCD statement, and Out function. Specify a valid output byte label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3222</td>
<td>Output word label was not specified.</td>
<td>Output word label is not specified in OutW, OutReal statement, OutW, and OutReal function. Specify a valid output word label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3223</td>
<td>Memory bit label was not specified.</td>
<td>Memory bit label is not specified in MemOn, MemOff statement, and MemSw function. Specify a valid memory bit label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3224</td>
<td>Memory byte label was not specified.</td>
<td>Memory byte label is not specified in MemOut statement and MemIn function. Specify a valid memory byte label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3225</td>
<td>Memory word label was not specified.</td>
<td>Memory word label is not specified in MemOutW statement and MemInW function. Specify a valid memory word label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3226</td>
<td>User error label was not specified.</td>
<td>User error label is not specified in Error statement. Specify a valid user error label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3227</td>
<td>Function name was not specified.</td>
<td>Function name is not specified in the statement that requires function name designation, such as Call and Xqt. Specify a valid function name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3228</td>
<td>Variable type was not specified.</td>
<td>Variable type is not specified for the parameter definition of Function statement and Preserve parameter specification of Global statement. Specify a correct variable type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3229</td>
<td>Invalid Trap statement parameter. Use Goto, Call, or Xqt.</td>
<td>Specify either GoTo, Call, or Xqt as a parameter of Trap statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3230</td>
<td>Expected For/Do/Function.</td>
<td>Specify either For, Do, or Function as a parameter of Exit statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3231</td>
<td>Above/Below was not specified.</td>
<td>Setting value for the elbow orientation is not specified in Elbow statement. Specify either Above or Below.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3232</td>
<td>Righty/Lefty was not specified.</td>
<td>Setting value for the hand orientation is not specified in Hand statement. Specify either Righty or Lefty.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3233</td>
<td>NoFlip/Flip was specified.</td>
<td>Setting value for the wrist orientation is not specified in Wrist statement. Specify either NoFilip or Flip.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3234</td>
<td>Port number was not specified.</td>
<td>Port number that indicates the file or communication port is not specified in Read, ReadBin, Write, and WriteBin statements. Refer to SPEL+ Language Reference “Read Statement” and specify a proper file number or port number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3235</td>
<td>String type variable was not specified.</td>
<td>String type variable is not specified in the command that requires specification of string type variable as a parameter. Specify a valid string type variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3236</td>
<td>RS-232C port number was not specified.</td>
<td>RS-232C port number is not specified in OpenCom, CloseCom, and SetCom statements. Refer to SPEL+ Language Reference “OpenCom Statement” and specify a proper port number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3237</td>
<td>Network communication port number was not specified.</td>
<td>Network communication port number is not specified in OpenNet, CloseNet, SetNet, and WaitNet statement. Specify an integer from 201 to 216.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3238</td>
<td>Communication speed was not specified.</td>
<td>Communication speed (baud rate) is not specified in SetCom statement. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper baud rate.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3239</td>
<td>Data bit number was not specified.</td>
<td>Data bit length is not specified in SetCom statement. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper data bit length.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3240</td>
<td>Stop bit number was not specified.</td>
<td>Stop bit length is not specified in SetCom statement. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper stop bit length.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3241</td>
<td>Parity was not specified.</td>
<td>Parity is not specified in SetCom statement. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper parity.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3242</td>
<td>Terminator was not specified.</td>
<td>Terminator (end of send/receive line) is not specified in SetCom and SetNet statements. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper terminator.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3243</td>
<td>Hardware flow was not specified.</td>
<td>Hardware flow is not specified in SetCom statement. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper flow control.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-------</td>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>3244</td>
<td>Software flow was not specified.</td>
<td>Software flow is not specified in SetCom statement. Refer to SPEL+ Language Reference “SetCom Statement” and specify a proper flow control.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3245</td>
<td>None was not specified.</td>
<td>“NONE” is not specified for software flow control setting in SetNet statement. Specify “NONE”.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3246</td>
<td>Parameter 'O' or 'C' was not specified.</td>
<td>Open or close parameter for the end of a curve is not specified in Curve statement. Refer to SPEL+ Language Reference “Curve Statement” and specify a proper open/close parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3247</td>
<td>NumAxes parameter was not specified.</td>
<td>The number of coordinate axes controlled during a curve motion is not specified in Curve statement. Refer to SPEL+ Language Reference “Curve Statement” and specify a proper number of the coordinate axes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3248</td>
<td>J4Flag value (0-1) was not specified.</td>
<td>Specify 0 or 1, or an expression for J4Flag value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3249</td>
<td>J6Flag value (0-127) was not specified.</td>
<td>Specify an integer from 0 to 127, or an expression for J6Flag value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3250</td>
<td>Array variable was not specified.</td>
<td>Array variable is not specified in the statement that requires specification of array variable. Specify a valid array variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3251</td>
<td>String Array variable was not specified.</td>
<td>Array which stores a token must be a string array variable in ParseStr statement and ParseStr function. Specify a string array variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3252</td>
<td>Device ID was not specified.</td>
<td>Device ID is not specified in DispDev statement or Cls command. Refer to SPEL+ Language Reference “DispDev Statement” and specify a proper device ID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3253</td>
<td>I/O type was not specified.</td>
<td>I/O type is not specified in IOLabel$ function. Refer to SPEL+ Language Reference “IOLabel$ Function” and specify a proper I/O type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3254</td>
<td>I/O bit width was not specified.</td>
<td>I/O bit size (i/O port width) is not specified in IODef, IOLabe function. Refer to SPEL+ Language Reference “IODef Function” and specify a proper I/O bit size.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3255</td>
<td>ByRef was not specified.</td>
<td>Although the ByRef is specified in the function declaration, no ByRef is specified for calling. Specify the ByRef parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3256</td>
<td>Variable type was not specified.</td>
<td>Variable type is not specified in Global statement. Specify a proper variable type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3257</td>
<td>Condition expression does not evaluate to Boolean value.</td>
<td>Condition expression in If, ElseIf, Do, and Loop statement must return a Boolean value. Correct the condition expression to return a Boolean value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3258</td>
<td>RS232C port number was not specified.</td>
<td>RS-232C port number is not specified in ChkCom function. Refer to SPEL+ Language Reference “ChkCom Function” and specify a proper port number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3259</td>
<td>Network communication port number was not specified.</td>
<td>Network communication port number is not specified in ChkNet function. Refer to SPEL+ Language Reference “ChkNet Function” and specify a proper port number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3260</td>
<td>Language ID was not specified.</td>
<td>Language ID is not specified in ErrMsg$ function. Refer to SPEL+ Language Reference “ErrMsg$ Function” and specify a proper language ID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3261</td>
<td>Expected ‘.’.</td>
<td>Add ‘.’</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3262</td>
<td>Vision Sequence Name was not specified.</td>
<td>Vision sequence name is not specified in Vision Guide commands such as VSet, VGet, and VRun. Add a sequence name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3263</td>
<td>Vision Sequence Name or Calibration Name was not specified.</td>
<td>Vision sequence name or calibration name is not specified in VSet and VGet statements. Add a sequence name or calibration name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3264</td>
<td>Vision Property Name or Result Name was not specified.</td>
<td>Vision property name or result name is not specified in VSet and VGet statements. Add a property name or result name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3265</td>
<td>Vision Property Name, Result Name or Object Name was not specified.</td>
<td>Either of Vision property name, result name, or object name is not specified in VSet and VGet statements. Add either of a property name, result name, or object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3266</td>
<td>Vision Calibration Property Name was not specified.</td>
<td>Vision calibration property name is not specified in VSet and VGet statements. Add a property name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3267</td>
<td>Task type was not specified.</td>
<td>Task type is not specified in Xqt statement. Refer to SPEL+ Language Reference “Xqt Statement” and specify a proper task type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3268</td>
<td>Form name was not specified.</td>
<td>Form name is not specified in GSet, GGet, GShow, GShowDialog, and GClose statements. Specify a form name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3269</td>
<td>Property Name or Control Name was not specified.</td>
<td>Property name or control name is not specified in GSet and GGet statements. Specify a property name or control name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3270</td>
<td>Property Name was not specified.</td>
<td>Property name is not specified in GSet and GGet statements. Specify a property name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3271</td>
<td>BackColorMode was not specified.</td>
<td>BackColorMode property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “BackColorMode Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3272</td>
<td>BorderStyle was not specified.</td>
<td>BorderStyle property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “BorderStyle Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3273</td>
<td>DropDownStyle was not specified.</td>
<td>DropDownStyle property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “DropDownStyle Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3274</td>
<td>EventTaskType was not specified.</td>
<td>EventTaskType property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “EventTaskType Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3275</td>
<td>ImageAlign was not specified.</td>
<td>ImageAlign property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “ImageAlign Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3276</td>
<td>IOType was not specified.</td>
<td>IOType property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “IOType Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3277</td>
<td>FormBorderStyle was not specified.</td>
<td>FormBorderStyle property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “FormBorderStyle Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3278</td>
<td>ScrollBars was not specified.</td>
<td>ScrollBars property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “ScrollBars Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3279</td>
<td>SizeMode was not specified.</td>
<td>SizeMode property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “SizeMode Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3280</td>
<td>StartPosition was not specified.</td>
<td>StartPosition property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “StartPosition Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3281</td>
<td>TextAlign was not specified.</td>
<td>TextAlign property setting value is not specified in GSet statement. This error occurs when the control type cannot be identified because the control is specified by a string variable. Refer to GUI Builder 7.0 manual “TextAlign Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3282</td>
<td>TextAlign was not specified.</td>
<td>TextAlign property setting value is not specified in GSet statement. This error occurs when the control is a text box. Refer to GUI Builder 7.0 manual “TextAlign Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3283</td>
<td>TextAlign was not specified.</td>
<td>TextAlign property setting value is not specified in GSet statement. This error occurs when the control is other than a text box. Refer to GUI Builder 7.0 manual “TextAlign Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3284</td>
<td>WindowState was not specified.</td>
<td>WindowState property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 manual “WindowState Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3285</td>
<td>J1FLAG was not specified.</td>
<td>Specify 0 or 1, or an expression for J1Flag value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3286</td>
<td>J2FLAG was not specified.</td>
<td>Specify 0 or 1, or an expression for J2Flag value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3287</td>
<td>Robot ID was not specified.</td>
<td>Specify a robot number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3288</td>
<td>Robot ID/All was not specified.</td>
<td>Robot number or All is not specified in InsideBox and InsidePlane function. Specify a robot number which performs intrusion detection, or All.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3289</td>
<td>Area ID was not specified.</td>
<td>Area number is not specified in InsideBox and InsidePlane function. Specify an approach check area number which returns status by an integer from 1 to 15.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3290</td>
<td>File number was not specified.</td>
<td>File number is not specified in the command related to file management. Specify a file number by an integer from 30 to 63 or an expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3292</td>
<td>Database type was not specified.</td>
<td>Database type is not specified in OpenDB statement. Refer to SPEL+ Language Reference “OpenDB Statement” and specify a proper database type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3293</td>
<td>Disk type was not specified.</td>
<td>Type of the disk that is subject to file manipulation is not specified in ChDisk statement. Refer to SPEL+ Language Reference “ChDisk Statement” and specify a proper disk type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3295</td>
<td>Conveyor area ID was not specified.</td>
<td>Area ID that is subject to count the queue data is not specified in Cnv_QueLen function. Refer to SPEL+ Language Reference “Cnv_QueLen Function” and specify a proper area ID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>3296</td>
<td>Database file number was not specified.</td>
<td>Data base number that is subject to operation is not specified in OpenDB, CloseDB, DeleteDB, UpdateDB, and SelectDB function. Refer to SPEL+ Language Reference “OpenDB Statement” and specify a proper database number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3297</td>
<td>Vision calibration name was not specified.</td>
<td>Calibration name is not specified in VCal statement. Specify a name of calibration that is subject to calibrate.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3298</td>
<td>Vision object type ID was not specified.</td>
<td>Vision object type is not specified in VCreateObject statement. Refer to Vision Guide 7.0 Properties &amp; Results Reference “VCreateObject Statement” and specify a proper object type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3299</td>
<td>Shutdown mode ID was not specified.</td>
<td>Shutdown mode value is not specified in ShutDown statement and ShutDown function. Refer to SPEL+ Language Reference “ShutDown Statement” and specify a proper mode value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3301</td>
<td>Version of linked OBJ file does not match.</td>
<td>Not all project files are compiled in the same version. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3302</td>
<td>Linked OBJ file does not match the compiled I/O label.</td>
<td>The project configuration has been changed. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3303</td>
<td>Linked OBJ file does not match the compiled user error label.</td>
<td>The project configuration has been changed. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3304</td>
<td>Linked OBJ file does not match the compiled compile option.</td>
<td>The project configuration has been changed. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3305</td>
<td>Linked OBJ file does not match the compiled link option.</td>
<td>The project configuration has been changed. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3306</td>
<td>Linked OBJ file does not match the compiled SPEL option.</td>
<td>The project configuration has been changed. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3307</td>
<td>Duplicate function.</td>
<td>The same function name is used for more than one file. Correct the program (function name).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3308</td>
<td>Duplicate global preserve variable.</td>
<td>The same global preserve variable name is used for more than one file. Correct the program (variable name).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3309</td>
<td>Duplicate global variable.</td>
<td>The same global variable name is used for more than one file. Correct the program (variable name).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3310</td>
<td>Duplicate module variable.</td>
<td>The same module variable name is used for more than one file. Correct the program (variable name).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3311</td>
<td>File cannot be found.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3312</td>
<td>OBJ file is corrupt.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3313</td>
<td>The specified file name includes character(s) that cannot be used.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3314</td>
<td>Cannot open the file.</td>
<td>The file is used for other application. Quit the other application.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3315</td>
<td>'***' is already used for a function name.</td>
<td>Correct the identifier '***' or the function name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3316</td>
<td>'***' is already used for a global preserve variable name.</td>
<td>Correct the identifier '***' or the global preserve variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3317</td>
<td>'***' is already used for a global variable name.</td>
<td>Correct the identifier '***' or the global variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3318</td>
<td>'***' is already used for a module variable name.</td>
<td>Correct the identifier '***' or the module variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3319</td>
<td>Dimension of the array variable does not match the declaration.</td>
<td>Correct the dimension of the array and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3320</td>
<td>Return value type of the function does not match the declaration.</td>
<td>Correct the return value type of the function and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3321</td>
<td>'***' is already used with function name.</td>
<td>Correct the identifier '***' or the function name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3322</td>
<td>'***' is already used with Global Preserve name.</td>
<td>Correct the identifier '***' or the global preserve variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3323</td>
<td>'***' is already used with Global name.</td>
<td>Correct the identifier '***' or the global variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3324</td>
<td>'***' is already used with Module name.</td>
<td>Correct the identifier '***' or the module variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3325</td>
<td>'***' is already used with Local name.</td>
<td>Correct the identifier '***' or the local variable name. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3326</td>
<td>The number of parameters does not match the declaration.</td>
<td>Check the number of parameters in the function, correct the program, and then rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3327</td>
<td>ByRef was not specified in Function declaration for parameter **.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3328</td>
<td>ByRef was not specified for parameter **.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3329</td>
<td>Parameter ** type mismatch.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3330</td>
<td>Linked OBJ file does not match the compiled Vision Project.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3331</td>
<td>OBJ file size is beyond the available size after linking.</td>
<td>The OBJ file size exceeds the limit value (8MB). Reduce the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3332</td>
<td>Variable '***' is redefined.</td>
<td>Variable '***' is overloaded. Delete unnecessary variable definition and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3333</td>
<td>Linked OBJ file does not match the compiled GUI Builder Project.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3334</td>
<td>The number of variables which is using Wait command is beyond the maximum allowed.</td>
<td>The number of variables which is using Wait command is exceeding the maximum allowed (64). Delete the variables and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3335</td>
<td>Call cannot be used in parallel processing.</td>
<td>Call cannot be used in parallel processing. Correct the program and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3336</td>
<td>Variable was redefined.</td>
<td>Correct the data type of the variable and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3351</td>
<td>Invalid object index was specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3352</td>
<td>Force Guide Sequence Name was not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3353</td>
<td>Force Guide Property Name or Result Name was not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3354</td>
<td>Force Guide Property Name, Result Name or Object Name was not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3355</td>
<td>Force Guide project file has unsupported file format.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3356</td>
<td>Linked OBJ file does not match the compiled Force Guide Project.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3400</td>
<td>Dialog ID was not specified.</td>
<td>Dialog ID is not specified in RunDialog statement. Refer to SPEL+ Language Reference “RunDialog Statement” and specify a dialog ID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3401</td>
<td>Main function name was not specified.</td>
<td>Name of the main function to execute is not specified in StartMain statement. Specify a main function name (main to main63).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3402</td>
<td>Vision object name was not specified.</td>
<td>Vision object name is not specified in VLoadModel, VSaveModel, VShowModel, VTeach, and VTrain statements. Specify an object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3403</td>
<td>Recover mode ID was not specified.</td>
<td>Recover mode is not specified in Recover statement or Recover function. Refer to SPEL+ Language Reference “Recover Statement” and specify a proper mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3404</td>
<td>Trap condition was not specified.</td>
<td>Trap number or trap event is not specified in Trap statement. Refer to SPEL+ Language Reference “Trap Statement” and specify a proper trap number or event.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3405</td>
<td>DialogResult was not specified.</td>
<td>DialogResult property setting value is not specified in GSet statement. Refer to GUI Builder 7.0 “DialogResult Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3406</td>
<td>MsgBox_Type was not specified.</td>
<td>Display type is not specified in MsgBox statement. Refer to SPEL+ Language Reference “MsgBox Statement” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3407</td>
<td>Byte type array variable was not specified.</td>
<td>Byte type array variable is not specified for send or receive data in FbusIO_SendMsg statement. Send/receive data must be specified by Byte type array.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3408</td>
<td>Single array variable was not specified.</td>
<td>The number of dimensions is not proper in the command where single array variable is only available. Correct the number of dimensions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3409</td>
<td>Point list is not specified.</td>
<td>Pixel coordinate or robot coordinate is not specified as a continuous point data in VxCalib statement. Specify a continuous point data in the following format: P (start : end)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>3410</td>
<td>Code type is not specified.</td>
<td>CodeType property setting value is not specified in VSet statement. Refer to Vision Guide 7.0 Properties &amp; Results Reference “CodeType Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3411</td>
<td>Edge type is not specified.</td>
<td>EdgeType property setting value is not specified in VSet statement. Refer to Vision Guide 7.0 Properties &amp; Results Reference “EdgeType Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3412</td>
<td>ECC type is not specified.</td>
<td>ErrorCorrection property setting value is not specified in VSet statement. This error is for the old RC+6.x. Setting of the old version is possible by compiler version setting. Refer to Vision Guide 6.0 Properties &amp; Results Reference “ErrorCorrection Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3413</td>
<td>ImageColor type is not specified.</td>
<td>ImageColor property setting value is not specified in VSet statement. Refer to Vision Guide 7.0 Properties &amp; Results Reference “ImageColor Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3414</td>
<td>Point type is not specified.</td>
<td>PointType property setting value is not specified in VSet statement. Refer to Vision Guide 7.0 Properties &amp; Results Reference “PointType Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3415</td>
<td>Reference type is not specified.</td>
<td>ReferenceType property setting value is not specified in VSet statement. Refer to Vision Guide 7.0 Properties &amp; Results Reference “ReferenceType Property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3416</td>
<td>Edge type is not specified.</td>
<td>Logic (edge type) of the trigger input is not specified in SetLatch statement. Specify either 0 (negative logic) or 1 (positive logic).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3417</td>
<td>Port number is not specified.</td>
<td>R-I/O input port number where the trigger input is connected is not specified in SetLatch statement. Refer to SPEL+ Language Reference “SetLatch Statement” and specify a proper port number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3418</td>
<td>Axis is not specified.</td>
<td>Axis parameter is not specified in Force_GetForce function or Force_SetTrigger statement. Refer to SPEL+ Language Reference “Force_GetForce Function” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3419</td>
<td>CompareType is not specified.</td>
<td>CompareType parameter to set judgment condition is not specified in Force_SetTrigger statement. Refer to SPEL+ Language Reference &quot;Force_SetTrigger Statement&quot; and specify a proper parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3420</td>
<td>Integer or Short type array variable is only available.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3421</td>
<td>Form name or window ID is not specified.</td>
<td>Form name or system window ID which is subject to operation is not specified in GShow and GClose statements. Specify a valid form name or window ID. For details of window ID, refer to GUI Builder7.0 manual &quot;GShow Statement&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3422</td>
<td>Window ID is not specified.</td>
<td>System window ID which is subject to operation is not specified in GShow and GClose statements. Refer to GUI Builder7.0 manual &quot;GShow Statement&quot; and specify a proper window ID.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3423</td>
<td>Performance mode ID was not specified.</td>
<td>Performance mode is not specified in PerformMode parameter of PerformMode statement, Go, BGo, TGo, Jump statement. Refer to SPEL+ Language Reference &quot;PerformMode&quot; and specify a proper performance mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3424</td>
<td>Protocol type was not specified.</td>
<td>Communication protocol setting is not specified in SetNet statement. Specify UDP or TCP.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3425</td>
<td>I/O type or I/O label was not specified.</td>
<td>I/O type or I/O label is not specified in IODef function. Specify the I/O label or I/O type to check existence of definition. For details of I/O types, refer to SPEL+ Language Reference &quot;IODef Function&quot;.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3426</td>
<td>Singularity avoidance mode was not specified.</td>
<td>Singularity avoidance mode is not specified in AvoidSingularity statement. Refer to SPEL+ Language Reference &quot;AvoidSingularity Statement&quot; and specify a proper mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3427</td>
<td>Acceleration value was not specified.</td>
<td>Setting number of acceleration is not specified in AccelR function. Refer to SPEL+ Language Reference &quot;AccelR Function&quot; and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3428</td>
<td>Acceleration value was not specified.</td>
<td>Setting number of acceleration is not specified in Accel function, AccelMax function, AccelS function, and RealAccel function. Refer to SPEL+ Language Reference &quot;Accel Function&quot; and specify a proper number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3429</td>
<td>Sorting order for work queue data was not specified.</td>
<td>Sorting order for work queue data is not specified in WorkQue_Sort statement. Refer to SPEL+ Language Reference “WorkQue_Sort Statement” and specify a proper sorting order.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3430</td>
<td>Coordinate axes number was not specified.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3431</td>
<td>Coordinate axes number was not specified.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3432</td>
<td>Point or point expression is not specified.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3433</td>
<td>Boolean type array variable was not specified.</td>
<td>Array which stores a value of Enabled or LPF_Enabled property must be a Boolean type array variable in FGet statement. Specify a Boolean array variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3434</td>
<td>Real or Double type array variable was not specified.</td>
<td>Real or Double type array variable is not specified in FGet or MPGet statement. Specify a real or double type array variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3435</td>
<td>Integral type array variable was not specified.</td>
<td>Array which stores a value of Polarities property must be an Integral type array variable in FGet statement. Specify an integral type array variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3436</td>
<td>Duration of FCKeep statement is not specified.</td>
<td>Duration of force control (timeout value) is not specified in FCKeep statement. Specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3437</td>
<td>Part kind of controller was not specified.</td>
<td>Specify the controller part type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3438</td>
<td>Part kind of robot was not specified.</td>
<td>Specify the robot part type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3439</td>
<td>Part kind of robot was not specified.</td>
<td>Specify the robot part type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3440</td>
<td>A value other than numerical value is specified for the parameter of the command. Specify a numerical value.</td>
<td>Specify a numerical value for AIO_TrackingSet command 7th parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3441</td>
<td>A value other than numerical value is specified for the parameter of the command. Specify a numerical value.</td>
<td>Specify a numerical value for AIO_TrackingSet command 6th parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3450</td>
<td>Force property name or status name is not specified.</td>
<td>Force property name or status name is not specified in FSet, FGet, MPSet, and MPGet statements. Add a property name or a status name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3451</td>
<td>Force property name, status name, or object name is not specified.</td>
<td>Either of Force property name, status name, or object name is not specified in FSet, FGet, MPSet, and MPGet statements. Add either of a property name, status name, or object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3452</td>
<td>Force object name is not specified.</td>
<td>Reboot the controller.</td>
<td>Initialize the controller firmware.</td>
<td>Replace the controller.</td>
</tr>
<tr>
<td></td>
<td>Add a force object name.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3453</td>
<td>Mass Property Object is not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Mass Property Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3454</td>
<td>Force Coordinate System Object is not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Coordinate System Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3455</td>
<td>Force Control Object is not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Control Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3456</td>
<td>Force Monitor Object is not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Monitor Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3457</td>
<td>Force Trigger Object is not specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Trigger Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3458</td>
<td>Force Control Object or Force Coordinate System Object is not specified.</td>
<td>Force Control data or Force Coordinate System data is not specified in FCSMove statement.</td>
<td>Specify a valid Force Control object or Force Coordinate System Object.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Control Object or Force Coordinate System Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3459</td>
<td>Force object is not specified.</td>
<td></td>
<td>Specify a Force object.</td>
<td></td>
</tr>
<tr>
<td>3460</td>
<td>Force object label is not specified.</td>
<td></td>
<td>Specify a Force object label.</td>
<td></td>
</tr>
<tr>
<td>3461</td>
<td>Force object or label is not specified.</td>
<td></td>
<td>Specify a Force object or label.</td>
<td></td>
</tr>
<tr>
<td>3462</td>
<td>Force Coordinate System Object or label is not specified.</td>
<td></td>
<td>Specify a valid Force Coordinate System object or label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Coordinate System Object or label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3463</td>
<td>Force Control Object or label is not specified.</td>
<td></td>
<td>Specify a valid Force Control object or label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Control Object or label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3464</td>
<td>Force Monitor Object or label is not specified.</td>
<td></td>
<td>Specify a valid Force Monitor object or label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Monitor Object or label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3465</td>
<td>Force Trigger Object or label is not specified.</td>
<td></td>
<td>Specify a valid Force Trigger object or label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Trigger Object or label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3466</td>
<td>Mass Property Object or label is not specified.</td>
<td></td>
<td>Specify a valid Mass Property object or label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Mass Property Object or label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3467</td>
<td>Force Coordinate System Object or label is not specified.</td>
<td></td>
<td>Specify a valid Force Coordinate System object or label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Coordinate System Object or label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3468</td>
<td>Force Control Object label is not specified.</td>
<td></td>
<td>Specify a valid Force Control object label.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a Force Control Object label.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3469</td>
<td>Force Monitor Object label is not specified. Specify a Force Monitor Object label.</td>
<td>Specify a valid Force Monitor object label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3470</td>
<td>Force Trigger Object label is not specified. Specify a Force Trigger Object label.</td>
<td>Specify a valid Force Trigger object label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3472</td>
<td>Mass Property Object label is not specified. Specify a Mass Property Object label.</td>
<td>Specify a valid Mass Property object label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3473</td>
<td>Mass Property Object label is not specified. Specify a Mass Property Object label.</td>
<td>Specify a valid Mass Property object label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3474</td>
<td>Fmag_Axes or Tmag_Axes property setting value is not specified. Specify a proper setting value.</td>
<td>Fmag_Axes or Tmag_Axes property setting value is not specified in FSet statement. Refer to “Fmag_Axes property” or “Tmag_Axes property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3475</td>
<td>TriggerMode property setting value is not specified. Specify a proper setting value.</td>
<td>TriggerMode property setting value is not specified in FSet statement. Refer to “TriggerMode property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3476</td>
<td>Operator property setting value is not specified. Specify a proper setting value.</td>
<td>Operator property setting value is not specified in FSet statement. Refer to “Operator property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3477</td>
<td>Orientation property setting value is not specified. Specify a proper setting value.</td>
<td>Orientation property setting value is not specified in FSet statement. Refer to “Orientation property” and specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3478</td>
<td>Polarity property setting value is not specified. Specify a proper setting value.</td>
<td>Polarity property setting value is not specified in FSet statement. Refer to “Fmag_Polarity, Fx_Polarity, Fy_Polarity, Fz_Polarity, Tmag_Polarity, Tx_Polarity, Ty_Polarity, and Tz_Polarity” property setting value is not specified in FSet statement. Specify a proper setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3500</td>
<td>Duplicate macro in #define statement.</td>
<td>Another macro with the same name has been defined. Change the macro name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3501</td>
<td>Macro name was not specified.</td>
<td>Macro name is not specified in #define, #ifdef, #ifndef, and #undef statements. Add a macro name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3502</td>
<td>Include file name cannot be found.</td>
<td>Include file name is not specified in #include statement. Add a valid include file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3503</td>
<td>Specified include file is not in the project.</td>
<td>The include file that is not registered in the project configuration is specified. Add the include file to the project configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3504</td>
<td>Parameter of the macro function does not match the declaration.</td>
<td>Check the number of parameters and correct the macro function.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3505</td>
<td>Macro has a circular reference.</td>
<td>The macro has a circular reference. Correct the circular reference.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3506</td>
<td>#define, #ifdef, #ifndef, #else, #endif, and variable declaration statements are only valid in an include file.</td>
<td>Check and correct the content of include file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3507</td>
<td>Over #ifdef or #ifndef nesting level.</td>
<td>The number of #endif statements that correspond to #ifdef and #ifndef statements is too many. Delete #endif statements or add the #ifdef and #ifndef statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3508</td>
<td>Cannot find corresponding #ifdef or #ifndef.</td>
<td>The number of #endif statements that correspond to #ifdef and #ifndef statements is not enough. Add the #endif statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3509</td>
<td>No #endif found for #ifdef or #ifndef.</td>
<td>The number of #endif statements that correspond to #ifdef and #ifndef statements is not enough. Add the #endif statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3510</td>
<td>Cannot obtain the macro buffer.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3550</td>
<td>Parameter for the macro function was not specified.</td>
<td>The macro declared as a macro function is called without argument. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3600</td>
<td>Tracking motion command cannot use Sense parameter.</td>
<td>When the queue data is specified in Jump, Jump3, and Jump3CP statements, Sense parameter cannot be specified. Delete the Sense statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3601</td>
<td>Parameter type is mismatch for the external function ' ** '. Check all statements where this function is called in this file.</td>
<td>LJM parameter cannot be specified in BGo, TGo, Arc, Arc3, BMove, Move, and TMove statements. Delete the LJM parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3602</td>
<td>The specified motion command cannot use LJM parameter.</td>
<td>InReal function cannot be used with Wait statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3603</td>
<td>InReal function cannot be used with Wait statement.</td>
<td>PerformMode parameter cannot be specified in Jump3, Jump3CP, Arc, Arc3, BMove, Move, and TMove statements. Delete the PerformMode parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3605</td>
<td>The specified motion command cannot use PerformMode parameter.</td>
<td>LJM parameter cannot be specified in BGo, TGo, Arc, Arc3, BMove, Move, and TMove statements. Delete the LJM parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3606</td>
<td>Cannot use the index.</td>
<td>Index number cannot be specified except List property in GSet and GGet statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3607</td>
<td>Invalid object index was specified.</td>
<td>Invalid index is specified in Objects property of VSet and VGet statements. The index must be larger than 1 and smaller than the number of objects in the specified sequence. Specify a proper index.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3608</td>
<td>Invalid control index was specified.</td>
<td>Invalid index is specified in Controls property of GSet and GGet statements. The index must be larger than 1 and smaller than the number of controls in the specified form. Specify a proper index.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3609</td>
<td>Modifier parameters are duplicated. Review the program.</td>
<td>Force Guide data or CF parameter is duplicated in CVMove, FCKeep or other statement for robot motion. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3610</td>
<td>Cannot use a keyword for a label name.</td>
<td>Keywords such as a command or a function are used for a label name. Change the label name which does not use these keywords.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3733</td>
<td>Vision Sequence Name or Calibration Name was not specified.</td>
<td>Vision sequence name or calibration name is not specified in VSet and VGet statements. Add a sequence name or calibration name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3800</td>
<td>Compile process aborted.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3801</td>
<td>Link process aborted.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3802</td>
<td>Compile process aborted. Compile errors reached the maximum count.</td>
<td>Correct the error in the program and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3803</td>
<td>Link process aborted. Link errors reached the maximum count.</td>
<td>Correct the error in the program and rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3804</td>
<td>Specified command cannot be executed from the Command window.</td>
<td>Declaration of variables and functions, program control statement, preprocessor commands, and some commands cannot be executed from the command window. For details, refer to SPEL+ Language Reference “Appendix A: SPEL+ Command Use Condition List”.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3805</td>
<td>Specified command can only be executed from the Command window.</td>
<td>Brake, SysConfig, Where, Cnv_QueList, and WorlQue_List statements can only be executed from the command window. Delete these statements from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3806</td>
<td>Specified function cannot be executed from the Command window.</td>
<td>LogIn function cannot be executed from the command window even when used with Print statement. Use the function in the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3808</td>
<td>Specified syntax cannot be used in the current version.</td>
<td>LJM and PerformMode parameters of motion commands may not be specified depending on the compiler version. LJM parameter: 6.0.x.x or later PerformMode parameter: 7.0.4.x or later Check the compiler version from the project property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3809</td>
<td>Module variables cannot be used in the command window.</td>
<td>Module variable cannot be accessed from the command window. Check the input command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3812</td>
<td>Specified function cannot be used with a Remote User Output.</td>
<td>Functions for condition expression of the user defined remote output are limited. Refer to EPSON RC+7.0 User's Guide “11.8 User-defined Remote Output I/O” and specify a valid function.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3813</td>
<td>User defined label, function and variable cannot be used with a Remote User Output.</td>
<td>User defined label, function and variable cannot be used with condition expression of the user-defined remote output. Correct the condition expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3814</td>
<td>Object code size is beyond the available size.</td>
<td>A combination of multiple statements is exceeding the available size of the intermediate code which can be executed at once (1024 bytes). Divide the statements.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3815</td>
<td>Parameter cannot be specified for property or status in the command window. Delete the parameter and execute again.</td>
<td>When executing FGet or MPGet statement from a command window, a parameter cannot be specified to a property or status. Delete the parameter and execute again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3850</td>
<td>File not found.</td>
<td>Failed to read the point file which configures the project. Check the project folder if the file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3851</td>
<td>Point file not found.</td>
<td>Failed to read the I/O label file which configures the project. Check the project folder if the file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3852</td>
<td>I/O label file not found.</td>
<td>Failed to read the I/O label file which configures the project. Check the project folder if the file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3853</td>
<td>User error label file not found.</td>
<td>Failed to read the user error label file which configures the project. Check the project folder if the file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3854</td>
<td>Force file does not exist. Check the project folder if the file exists.</td>
<td>Failed to read the force file which configures the project. Check the project folder if the file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3860</td>
<td>I/O label file not supported format.</td>
<td>Regenerate the I/O label file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3861</td>
<td>User error label file has unsupported file format.</td>
<td>Regenerate the user error file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3862</td>
<td>Point file has unsupported file format.</td>
<td>Regenerate the point file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3863</td>
<td>Vision project file has unsupported file format.</td>
<td>Regenerate the vision sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3864</td>
<td>GUI Builder project file has unsupported file format.</td>
<td>Regenerate the GUI Builder form.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3865</td>
<td>OBJ file not supported format.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3866</td>
<td>Force file has unsupported file format.</td>
<td>Regenerate the force file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3870</td>
<td>Cannot specify Mass Property Object. Review the program.</td>
<td>Mass Property object cannot be specified in FSet, FGet, FDef, and FList statements, FDef, and FLabel$ functions. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>3871</td>
<td>Cannot specify Force Coordinate System Object. Review the program.</td>
<td>Force coordinate system object cannot be specified in Go, BGo, TGo, Jump, Jump3, Mode, BMove, TMove, Arc, Arc3 statement, MPSet, MPGet, MPDel, MPList statement, and MPDef, MPLabel$ function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3872</td>
<td>Cannot specify Force Control Object. Review the program.</td>
<td>Force control object cannot be specified in Go, BGo, TGo, Jump, Jump3 statement, and MPSet, MPGet MPDel, MPList statement, and MPDef, MPLabel$ function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3873</td>
<td>Cannot specify Force Monitor Object. Review the program.</td>
<td>Force monitor object cannot be specified in MPSet, MPGet, MPDel, MPList statement, and MPDef, MPLabel$ function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3874</td>
<td>Cannot specify Force Trigger Object. Review the program.</td>
<td>Force trigger object cannot be specified in MPSet, MPGet, MPDel, MPList statement, and MPDef, MPLabel$ function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3875</td>
<td>Cannot specify Force Sensor object. Review the program.</td>
<td>Force Sensor object cannot be specified in FDel, FList statement, FDef, FLabel$ function, MPSet, MPGet, MPDel, MPList statement, and MPDef, MPLabel$ function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3876</td>
<td>Cannot specify Robot object. Review the program.</td>
<td>Robot object cannot be specified in FDel, FList statement, FDef, FLabel$ function, MPSet, MPGet, MPDel, MPList statement, and MPDef, MPLabel$ function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3877</td>
<td>Cannot specify Force Control Object and Force Coordinate System Object at the same time. Review the program.</td>
<td>Force control object and Force coordinate system object cannot be specified at the same time in FCSMove statement. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3878</td>
<td>Cannot specify CF parameter. Delete the CF parameter.</td>
<td>CF parameter cannot be specified in Go, BGo, TGo, Jump, Jump3. Delete the CF parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3879</td>
<td>Cannot specify Mass Property Object label. Review the program.</td>
<td>Mass property object label cannot be specified in MPDel, and MPList statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3880</td>
<td>Cannot specify Force Coordinate System Object label. Review the program.</td>
<td>Force coordinate system object label cannot be specified in FDel and FList statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3881</td>
<td>Cannot specify Force Control Object label. Review the program.</td>
<td>Force control object label cannot be specified in FDel and FList statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>3882</td>
<td>Cannot specify Force Monitor Object label.</td>
<td>Force monitor object label cannot be specified in FDel and FList statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3883</td>
<td>Cannot specify Force Trigger Object label.</td>
<td>Force trigger object label cannot be specified in FDel and FList statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3884</td>
<td>Cannot specify Force Sensor Object label.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3885</td>
<td>Cannot specify Mass Property Object number.</td>
<td>Mass property object number cannot be specified in MPNumber function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3886</td>
<td>Cannot specify Force Coordinate System Object number.</td>
<td>Force coordinate system object number cannot be specified in FNumber function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3887</td>
<td>Cannot specify Force Control Object number.</td>
<td>Force control object number cannot be specified in FNumber function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3888</td>
<td>Cannot specify Force Monitor Object number.</td>
<td>Force monitor object number cannot be specified in FNumber function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3889</td>
<td>Cannot specify Force Trigger Object number.</td>
<td>Force trigger object number cannot be specified in FNumber function. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3890</td>
<td>Cannot specify Force Sensor Object number.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3891</td>
<td>Type of the specified two objects does not match. Specify the same type of the objects.</td>
<td>The data type of the first and the second parameter does not match in FDel, FList, MPDel, MPList statements. Correct the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3900</td>
<td>Cannot obtain the internal communication buffer.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3901</td>
<td>Buffer size is not enough.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3910</td>
<td>Undefined command was specified.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3911</td>
<td>Cannot enter the file name in the file name buffer.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3912</td>
<td>Cannot obtain the internal buffer.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3913</td>
<td>Cannot set priority.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3914</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3915</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3916</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3917</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3918</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3919</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3920</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3921</td>
<td>Invalid ICode.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3930</td>
<td>VDefTool Type ID is not specified.</td>
<td>Specify a numerical value for VDefTool command 2nd parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------</td>
<td>--------------------------------------------------------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>3931</td>
<td>VDefArm Type ID is not specified.</td>
<td>Specify a numerical value for VDefArm command 2nd parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3932</td>
<td>VDefArm ArmSetMode is not specified.</td>
<td>Specify a numerical value for VDefArm command 3rd parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3933</td>
<td>VDefLocal Type ID is not specified.</td>
<td>Specify a numerical value for VDefLocal command 2nd parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3934</td>
<td>VDefLocal CalibPlate Type ID is not specified.</td>
<td>Specify a numerical value for VDefLocal command 3rd parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3940</td>
<td>LatchPos Type ID is not specified.</td>
<td>Specify a numerical value for LatchPos command 1st parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3960</td>
<td>Neither Robot, Object nor ResetCollision were specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3961</td>
<td>Neither Hand nor Property were specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3962</td>
<td>Invalid Property was specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3963</td>
<td>Neither Robot nor Object were specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3964</td>
<td>Invalid Object was specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3965</td>
<td>Invalid Object index was specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3990</td>
<td>Analog I/O TCPSpeed Type is not specified.</td>
<td>Specify a numerical value for AIO_Set command 3rd parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>4001</td>
<td>Arm reached the limit of motion range.</td>
<td>Check the point to move, current point, and Range setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4002</td>
<td>Specified value is out of allowable range.</td>
<td>Review the setting parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4003</td>
<td>Motion device driver failure. Communication error within the motion control module.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4004</td>
<td>Motion device driver failure. Event waiting error within the motion control module.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4005</td>
<td>Current point position is above the specified LimZ value.</td>
<td>Lower the Z axis. Increase the specified LimZ value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4006</td>
<td>Target point position is above the specified LimZ value.</td>
<td>Lower the Z coordinate position of the target point. Increase the specified LimZ value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4007</td>
<td>Coordinates conversion error. The end/mid point is out of the motion area. Jogging to the outside of the motion area.</td>
<td>Check whether the coordinate out of the motion range is not specified.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4008</td>
<td>Current point position or specified LimZ value is out of motion range.</td>
<td>Change the specified LimZ value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4009</td>
<td>Motion device driver failure. Timeout error within motion control module.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4010</td>
<td>Specified Local coordinate was not defined.</td>
<td>Define the Local coordinate system.</td>
<td>Local number</td>
<td></td>
</tr>
<tr>
<td>4011</td>
<td>Arm reached the limit of XY motion range specified by XYLim statement.</td>
<td>Check the area limited by the XYLim statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4012</td>
<td>Upper limit value of Box is smaller than the lower limit value. Change the upper and lower limit values.</td>
<td>Calculation of the timing of Arch motion failed. Perform either of the following: - Check and modify Arch parameter - Disable Arch</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4013</td>
<td>Motion control module internal calculation error.</td>
<td>Execute MCAl. Make sure the MCOdr is set for the joint connected to the PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4014</td>
<td>MCAL was not completed.</td>
<td>Execute MCAl. Make sure the MCOdr is set for the joint connected to the PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4016</td>
<td>SFree statement was attempted for prohibited joint(s).</td>
<td>Due to robot mechanistic limitation, setting some joint(s) to servo free status is prohibited. Check the robot specifications.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4018</td>
<td>Communication error within the motion control module. Check sum error.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4021</td>
<td>Point positions used to define the Local are too close.</td>
<td>Set the distance between points more than 1μm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4022</td>
<td>Point coordinate data used to define the Local is invalid.</td>
<td>Match the coordinate data for the points to be specified.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4023</td>
<td>Cannot execute when the motor is in the off state.</td>
<td>Turn the motor power ON and then execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>4024</td>
<td>Cannot complete the arm positioning using the current Fine specification.</td>
<td>Check whether the robot does not generate vibration or all parts and screws are secured firmly. Increase the Fine setting value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4025</td>
<td>Cannot execute a motion command during emergency stop condition.</td>
<td>Clear the emergency stop condition and execute the motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4026</td>
<td>Communication error within the motion control module. Servo I/F failure.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4028</td>
<td>Communication error within the motion control module. Device driver status failure.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4030</td>
<td>Buffer for the average torque calculation has overflowed. Shorten the time interval from Atclr to Atrq.</td>
<td>Shorten the time interval from Atclr to Atrq less than about two minutes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4031</td>
<td>Cannot execute a motion command when the motor is in the off state.</td>
<td>Turn the motor power ON and then execute the motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4032</td>
<td>Cannot execute a motion command when one or more joints are in SFree state.</td>
<td>Set all joints to the SLock state and execute the motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4033</td>
<td>The specified command is not supported for Pulse Generator Board joints.</td>
<td>The specified command is not permitted for the joints with PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4034</td>
<td>Specified command is not supported for this robot model.</td>
<td>Remove the unsupported command from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4035</td>
<td>Only the tool orientation was attempted to be changed by the CP statement.</td>
<td>Set a move distance between points. Use the ROT modifier, SpeedR statement, and AccelR statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4036</td>
<td>Rotation speed of tool orientation by the CP statement is too fast.</td>
<td>Decrease the setting values for the SpeedS and AccelS statements. Use the ROT modifier, SpeedR statement, and AccelR statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4037</td>
<td>The point attribute of the current and target point positions differ for executing a CP control command.</td>
<td>Match the point attribute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4038</td>
<td>Two point positions are too close to execute the Arc statement.</td>
<td>Set the distance between points more than 1μm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4039</td>
<td>Three point positions specified by the Arc statement are on a straight line.</td>
<td>Use the Move statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4041</td>
<td>Motion command was attempted to the prohibited area at the backside of the robot.</td>
<td>Check the robot motion range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4042</td>
<td>Motion device driver failure. Cannot detect the circular format interruption.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4043</td>
<td>Specified command is not supported for this robot model or this joint type.</td>
<td>Remove the unsupported command from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4044</td>
<td>Curve failure. Specified curve form is not supported.</td>
<td>Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4045</td>
<td>Curve failure. Specified mode is not supported.</td>
<td>Specify the Curve mode properly. Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4046</td>
<td>Curve failure. Specified coordinate number is out of the allowable range.</td>
<td>The number of the available coordinate axes is 2, 3, 4, and 6. Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4047</td>
<td>Curve failure. Point data was not specified.</td>
<td>Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4048</td>
<td>Curve failure. Parallel process was specified before the point designation.</td>
<td>Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4049</td>
<td>Curve failure. Number of parallel processes is out of the allowable range.</td>
<td>Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4050</td>
<td>Curve failure. Number of points is out of the allowable range.</td>
<td>The number of available point numbers differs according to the curve form. Check the number of points again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4051</td>
<td>Curve failure. Local attribute and the point attribute of all specified points do not match.</td>
<td>Match the local and point flag for all the specified points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4052</td>
<td>Curve failure. Not enough memory to format the curve file.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4053</td>
<td>Curve failure. Failed to format the curve file.</td>
<td>Review the point data. Check whether adjacent two points do not overlap on the specified point line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4054</td>
<td>Curve failure. Curve file error</td>
<td>The Curve file is broken. Create a Curve file again with the Curve statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4055</td>
<td>Curve failure. No distance for curve file movement.</td>
<td>Review the point data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4056</td>
<td>Curve failure. Point positions for the Curve statement are too close.</td>
<td>Set the distance between two points adjacent to the specified point more than 0.001 mm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4058</td>
<td>Prohibited command while tracking was executed.</td>
<td>Remove the prohibited command from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4059</td>
<td>Executed encoder reset command while the motor is in the on state.</td>
<td>Turn the motor power OFF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4060</td>
<td>Executed an invalid command while the motor is in the on state.</td>
<td>Turn the motor power OFF.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4061</td>
<td>Specified parameter is in use.</td>
<td>You attempted to clear the currently specified Arm and Tool. Select other Arm and Tool and execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4062</td>
<td>Orientation variation is over 360 degrees.</td>
<td>You attempted to rotate the joint #J6 more than 360 degrees with a CP motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4063</td>
<td>Orientation variation of adjacent point is over 90 degrees.</td>
<td>On the specified point line by the Curve statement, set the orientation variation of U, V, and W coordinate values between two adjacent points to under 90 degrees.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4064</td>
<td>Cannot execute the orientation correction automatically.</td>
<td>On the specified point line, a curve cannot be created by automatic orientation correction. Change the specified point line so that the joint #J6 orientation variation decreases.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4065</td>
<td>Attempt to revolve J6 one rotation with the same orientation in CP</td>
<td>You attempted to rotate the joint #J6 more than 360 degrees with a CP motion command. You attempted to revolve the joint 6 one rotation with the same as motion start orientation. Change the target point so that the joint #J6 revolves less than one rotation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4066</td>
<td>Motion command was attempted in the prohibited area depended on joint</td>
<td>You attempted to move the joints to the robot's interference limited area.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>combination.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4068</td>
<td>ROT modifier parameter was specified for the CP motion command without</td>
<td>Delete the ROT from the CP motion command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>orientation rotation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4069</td>
<td>Specified ECP without selecting ECP in CP statement.</td>
<td>Specify a valid ECP.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4070</td>
<td>Specified ECP number does not match the ECP number used in curve file</td>
<td>Specify a valid ECP.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>creation.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4071</td>
<td>Attempted motion command during electromagnetic brake lock condition.</td>
<td>Release the electromagnetic brake</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4072</td>
<td>Initialization failure. Hardware monitor was not initialized.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Initialize the controller firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4073</td>
<td>Orientation variation of adjacent point is over 90 degrees.</td>
<td>Any of U, V, or W changes 90 degrees or more. Change the point or the orientation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4074</td>
<td>Motor type does not match the current robot setting.</td>
<td>Check whether the specified robot model is connected.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4075</td>
<td>Option is not active.</td>
<td>Enable the option.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4076</td>
<td>Point positions used to define the Plane are too close.</td>
<td>Set the distance between points more than 1 μm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4077</td>
<td>Point coordinate data used to define the Plane is invalid.</td>
<td>Match the coordinate data for the points to be specified.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4078</td>
<td>Only the additional ST axis was attempted to be changed by the CP</td>
<td>Use PTP motion commands in order to move the additional axis only.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>statement.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4079</td>
<td>Speed of additional ST axis by the CP statement is too fast.</td>
<td>Reduce the set values of SpeedS and AccelS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4080</td>
<td>Cannot execute when the Enable Switch is OFF.</td>
<td>Turn the Enable Switch ON and then execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4081</td>
<td>Error was detected during operation.</td>
<td>Check the PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the connection with the motor driver.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4082</td>
<td>Pulse Generator Board error was detected during operation.</td>
<td>Check the PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the connection with the motor driver.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the PG board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4083</td>
<td>MCAL did not complete in time.</td>
<td>Set PG parameter so that MCAL can complete within 120 seconds.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4084</td>
<td>Limit Sensor error was detected during operation.</td>
<td>Check the limit sensor.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>4085</td>
<td>Failed to change to specified location.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4086</td>
<td>Cannot execute because it is not dry run mode.</td>
<td>Change to the dry run mode and execute.</td>
</tr>
<tr>
<td>4087</td>
<td>Failed to format the playback file.</td>
<td>Check the amount of free space of the computer. Reboot the computer. Replace the computer.</td>
</tr>
<tr>
<td>4089</td>
<td>The time interval from HealthRBStart to HealthRBStop is too long or too short.</td>
<td>Set the time interval from HealthRBStart to HealthRBStop to be within 1 to 3600 seconds.</td>
</tr>
<tr>
<td>4090</td>
<td>HealthRBStop is executed without HealthRBStart.</td>
<td>Execute HealthRBStop after executing HealthRBStart. This error also occurs when HealthRBStop is executed again without executing HealthRBStart after HealthRBStop.</td>
</tr>
<tr>
<td>4091</td>
<td>Specified analog I/O channel does not exist.</td>
<td>Check the channel number. Mount the analog I/O option board.</td>
</tr>
<tr>
<td>4092</td>
<td>Specified analog output channel is used for a speed data output.</td>
<td>Execute after stopping the speed output of the specified channel.</td>
</tr>
<tr>
<td>4093</td>
<td>If the motion is paused during the singularity-avoiding, the motion cannot resume. Abort the motion command.</td>
<td>Abort the motion command.</td>
</tr>
<tr>
<td>4094</td>
<td>The current position is out of the motion range.</td>
<td>Either J1 or J2 axis is out of the motion range. Follow the procedures below and move the robot within the motion range. Use Pulse command and JTran to move the robot within the motion range. Move the robot within the motion range manually. (This error only occurs in RS series and N series.)</td>
</tr>
<tr>
<td>4096</td>
<td>Robot in use. Cannot execute the motion command when other tasks are using the robot.</td>
<td>The motion command for the robot cannot be simultaneously executed from more than one task. Review the program. This error cannot be recovered automatically by OnErr.</td>
</tr>
<tr>
<td>4099</td>
<td>Servo error was detected during operation.</td>
<td>Check if a 5000 number error is occurring in the system history. If the error is occurring, take measures for a 5000 number error.</td>
</tr>
<tr>
<td>4100</td>
<td>Communication error in motion control module. Cannot calculate the current point or pulse.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4101</td>
<td>Communication error in the motion control module. Cannot calculate the current point or pulse.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4103</td>
<td>Initialization failure. Motion control module initialization error.</td>
<td>T/VT series Manipulators: Reboot the controller and take the measure against noise. If the joint number is displayed in the system history, replace the motor unit. If not, replace the CPU board. Other Manipulators: Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4104</td>
<td>Positioning timeout of the joint connected to the Pulse Generator Board.</td>
<td>Cannot receive the positioning completion signal (DEND) from the servo motor connected to PG board.</td>
</tr>
<tr>
<td>4108</td>
<td>Initialization failure. Motor unit connection error.</td>
<td>Check the wiring of the motor unit.</td>
</tr>
<tr>
<td>4150</td>
<td>Redundant input signal failure of the emergency stop.</td>
<td>The input status of the redundant emergency stop input continuously differs for more than two seconds. Check whether no disconnection, earth fault, or short-circuit of the emergency stop input signal exits. Then reboot the controller.</td>
</tr>
<tr>
<td>4151</td>
<td>Redundant input signal failure of the safeguard.</td>
<td>The input status of the redundant emergency stop input continuously differs for more than two seconds. Check whether no disconnection, earth fault, or short-circuit of the emergency stop input signal exits. Then reboot the controller.</td>
</tr>
<tr>
<td>4152</td>
<td>Relay welding error of the main circuit.</td>
<td>A relay welding error was detected due to power system over current. Replace the controller. Replace the robot.</td>
</tr>
<tr>
<td>4153</td>
<td>Redundant input signal failure of the enable switch.</td>
<td>The input status of the redundant enable signal differs continuously for more than two seconds. Check the TP connector connection. Replace the TP. Replace the controller.</td>
</tr>
<tr>
<td>4154</td>
<td>Temperature of regeneration resistor was higher than the specified temperature.</td>
<td>Robot's Duty is too high. Lengthen the waiting time or reduce the Accel value. If the error occurs although Duty was lowered, replace the DPB.</td>
</tr>
<tr>
<td>4180</td>
<td>Robot initialization failure. Specified robot was is not found</td>
<td>Configure the manipulator.</td>
</tr>
<tr>
<td>4181</td>
<td>Robot initialization failure. Specified robot was in use by another task.</td>
<td>Specified manipulator cannot be configured since it is already configured.</td>
</tr>
<tr>
<td>4182</td>
<td>Robot initialization failure. Robot name is too long.</td>
<td>Shorten the manipulator name.</td>
</tr>
<tr>
<td>4187</td>
<td>Robot initialization failure. Communication error with the module: VSRCMNPK.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4188</td>
<td>Robot initialization failure. Joint angle interference matrix is invalid.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4189</td>
<td>Robot initialization failure. Communication error with the module: VSRCMC.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4191</td>
<td>Robot initialization failure. Physical-logical pulse transformation matrix is invalid.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4192</td>
<td>Robot initialization failure. Communication error with the servo module.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4210</td>
<td>RAS circuit detected a servo system malfunction. Reboot the controller. Check for noise. Replace the controller.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4211</td>
<td>Servo CPU internal RAM failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4212</td>
<td>RAM for the main and servo CPU communication failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4213</td>
<td>Servo CPU internal RAM failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4214</td>
<td>Initialization communication of main CPU and servo CPU failure. Reboot the Controller. Check for noise. Replace DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4215</td>
<td>Initialization communication of the main and servo CPU failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4216</td>
<td>Communication of the main and servo CPU failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4217</td>
<td>Communication of the main and servo CPU failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4218</td>
<td>Servo long time command overrun.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4219</td>
<td>Servo long time command check sum error.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4220</td>
<td>System watchdog timer detected a failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4221</td>
<td>Drive unit check failure.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4222</td>
<td>RAM failure of the servo CPU. Reboot the controller. Check for noise.</td>
<td>Reboot the controller, take the measure against noise, or replace the DMB.</td>
</tr>
<tr>
<td>4223</td>
<td>Failure of duplicate circuit of the emergency stop or the safeguard. Check the wiring.</td>
<td>Check the wiring of the emergency stop or the safeguard.</td>
</tr>
<tr>
<td>4224</td>
<td>Low voltage of the main circuit power supply is detected. Check the power supply voltage. Reboot the controller.</td>
<td>Check the power supply voltage, or reboot the controller.</td>
</tr>
<tr>
<td>4225</td>
<td>Control relay contact of the main circuit power supply is welded. Replace the DPB.</td>
<td>Replace the DPB.</td>
</tr>
<tr>
<td>4226</td>
<td>Detect the recognition mismatch of the sub CPU and main CPU.</td>
<td>Reboot the controller, take the measure against noise, or replace the CPU board.</td>
</tr>
<tr>
<td>4227</td>
<td>Temperature of regeneration resistor was higher than the specified temperature.</td>
<td>Specify the Weight/Inertia setting. Check the load. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection.</td>
</tr>
<tr>
<td>4228</td>
<td>Over voltage of the sub CPU.</td>
<td>Replace the DPB board.</td>
</tr>
<tr>
<td>4230</td>
<td>Servo real time status failure. Check sum error.</td>
<td>A data checksum error was detected in the controller. Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency, D-I/O, and Expansion I/O connectors) Replace the controller.</td>
</tr>
<tr>
<td>4232</td>
<td>Servo real time status failure. Free running counter error with the servo.</td>
<td>A free running counter error was detected in the controller. Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency, D-I/O, and Expansion I/O connectors) Replace the controller.</td>
</tr>
<tr>
<td>4233</td>
<td>Servo real time status failure. Communication error with the servo CPU.</td>
<td>A communication error was detected in the controller. Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency, D-I/O, and Expansion I/O connectors) Replace the controller.</td>
</tr>
<tr>
<td>4240</td>
<td>Irregular motion control interruption was detected. Interruption duplicate.</td>
<td>A interruption error was detected in the controller. Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency, D-I/O, and Expansion I/O connectors) Replace the controller.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4241</td>
<td>Over speed during low power mode was detected.</td>
<td>The robot over speed was detected during low power mode. Check the robot mechanism. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check whether the robot does not interfere with peripheral equipment. (Collision, contact) Replace the motor driver. Replace the motor. (Motor and encoder failure) Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency, D-I/O, and Expansion I/O connectors)</td>
</tr>
<tr>
<td>4242</td>
<td>Improper acceleration reference was generated.</td>
<td>You attempted to operate the robot with the acceleration reference exceeding the specified value. For a CP motion, decrease the AccelS value.</td>
</tr>
<tr>
<td>4243</td>
<td>Improper speed reference was generated in high power mode.</td>
<td>The robot over speed was detected during high power mode. Check the robot mechanism. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check whether the robot does not interfere with peripheral equipment. (Collision, contact) Replace the motor driver. Replace the motor. (Motor and encoder failure) Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency, D-I/O, and Expansion I/O connectors)</td>
</tr>
<tr>
<td>4248</td>
<td>The robot collides with itself. Please change the target position.</td>
<td>Set a relay point. Or change the target point.</td>
</tr>
<tr>
<td>4250</td>
<td>Arm reached the limit of motion range during the operation.</td>
<td>Check whether a CP motion trajectory is within the motion range.</td>
</tr>
<tr>
<td>4251</td>
<td>Arm reached the limit of XY motion range specified by XYLim during the operation.</td>
<td>Check the XYLim setting.</td>
</tr>
<tr>
<td>4252</td>
<td>Coordinate conversion error occurred during the operation.</td>
<td>Check whether a CP motion trajectory is within the motion range.</td>
</tr>
<tr>
<td>4255</td>
<td>Because SpeedS is too big, the robot cannot pass elbow specific posture</td>
<td>Reduce the SpeedS value.</td>
</tr>
<tr>
<td>4256</td>
<td>When a robot passed elbow specific posture, Stop or Pause were carried out</td>
<td>Do not execute Stop or Pause.</td>
</tr>
<tr>
<td>4257</td>
<td>The robot cannot pass Singularity Area of Elbow</td>
<td>The robot cannot pass the elbow singularity area. To pass the elbow singularity area, use SING_AVOID of “AvoidSingularity”.</td>
</tr>
<tr>
<td>4261</td>
<td>The Arm reached the limit of motion range in conveyor tracking.</td>
<td>Place the conveyor inside the motion range. Meanwhile, allow the tracking</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4262</td>
<td>The Arm reached the limit of XY motion range in conveyor tracking.</td>
<td>range for the deceleration when switching from tracking motion to non-tracking. If error occurs during the shift from tracking motion, it may be prevented by increasing the accel speed to complete the tracking motion.</td>
</tr>
<tr>
<td>4263</td>
<td>The Arm reached the limit of pulse motion range in conveyor tracking.</td>
<td></td>
</tr>
<tr>
<td>4267</td>
<td>Attempt to exceed the J4Flag attribute without indication.</td>
<td>You attempted to exceed the J4Flag attribute during motion without the J4Flag indication. Change the J4Flag for the target point.</td>
</tr>
<tr>
<td>4268</td>
<td>Attempt to exceed the J6Flag attribute without indication.</td>
<td>You attempted to exceed the J6Flag attribute during motion without the J6Flag indication. Change the J6Flag for the target point.</td>
</tr>
<tr>
<td>4269</td>
<td>Attempt to exceed the particular wrist orientation attribute without indication.</td>
<td>You attempted to exceed the particular wrist orientation attribute during motion without the Wrist indication. Change the Wrist attribute for the target point. Change the target point to avoid a particular wrist orientation.</td>
</tr>
<tr>
<td>4270</td>
<td>Attempt to exceed the particular arm orientation attribute without indication.</td>
<td>You attempted to exceed the particular arm orientation attribute during motion without the Hand indication. Change the Hand attribute for the target point. Change the target point to avoid a particular hand orientation.</td>
</tr>
<tr>
<td>4271</td>
<td>Attempt to exceed the particular elbow orientation attribute without indication.</td>
<td>You attempted to exceed the particular elbow orientation attribute during motion without the Elbow indication. Change the Elbow attribute for the target point. Change the target point to avoid a particular elbow orientation.</td>
</tr>
<tr>
<td>4272</td>
<td>Specified point flag is invalid.</td>
<td>For a CP motion command, the arm form at the target point is different from the point flag specified with the target point. Change the point flag for the target point.</td>
</tr>
<tr>
<td>4273</td>
<td>J6Flag switched during the lift motion in conveyor tracking.</td>
<td>Adjust the Tool orientation so that J6Flag will not switch.</td>
</tr>
<tr>
<td>4274</td>
<td>Specified J6Flag is not reached. Change J6Flag for target point.</td>
<td>For a CP motion command, the manipulator reached to the target point with J6Flag which differs from the one specified for the target point. Change J6Flag for the target point.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>-----</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4275</td>
<td>Specified J4Flag is not reached. Change J4Flag for target point.</td>
<td>For a CP motion command, the manipulator reached to the target point with J4Flag which differs from the one specified for the target point. Change J4Flag for the target point.</td>
</tr>
<tr>
<td>4276</td>
<td>Specified ArmFlag is not reached. Change ArmFlag for target point.</td>
<td>For a CP motion command, the manipulator reached to the target point with ArmFlag which differs from the one specified for the target point. Change ArmFlag for the target point.</td>
</tr>
<tr>
<td>4277</td>
<td>Specified Elbow Flag is not reached. Change Elbow Flag for target point.</td>
<td>For a CP motion command, the manipulator reached to the target point with ElbowFlag which differs from the one specified for the target point. Change ElbowFlag for the target point.</td>
</tr>
<tr>
<td>4278</td>
<td>Specified WristFlag is not reached. Change WristFlag for target point.</td>
<td>For a CP motion command, the manipulator reached to the target point with WristFlag which differs from the one specified for the target point. Change WristFlag for the target point.</td>
</tr>
<tr>
<td>4279</td>
<td>Specified J1Flag is not reached. Change J1Flag for target point.</td>
<td>For a CP motion command, the manipulator reached to the target point with J1Flag which differs from the one specified for the target point. Change J1Flag for the target point.</td>
</tr>
<tr>
<td>4291</td>
<td>Data sending failure in motion network.</td>
<td>Check the connection of the cable for Drive Unit.</td>
</tr>
<tr>
<td>4292</td>
<td>Data receiving failure in motion network.</td>
<td>Check the connection of the cable for Drive Unit.</td>
</tr>
<tr>
<td>4297</td>
<td>Data sending failure of Force Sensor I/F board. Check connection of the Force Sensor I/F board and Force Sensor.</td>
<td>Check connection of the Force Sensor I/F board and Force Sensor. Reboot the controller. Please inquire with us if a similar error occurs even after rebooting the controller.</td>
</tr>
<tr>
<td>4298</td>
<td>Data receiving failure of Force Sensor I/F board. Check connection of the Force Sensor I/F board and Force Sensor.</td>
<td>Check connection of the Force Sensor I/F board and Force Sensor. Reboot the controller. Please inquire with us if a similar error occurs even after rebooting the controller.</td>
</tr>
<tr>
<td>4301</td>
<td>The Pulse Generating Board detected a limit signal.</td>
<td>Reset and then execute the next motion.</td>
</tr>
<tr>
<td>4302</td>
<td>The Pulse Generating Board detected an alarm signal.</td>
<td>Release the alarm of the pulse motor driver.</td>
</tr>
<tr>
<td>4401</td>
<td>The specified conveyor number is invalid.</td>
<td>Review the conveyor number.</td>
</tr>
<tr>
<td>4402</td>
<td>The specified conveyor queue is full.</td>
<td>The number of registration reached the upper limit (1000 pcs.) Delete the queue.</td>
</tr>
<tr>
<td>4403</td>
<td>Continue operation cannot be done in tracking motion.</td>
<td>Tracking operation cannot be continued after aborted/paused?</td>
</tr>
<tr>
<td>4404</td>
<td>The specified conveyor queue data does not exist.</td>
<td>Review the queue number. Or, check whether the queue is registered.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4405</td>
<td>The conveyor is not correctly initialized.</td>
<td>Rebuild the project. Delete the conveyor and then reestablish the setting.</td>
</tr>
<tr>
<td>4406</td>
<td>The specified queue data is outside the set area.</td>
<td>The queue outside of the range cannot be tracked. If the specified queue is above the upstream limit, change the program so that tracking does not start until the queue enters the area below the upper limit. If the specified queue is below the downstream limit, change the program to delete the queue data.</td>
</tr>
<tr>
<td>4407</td>
<td>The encoder is not correctly assigned.</td>
<td>Set the encoder.</td>
</tr>
<tr>
<td>4409</td>
<td>The parameter of the conveyor instruction is invalid.</td>
<td>Review the parameter.</td>
</tr>
<tr>
<td>4410</td>
<td>A conveyor coordinates conversion error occurred.</td>
<td>Rebuild the project. Delete the conveyor and then reestablish the setting.</td>
</tr>
<tr>
<td>4411</td>
<td>Communication error within the Conveyor Modules.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4413</td>
<td>Conveyor tracking starting error.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
</tr>
<tr>
<td>4414</td>
<td>Conveyor tracking cannot start after motion with CP ON.</td>
<td>Start the conveyor tracking using CP OFF.</td>
</tr>
<tr>
<td>4415</td>
<td>The setting of Diagonal Upstream Limit or Diagonal Downstream Limit is not appropriate.</td>
<td>The diagonal downstream limit is above the upstream limit, or the diagonal upstream/downstream limit is horizontal to the conveyor direction. Review the setting of diagonal upstream/downstream limit.</td>
</tr>
<tr>
<td>4500</td>
<td>Cannot execute the specified functions at the same time.</td>
<td>Review the program.</td>
</tr>
<tr>
<td>4501</td>
<td>Device is in use.</td>
<td>Check whether the device is used in other task or command.</td>
</tr>
</tbody>
</table>

Type of the functions
1: External control point motion (ECP)
2: Torque control
3: Conveyor tracking
4: Force control function
5: FCSMove
6: Distance tracking function

Type of the device
1: Analog I/O input channel

* Same as Note 1
<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>4502</td>
<td>Cannot execute this command during the control function. Note 1: Type of the control function. Exit the control function.</td>
<td>Check whether the control function is executed in other task or command. Exit the control function to execute the command.</td>
<td>Type of the control function 1: Distance tracking function</td>
<td></td>
</tr>
<tr>
<td>4503</td>
<td>An undefined parameter was specified. Note 1: Type of the parameter. Check the parameter.</td>
<td>Check whether the parameter is set. Set the parameter.</td>
<td>Type of the parameter 1: Parameter of the distance tracking function</td>
<td></td>
</tr>
<tr>
<td>4504</td>
<td>Option is not enabled. Check the option setting.</td>
<td>Enable the option.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4601</td>
<td>Failed to initialize the motion. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4602</td>
<td>Motion calculation error. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4603</td>
<td>Sensor value is out of range. Check the sensor.</td>
<td>Check the measured value by the sensor. Check the status of the sensor. Check the range setting of the sensor. When Note 1 is 1 (sensor used by the distance tracking function), check and adjust the parameter specified by AIO_TrackingStart or AIO_TrackingSet.</td>
<td>Type of the sensor 1: Sensor used by the distance tracking function</td>
<td></td>
</tr>
<tr>
<td>4604</td>
<td>Approached the singularity point. Avoid the singularity point.</td>
<td>Check whether the coordinates near the singularity is specified. Check whether the robot moves closer to the singularity during the operation. Review the installation position of the robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5000</td>
<td>Servo control gate array failure. Check the DMB.</td>
<td>Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency and I/O connectors) Replace the DMB. Replace the additional axis unit. For T series, reboot the controller, take the measure against noise, and replace the CPU board and motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5001</td>
<td>Disconnection of the parallel encoder signal. Check the signal cable connection or the robot internal wiring.</td>
<td>Check the M/C cable signal. Check the robot signal wiring. (Missing pin, disconnection, short-circuit) Replace the motor. Replace the DMB. Check the connector connection in the controller. (Loosening, connecting to the serial encoder terminal on the DMB) Check the model setting. Check the peripheral equipment wiring. (Emergency and I/O) For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5002</td>
<td>Motor driver is not installed. Install the motor driver. Check the DMB or the motor driver.</td>
<td>Check whether the motor driver is mounted. Check the model setting and hardware setting. Replace the motor driver. Replace the DMB. For T series, check the wiring of the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5003</td>
<td>Initialization communication failure of incremental encoder. Check the signal cable connection and the robot setting.</td>
<td>Check the model setting. Replace the motor. (Encoder failure) Replace the DMB. For T series, reboot the controller, take the measure against noise, and replace the CPU board and motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5004</td>
<td>Initialization failure of absolute encoder. Check the signal cable connection or the robot setting.</td>
<td>Check the model setting. Replace the motor. (Encoder failure) Replace the DMB. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5005</td>
<td>Encoder division setting failure. Check the robot setting.</td>
<td>Check the model setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5006</td>
<td>Data failure during absolute encoder initialization. Check the signal cable connection, the controller, or the motor.</td>
<td>Replace the motor. Replace the DMB. (Encoder failure) Check the noise countermeasures. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5007</td>
<td>Absolute encoder multi-turn is beyond the maximum range. Reset the encoder.</td>
<td>Reset the encoder. Replace the motor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5008</td>
<td>Position is out of the range. Reset the encoder.</td>
<td>Reset the encoder. Replace the DMB. Replace the motor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5009</td>
<td>No response from the serial encoder. Check the signal cable connection, the motor, the DMB, or the encoder IF board.</td>
<td>Check the model setting. (Improperly setting of the parallel encoder model) Check the signal cable connection. Replace the DMB and encoder I/F board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5010</td>
<td>Serial encoder initialization failure. Reboot the controller. Check the motor, the DMB, or the encoder IF board.</td>
<td>Check the robot configuration. Check the signal cable connection. Replace the DMB and encoder I/F board. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5011</td>
<td>Serial encoder communication failure. Reboot the controller. Check the motor, the DMB, or the encoder IF board.</td>
<td>Check the robot configuration. Check the signal cable connection. Replace the DMB and encoder I/F board. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5012</td>
<td>Servo CPU watchdog timer failure. Reboot the controller. Check the motor or the DMB.</td>
<td>Replace the DMB. Check the noise countermeasures. For T series, check the connection of the signal cable. Reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5013</td>
<td>Current control circuit WDT failure. Reboot the controller. Check the controller.</td>
<td>Check the power cable connection. Check the 15V power supply and cable connection. Replace the DMB. Check the noise countermeasures. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5014</td>
<td>The DMB is not for this robot.</td>
<td>Check robot setting. Replace with the supported DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5015</td>
<td>Encoder is reset. Reboot the controller.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5016</td>
<td>Power supply failure of the absolute encoder. Replace the battery. Check the robot internal wiring.</td>
<td>Reset the encoder. Check the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5017</td>
<td>Backup data failure of the absolute encoder. Reset the encoder.</td>
<td>Reset the encoder. Check the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5018</td>
<td>Absolute encoder battery alarm.</td>
<td>Replace the battery. Check the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5019</td>
<td>Position failure of the absolute encoder. Reset the encoder. Replace the motor.</td>
<td>Reset the encoder. Replace the motor. (Encoder failure) For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5020</td>
<td>Speed is too high at controller power ON. Stop the robot and reboot the controller.</td>
<td>Reboot the controller. Reset the encoder. For T series, reboot the controller, take the measure against noise, and replace the motor unit. Check the interference with the other devices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5021</td>
<td>Absolute encoder overheat.</td>
<td>Lower the motion duty. Wait until the temperature of the encoder decreases.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5022</td>
<td>R/D transducer failure. Reset the encoder. Check resolver board or the robot internal wiring.</td>
<td>Reset the encoder. Check the signal wiring of the manipulator (loose pin, disconnection, short). Replace the resolver board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5023</td>
<td>G sensor communication failure. Check for the signal cable connection or manipulator internal wiring.</td>
<td>Check the signal wiring connection. Check the signal wiring of the manipulator (loose pin, disconnection, short). Check the noise countermeasure. Replace the control board. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5024</td>
<td>G sensor data error. Check for the control board.</td>
<td>Replace the control board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5025</td>
<td>Gap occurred between multi-turn data and R/D conversion data. Encoder reset.</td>
<td>Reset the resolver. Check the noise countermeasure. Replace the resolver board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5026</td>
<td>Disconnection of the resolver excitation signal. Reset the encoder. Check the resolver board or the robot internal wiring.</td>
<td>Check the signal wiring of the manipulator (loose pin, disconnection, short). Replace the resolver board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5027</td>
<td>S-DSP detected the communication error in DSP. Check for DMB.</td>
<td>Reboot the controller. Check the noise countermeasure. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5028</td>
<td>Current feedback data error is detected. Check for DMB.</td>
<td>Reboot the controller. Check the noise countermeasure. Replace the DMB. For T series, check the short-circuit and earth fault of the power cable. Reboot the controller or replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5029</td>
<td>D-DSP communication failure. Check the DMB.</td>
<td>Reboot the controller. Check the noise countermeasure. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5030</td>
<td>Speed is too high at controller power OFF. Reset the encoder.</td>
<td>Reset the encoder. Replace the motor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5031</td>
<td>Speed is too high. Reset the encoder. Excess the calculation amount</td>
<td>Reset the encoder. Replace the motor. For T series, reboot the controller and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5032</td>
<td>Servo alarm A.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5040</td>
<td>Motor torque output failure in high power state. Check the power cable</td>
<td>Specify the Weight/Inertia setting. Check the load. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>connection, the robot, the driver or the motor.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5041</td>
<td>Motor torque output failure in low power state. Check the power cable</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>connection, robot, brake, driver, or motor.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>5042</td>
<td>Position error overflow in high power state. Check the power cable connection, the robot, the driver and the motor.</td>
<td>Specify the Weight/Inertia setting. Check the load. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5043</td>
<td>Position error overflow in low power state. Check the power cable connection, robot, brake, driver, or motor.</td>
<td>Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5044</td>
<td>Speed error overflow in high power state. Check the power cable connection, robot, brake, driver, or motor.</td>
<td>Specify the Weight/Inertia setting. Check the load. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5045</td>
<td>Speed error overflow in low power state. Check the power cable connection, robot, brake, drive, or motor.</td>
<td>Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5046</td>
<td>Over speed in high power state. Check the signal cable connection, robot, brake, driver or motor.</td>
<td>Reduce SpeedS of the CP motion. Change the orientation of the CP motion. Specify the Weight/Inertia setting. Check the load. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5047</td>
<td>Over speed in low power state. Check the signal cable connection, robot, brake, driver, or motor.</td>
<td>Check the motion in high power state. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor. For T series, replace the CPU board and motor unit in addition to the above.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5048</td>
<td>Over voltage of the main power circuit. Check the main power voltage or the regeneration module.</td>
<td>Specify the Weight/Inertia setting. Check the load. Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake) Check the interference with the peripheral equipment. (Collision, contact) Check the model setting. Check the power cable connection. Check the robot power wiring. (Missing pin, disconnection, short-circuit) Check the power supply voltage. (Low power supply voltage) Replace the motor driver. Replace the DMB. Replace the motor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5049</td>
<td>Over current of the motor driver. Check the power cable connection or the robot internal wiring.</td>
<td>Check the short-circuit and earth fault of the power line. Replace the motor driver. Replace the DMB. For T series, check the short-circuit and earth fault of the power cable. Reboot the controller or replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5050</td>
<td>Over speed during torque control. Check the work motion speed range.</td>
<td>Check the motion speed during torque control.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5051</td>
<td>15V PWM drive power supply failure. Reboot the controller. Replace the 15V power supply.</td>
<td>Check the 15V power supply and cable connection. Replace the motor driver. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5054</td>
<td>Overload of the motor. Decrease the motion duty and the Accel.</td>
<td>Lower the motion duty. Check the Weight/Inertia setting. Check the robot. (Backlash, large load, loose belt tension, brake)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5055</td>
<td>Overload of the motor. Decrease the operation duty and the Accel.</td>
<td>Lower the motion duty. Check the Weight/Inertia setting. Check the robot. (Backlash, large load, loose belt tension, brake)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5056</td>
<td>G sensor data has changed rapidly. Check the control board.</td>
<td>Check the noise countermeasure. Replace the control board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5057</td>
<td>Collision was detected in High power mode (Detection of robot motion error)</td>
<td>Collision detection (detection of robot motion error) was functioned. The following errors have detected: - Collision or contact of the robot arm occurs. - Torque saturation due to little setting of Weight or Inertia. - Torque saturation due to combined motion of multiple joints and throwing around the long object. - Torque saturation due to supply voltage reduction. - Error motion due to hardware error or software malfunction. Countermeasures: Check there is no collision or contact of the robot arm and change the arrangement to avoid interfere. Confirm that torque saturation is occurred. During torque saturation: check the setting of Weight and Inertia are properly and correct them if necessary. During combined motion: adjust the acceleration and deceleration to avoid torque saturation. Check the power supply voltage and correct them if necessary. If other error occurs at the same time, take a countermeasure for that first. Reference: EPSON RC+ 7.0 User's Guide &quot;6.18.10 Collision Detection Function (Error detection function of robot motion)&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5058</td>
<td>Collision was detected in Low power mode (Detection of robot motion error)</td>
<td>Collision detection (Detection of robot motion error) was functioned. The following errors have detected: - Collision or contact of the robot arm occurs. - Torque saturation due to holding a hand or long object that exceeds the weight described in the specifications. - Error motion due to hardware error or software malfunction. Countermeasures: Check the there is no collision or contact of the robot arm and change the arrangement to avoid interfere. Check the hand weight and correct it if necessary. Joint #4 and 5 of 6-axis robot: confirmed that torque saturation is occurred. If torque saturation is occurred: change to hold in high power mode. If other error occurs at the same time, take a countermeasure for that first. Reference: EPSON RC+ 7.0 User’s Guide “6.18.10 Collision Detection Function (Error detection function of robot motion)”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5072</td>
<td>Servo alarm B.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5080</td>
<td>Motor is overloaded. Decrease the duty and the Accel.</td>
<td>Lower the motion duty. Check the Weight/Inertia setting. Check the robot. (Backlash, large load, loose belt tension, brake)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5098</td>
<td>High temperature of the encoder. Decrease the duty. Check the reduction gear unit of the robot.</td>
<td>Wait until the temperature of the encoder decreases. Lower the motion duty. Check the Weight/Inertia setting. Check the robot. (Backlash, large load, loose belt tension, brake)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5099</td>
<td>High temperature of the motor driver. Clean the controller fan filter. Check the ambient temperature. Decrease the duty.</td>
<td>Clean the cooling fan filter. Lower the motion duty. Check the Weight/Inertia setting. Lower the ambient temperature.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5112</td>
<td>Servo alarm C.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5501</td>
<td>Failed to initialize the force control. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5510</td>
<td>Force control calculation error. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>5511</td>
<td>Coordinate transformation error in force control.</td>
<td>Check whether the coordinate out of the motion range is specified.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check whether the robot moves outside of the motion range.</td>
<td>Check whether the robot moves outside of the motion range during the execution of force control.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5520</td>
<td>Impedance parameter error.</td>
<td>Check the combination of Spring, Damper, and Mass.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check the combination of Mass, Damper, and Spring</td>
<td>Check whether the Mass property is too small for the Damper property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5521</td>
<td>The coordinate system mode other than the custom mode is specified for</td>
<td>Check association with the robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>the Force Sensor which is not associated with the robot.</td>
<td>Check if the direction other than the Custom coordinate is specified for the Orientation property of the force coordinate system object for the sensor which is not associated with the robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check configuration of the Force Sensor or the coordinate system mode.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5522</td>
<td>Undefined data is selected.</td>
<td>Check if the specified parameter is defined.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check if the selected data is defined.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5523</td>
<td>The parameter which cannot be continued when the CF continues force</td>
<td>Check the force control object and the force coordinate object which are used by the motion commands before and after continuing the force control by the CF parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>control is specified.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check the Force Control Object and the Force Coordinate System Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5530</td>
<td>The specified time has passed after resetting the Force Sensor.</td>
<td>Execute the Reset property for the Force Sensor object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Execute the Reset property of the Force Sensor Object.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5531</td>
<td>Approached the singularity point while executing force control.</td>
<td>Check whether the coordinates near the singularity is specified.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Avoid the singularity point when using force control.</td>
<td>Check whether the robot moves to the vicinity of the singularity during the execution of force control. Or, review the installation position of the robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5532</td>
<td>Buffer for Force Sensor averaging is saturated.</td>
<td>Shorten the time interval between AvgForceClear and AvgForce to be shorter than a minute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Shorten the time interval from AvgForceClear to AvgForce.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5533</td>
<td>The continuing time for CF to execute force control has passed.</td>
<td>Check whether the interval of the motion commands is one minute or less.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>To continue force control, use FCKeep.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5535</td>
<td>SCARA robot cannot execute force control if the Orientation property</td>
<td>Set “0” to the Orientation property or V and W of the current command position for Base, Tool, Local, and FCS objects.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>of Base, Tool, Local, and FCS objects, or V and W of the current</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>command position are other than 0.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check the parameters.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5536</td>
<td>Force control is not supported for this robot model.</td>
<td>Check if the specified robot is correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check the robot model and the controller firmware version.</td>
<td>Check if the Controller firmware supports the robot model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5540</td>
<td>Force Sensor transmission error. Check connection of the Force Sensor I/F unit (board) and Force Sensor.</td>
<td>Execute the Reboot property for the Force Sensor object. Check connection of the Force Sensor and Force Sensor I/F unit (board). Reboot the Force Sensor and Force Sensor I/F unit (board). Please inquire with us if a similar error occurs even after the above countermeasures are taken.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5541</td>
<td>Force Sensor reception error. Check connection of the Force Sensor I/F unit (board) and Force Sensor.</td>
<td>Execute the Reboot property for the Force Sensor object. Check connection of the Force Sensor and Force Sensor I/F unit (board). Reboot the Force Sensor and Force Sensor I/F unit (board). Please inquire with us if a similar error occurs even after the above countermeasures are taken.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5542</td>
<td>Force Sensor in use. Check if other commands are using the Force Sensor.</td>
<td>Check whether the Reset property or Reboot property of the Force Sensor object are executed in another task.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5544</td>
<td>Element error of Force Sensor. Check whether force exceeding the rated value is applied to the Force Sensor. Execute the Reset property of the Force Sensor Object.</td>
<td>This error may occur if a long time passed while the Force Sensor is not reset. Execute the Reset property for the Force Sensor object. Check whether force exceeding the rated value is applied to the Force Sensor. Reboot the Force Sensor and Force Sensor I/F unit (board). Please inquire with us if a similar error occurs even after the above countermeasures are taken.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5545</td>
<td>Circuit error 1 of Force Sensor. Execute the Reset property of the Force Sensor Object.</td>
<td>Execute the Reset property for the Force Sensor object. Reboot the Force Sensor and Force Sensor I/F unit (board). Please inquire with us if a similar error occurs even after the above countermeasures are taken.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>5546</td>
<td>Circuit error 2 of Force Sensor. Execute the Reset property of the</td>
<td>Execute the Reset property for the Force Sensor object. Reboot the</td>
<td>If a similar error occurs even after the above countermeasures are</td>
<td>Check the configuration of the force sensor. Reboot the Force Sensor</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5570</td>
<td>Force monitor buffer overflow. Reboot the force monitor.</td>
<td>Close and re-open the force monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5571</td>
<td>Force log buffer overflow. Set the data measurement interval longer.</td>
<td>Set the data measuring interval longer. The computer receiving the data may be under heavy load state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5573</td>
<td>The log data of the force guide sequence overflowed.</td>
<td>Reboot the controller. The computer receiving the data may be under heavy load state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5574</td>
<td>RecordStart buffer overflowed. Set the data measurement interval longer.</td>
<td>Set the data measuring interval longer. The computer receiving the data may be under heavy load state.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5800</td>
<td>Failed to initialize the force control. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5801</td>
<td>Force control failed to allocate memory. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5802</td>
<td>Force control calculation error. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5810</td>
<td>Force control parameter error. Check the parameter range.</td>
<td>Check the range of the specified parameter.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5811</td>
<td>Force Control Object parameter is out of the range. Note 1: Property</td>
<td>Check the property of force control object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Fx 2: Fy 3: Fz 4: Tx 5: Ty 6: Tz or 1: J 2: S 3: R</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5812</td>
<td>LimitSpeed or LimitAccel of the Force Control Object is lower than speed or accel setting of the robot. Check the parameter.</td>
<td>Check the values of Speed, SpeedS, SpeedR, Accel, AccelS, AccelR, LimitSpeed and LimitAccel.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5813</td>
<td>Enabled properties of the Force Control Object are all false. Set true to 1 or more axis.</td>
<td>Enable the “Enabled” property for at least one axis.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>5814</td>
<td>SCARA robot cannot execute force control if the Orientation property of Base, Tool, Local, and FCS objects, or V and W of the current command position are other than 0, or if Tx_Enabled and Ty_Enabled are not False. Check the parameters.</td>
<td>Disable the enabled state of Tx and Ty. Set “0” to the Orientation property or V and W of the current command position for Base, Tool, Local, and FCS objects.</td>
<td>1: Number 2: Force Sensor 3: Coordinate System 4: Trigger Mode 5: Operator 6: Enabled 7: FMag_Axes 8: TMag_Axes 9: Polarity 10: Upper Level 11: Lower Level 12: Upper Level smaller than Lower Level 13: LPF Enabled 14: LPF Time Constant</td>
<td></td>
</tr>
<tr>
<td>5815</td>
<td>Force Trigger Object parameter is out of the range. Note 1: Property Note 2: Axis Check the parameter.</td>
<td>Check the property of force trigger object.</td>
<td>1: Fx 2: Fy 3: Fz 4: Tx 5: Ty 6: Tz 7: Fmag 8: Tmag</td>
<td></td>
</tr>
<tr>
<td>5816</td>
<td>Force Coordinate System Object parameter is out of the range. Note 1: Property Note 2: Axis Check the parameter.</td>
<td>Check the property of force coordinate system object.</td>
<td>1: X 2: Y 3: Z or 1: U 2: V 3: W</td>
<td></td>
</tr>
<tr>
<td>5817</td>
<td>Force Monitor Object parameter is out of the range. Note 1: Property Note 2: Axis Check the parameter.</td>
<td>Check the property of force monitor object.</td>
<td>1: Number 2: Force Sensor 3: Coordinate System 4: FMag_Axes 5: TMag_Axes 6: LPF Enabled 7: LPF Time Constant</td>
<td></td>
</tr>
<tr>
<td>5818</td>
<td>Force motion restriction object parameter is out of the range.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>5819</td>
<td>Specified duration of FCKeep is out of the allowable range. Check the</td>
<td>Check whether the specified duration is 600 seconds or less.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>duration.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5830</td>
<td>Force control cannot resume from the pause. Abort the motion command.</td>
<td>Abort the motion.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5831</td>
<td>Cannot execute this command during force control. Exit force control by</td>
<td>Abort force control. Execute FCEnd command.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FCEnd.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5832</td>
<td>Cannot execute the motion command which has no Force Control Object</td>
<td>Check whether the motion command right after continuing the force</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>during force control. Exit force control by FCEnd.</td>
<td>control by CT does not contain force control.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5833</td>
<td>Cannot use gravity compensation. Can use only Mass Property object</td>
<td>This is the combination of the force sensor and the robot that</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>number '0'.</td>
<td>cannot use gravity compensation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set Mass Property object number '0'.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5834</td>
<td>Unsupported function is executed on the connected force sensor. Review</td>
<td>Check the force sensor settings. Review the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>the program.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5840</td>
<td>Force Sensor in use. Check whether other commands are sing the Force</td>
<td>Check whether the Reset property or Reboot property of the Force Sensor</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sensor.</td>
<td>object are executed in another task.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5841</td>
<td>Failed to reset the force sensor. Reset the force sensor again.</td>
<td>When the parameter is omitted or FG_RESET_FINE is specified, specify</td>
<td>1: Timeout since Fine condition is not satisfied. 2: Timeout since the vibration did not stop.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Note1: Detailed error information</td>
<td>FG_RESET_WAIT_VIBRATION for parameter. When FG_RESET_WAIT_VIBRATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>is specified, adjust the reset timing by Wait statement or remove</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the source of external vibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5901</td>
<td>Force control failed to allocate memory. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5902</td>
<td>Force control failed to release memory. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5903</td>
<td>The specified robot cannot be found.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5904</td>
<td>Force control failed to allocate memory. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5906</td>
<td>Specified force data number cannot be found. Specify a valid force data</td>
<td>Reboot the controller. Initialize the controller firmware. Replace</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>data number.</td>
<td>the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5907</td>
<td>Specified force data number was not defined. Specify a teach force data</td>
<td>Specify a defined force data number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>number.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5908</td>
<td>Specified force coordinate system data number was not defined. Specify</td>
<td>Specify a defined force coordinate system data number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>a teach force coordinate system data number.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>5909</td>
<td>Specified force data can't be updated.</td>
<td>Specified force data cannot be updated nor deleted by FSet, FDel, MPSet, or MPDel statement.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5910</td>
<td>Specified force data value is out of allowable range.</td>
<td>Specify the value within the range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5911</td>
<td>The Upper level value is smaller than the lower level value. Change the upper and lower level values.</td>
<td>Change the upper and lower level values.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5912</td>
<td>The number of specified command parameters is not correct. Specify a valid number of parameters.</td>
<td>Specify a valid number of parameters.</td>
<td>Number of parameters</td>
<td></td>
</tr>
<tr>
<td>5913</td>
<td>The number of specified function parameters is not correct. Specify a valid number of parameters.</td>
<td>Specify a valid number of parameters.</td>
<td>Number of parameters</td>
<td></td>
</tr>
<tr>
<td>5914</td>
<td>The type of a specified command parameter is not correct. Specify valid parameters.</td>
<td>Specify valid parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5915</td>
<td>The type of a specified function parameter is not correct. Specify valid parameters.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5918</td>
<td>Specified force data label cannot be found. Specify a valid force data label.</td>
<td>Specify a defined valid force data label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5921</td>
<td>Duplicate force data label. Specified label name is already used. Change the label name.</td>
<td>Change the label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5924</td>
<td>Force control of the specified robot failed to allocate memory. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5927</td>
<td>Cannot read the force data from the force file. Re-create the force file.</td>
<td>The force data is invalid and cannot be read. Re-create the force file.</td>
<td>0:FC 1:FCS 2:FT 3:FM 4:MASS</td>
<td>Force data number</td>
</tr>
<tr>
<td>5928</td>
<td>Force control failed to allocate memory. Reboot the controller.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5929</td>
<td>Specified force file name is not correct. Specify a valid force file name.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5930</td>
<td>Specified force data label is beyond the maximum length. Specify a valid force data label.</td>
<td>Specify a valid force data label. Refer to &quot;Label Property&quot; for details.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5931</td>
<td>Description for the specified force data is beyond the maximum length. Specify a valid description.</td>
<td>Specify a valid description. Refer to &quot;Description Property&quot; for details.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5932</td>
<td>The force file is corrupted. Re-create the force file.</td>
<td>Cannot load the force file because it is corrupted or was edited by tools other than Force Guide 7.0. Re-create the force file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>5933</td>
<td>Specified force file cannot be found.</td>
<td>Specify a valid force file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5934</td>
<td>Cannot save the force file.</td>
<td>Make an enough space to write the force file.</td>
<td></td>
<td>Robot number</td>
</tr>
<tr>
<td>5940</td>
<td>The force data label is not correct.</td>
<td>Specify a valid force data label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a valid force data label.</td>
<td>Refer to &quot;Label Property&quot; for details.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5941</td>
<td>The force data label is not correct.</td>
<td>Specify a valid force data label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Specify a valid force data label.</td>
<td>Refer to &quot;Label Property&quot; for details.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5943</td>
<td>Invalid force file version.</td>
<td>Cannot load the force file because it is a newer version.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5944</td>
<td>Failed to read the force file.</td>
<td>Cannot load the force file because the format is not supported.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Re-create the force file.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>---------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>6001</td>
<td>Calibration number is out of range.</td>
<td>Correct the calibration number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6002</td>
<td>Calibration is not defined.</td>
<td>Perform calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6003</td>
<td>Camera orientation is out of range.</td>
<td>Correct the CameraOrientation value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6004</td>
<td>TwoRefPoints flag is out of range.</td>
<td>Correct the TwoRefPoint value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6005</td>
<td>Cannot calculate the point position because there is invalid data.</td>
<td>Re-teach the points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6006</td>
<td>Calibration failed. Cannot calculate because there is invalid data.</td>
<td>Perform point teaching and calibration again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6007</td>
<td>Coordinate transformation failed. Cannot calculate because there is invalid data.</td>
<td>Reteach the points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6009</td>
<td>Calibration file name is invalid.</td>
<td>Correct the calibration file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6010</td>
<td>Calibration file is not found.</td>
<td>Correct the calibration file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6012</td>
<td>Failed to read the calibration file.</td>
<td>Correct the calibration file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6013</td>
<td>Failed to write the calibration file.</td>
<td>Check access permission for the project folder.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6014</td>
<td>9 pixel coordinate points should be specified.</td>
<td>Make sure that at least 9 results are obtained in the vision sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6015</td>
<td>18 pixel coordinate points should be specified.</td>
<td>Make sure that at least 18 results are obtained in the vision sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6016</td>
<td>9 robot coordinate points should be specified.</td>
<td>Reteach the points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6017</td>
<td>18 robot coordinate points should be specified.</td>
<td>Reteach the points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6018</td>
<td>9 robot coordinate points and 1 reference point should be specified.</td>
<td>Perform point teaching and calibration again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6019</td>
<td>9 robot coordinate points and 2 reference points should be specified.</td>
<td>Perform point teaching and calibration again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6502</td>
<td>Vision process Communication error (-3)</td>
<td>Check the connection with the camera (cable, setting).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6503</td>
<td>Vision process Memory error (-11)</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6506</td>
<td>Vision process Error at modeling (-14)</td>
<td>Change the target and teach again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6507</td>
<td>Vision process Recovery error(-15)</td>
<td>Specify the file of appropriate format.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6508</td>
<td>Vision process Invalid number of iterations (-16)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6509</td>
<td>Vision process Invalid mode (-17)</td>
<td>Set a valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6510</td>
<td>Vision process Invalid threshold value (-18)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6511</td>
<td>Vision process Invalid polarity (-19)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6512</td>
<td>Vision process File open failed (-20)</td>
<td>Specify a correct file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6513</td>
<td>Vision process Initialization error (-21)</td>
<td>Reinstall the RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6514</td>
<td>Vision process Status error (-22)</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6517</td>
<td>Vision process Invalid image format (-25)</td>
<td>Specify the image file of readable format.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6520</td>
<td>Vision process Invalid property value (-100)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6521</td>
<td>Vision process Exposure termination process failed (-201)</td>
<td>Disable Windows Firewall.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6533</td>
<td>Vision process Invalid Blob property ThresholdLow value (-11004)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6534</td>
<td>Vision process Invalid Blob property ThresholdHigh value (-11005)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>6535</td>
<td>Vision process Invalid Blob property Polarity value (-11006)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6536</td>
<td>Vision process Invalid Blob property NumberToFind value (-11007)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6537</td>
<td>Vision process Invalid Blob property MinArea value (-11008)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6538</td>
<td>Vision process Invalid Blob property MaxArea value (-11009)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6539</td>
<td>Vision process Invalid Blob property RejectOnEdge value (-11010)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6540</td>
<td>Vision process Invalid Blob property SizeToFind value (-11011)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6553</td>
<td>Vision process Invalid Geom property Accept value (-11504)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6554</td>
<td>Vision process Invalid Geom property NumberToFind value (-11505)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6555</td>
<td>Vision process Invalid Geom property AngleEnable value (-11506)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6556</td>
<td>Vision process Invalid Geom property AngleRange value (-11507)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6557</td>
<td>Vision process Invalid Geom property AngleStart value (-11508)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6558</td>
<td>Vision process Invalid Geom property ScaleEnable value (-11509)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6559</td>
<td>Vision process Invalid Geom property ScaleFactorMax value (-11510)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6560</td>
<td>Vision process Invalid Geom property ScaleFactorMin value (-11511)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6561</td>
<td>Vision process Invalid Geom property ScaleTarget value (-11512)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6562</td>
<td>Vision process Invalid Geom property SeparationMinX value (-11513)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6563</td>
<td>Vision process Invalid Geom property SeparationMinY value (-11514)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6564</td>
<td>Vision process Invalid Geom property SeparationAngle value (-11515)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6565</td>
<td>Vision process Invalid Geom property SeparationScale value (-11516)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6566</td>
<td>Vision process Invalid Geom property Confusion value (-11517)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6567</td>
<td>Vision process Invalid Geom property ModelOrgAutoCenter value (-11518)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6570</td>
<td>Vision process Invalid Geom property DetailLevel value (-11521)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6571</td>
<td>Vision process Invalid Geom property Smoothness value (-11522)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6572</td>
<td>Vision process Invalid Geom property RejectOnEdge value (-11523)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6573</td>
<td>Vision process Invalid Geom property SharedEdges value (-11524)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6574</td>
<td>Vision process Invalid Geom property Timeout value (-11525)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6575</td>
<td>Vision process Invalid Geom property RejectByArea value (-11526)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6576</td>
<td>Vision process Invalid Geom property SearchReversed value (-11527)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>6577</td>
<td>Vision process Invalid Geom property ScaleTargetPriority value (-11528)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6578</td>
<td>Vision process Invalid Geom property SearchReducedImage value (-11529)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6586</td>
<td>Vision process Invalid Geom Model property DetailLevel value (-11602)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6587</td>
<td>Vision process Invalid Geom Model property Smoothness value (-11603)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6603</td>
<td>Vision process Invalid Corr property Accept value (-12004)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6604</td>
<td>Vision process Invalid Corr property NumberToFind value (-12005)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6605</td>
<td>Vision process Invalid Corr property AngleEnable value (-12006)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6606</td>
<td>Vision process Invalid Corr property AngleRange value (-12007)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6607</td>
<td>Vision process Invalid Corr property AngleStart value (-12008)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6608</td>
<td>Vision process Invalid Corr property AngleAccuracy value (-12009)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6609</td>
<td>Vision process Invalid Corr property Confusion value (-12010)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6610</td>
<td>Vision process Invalid Corr property ModelOrgAutoCenter value (-12011)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6613</td>
<td>Vision process Invalid Corr property RejectOnEdge value (-12014)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6614</td>
<td>Vision process Invalid Corr property Timeout value (-12015)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6615</td>
<td>Vision process Invalid Corr property RejectByArea value (-12016)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6630</td>
<td>Vision process Invalid Edge property structure size (-12501)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6631</td>
<td>Vision process Invalid Edge result header structure size (-12502)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6632</td>
<td>Vision process Invalid Edge result item structure size (-12503)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6633</td>
<td>Vision process Invalid Edge property EdgeType value (-12504)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6634</td>
<td>Vision process Invalid Edge property NumberToFind value (-12505)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6635</td>
<td>Vision process Invalid Edge property Polarity value (-12506)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6636</td>
<td>Vision process Invalid Edge property SearchWidth value (-12507)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6637</td>
<td>Vision process Invalid Edge property Accept value (-12508)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6638</td>
<td>Vision process Invalid Edge property ScoreWeightContrast value (-12509)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6639</td>
<td>Vision process Invalid Edge property ContrastTarget value (-12510)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6640</td>
<td>Vision process Invalid Edge property ContrastVariation value (-12511)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6641</td>
<td>Vision process Invalid Edge property StrengthTarget value (-12512)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6642</td>
<td>Vision process Invalid Edge property StrengthVariation value (12513)</td>
<td>Set a value in the valid range.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>6653</td>
<td>Vision process Code Reader</td>
<td>Checksum error (-1010)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Change to the code with a proper checksum. Or, change the setting not to use the checksum.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6654</td>
<td>Vision process Code Reader</td>
<td>Invalid quiet zone (-1011)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ensure a quiet zone (blank margin) around the code. Set the quiet zone narrower.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6655</td>
<td>Vision process Code Reader</td>
<td>Message is too long (-1012)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Change the code.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6686</td>
<td>Vision process OCR</td>
<td>Recognition dictionary is full (-2132)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Delete the registered characters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>7003</td>
<td>The specified robot cannot be found.</td>
<td>Reboot the controller. Initialize the control firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7004</td>
<td>Duplicate allocation of the point data area.</td>
<td>Reboot the controller. Initialize the control firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7006</td>
<td>Specified point number cannot be found. Specify a valid point number.</td>
<td>Check the specified point number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7007</td>
<td>Specified point number was not defined. Specify a teach point number.</td>
<td>Check whether point data is registered in the specified point. Perform the teaching.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7010</td>
<td>Cannot allocate the memory area for the pallet definition.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7011</td>
<td>Cannot free the memory area for the pallet definition.</td>
<td>Reboot the controller. Initialize the controller firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7012</td>
<td>Specified pallet number cannot be found. Specify a valid pallet number.</td>
<td>Check the pallet number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7013</td>
<td>Specified pallet is not defined. Specify a defined pallet or define the pallet.</td>
<td>Check whether the specified pallet is defined by the Pallet statement. Declare the pallet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7014</td>
<td>Specified division number is beyond the pallet division number definition. Specify a valid division.</td>
<td>Check the specified division number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7015</td>
<td>Specified coordinate axis number does not exist.</td>
<td>Check the specified coordinate axis number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7016</td>
<td>Specified arm orientation number does not exist.</td>
<td>Check the specified arm orientation number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7017</td>
<td>Cannot allocate the required memory.</td>
<td>Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7018</td>
<td>Specified point label cannot be found. Specify a valid point label.</td>
<td>Check the specified point label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7019</td>
<td>Parameter setup in the initialization file is invalid.</td>
<td>Reboot the controller. Initialize the controller firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7021</td>
<td>Duplicate point label. Specified label name is already used. Change the label name.</td>
<td>Change the point label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7022</td>
<td>Specified local coordinate system is not defined. Specify a valid local coordinate system number.</td>
<td>Check the specified local number. Define the Local coordinate system.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7024</td>
<td>Point data memory area for the specified robot is not allocated.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7026</td>
<td>Cannot open the point file. Specify a valid point file name.</td>
<td>Check the point file name. Check whether the point file specified for the project exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7027</td>
<td>Cannot read the point data from the point file.</td>
<td>Create the point file again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7028</td>
<td>Point area is allocated beyond the available point number.</td>
<td>There are too many points. Review the number of points.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7029</td>
<td>Specified point file name is not correct. Specify a valid point file name.</td>
<td>Check the file extension.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>7030</td>
<td>Specified point label is beyond the maximum length. Specify a valid point label.</td>
<td>Change the point label.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7031</td>
<td>Description for the specified point is beyond the maximum length. Specify a valid description.</td>
<td>Change the comment.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7032</td>
<td>Point file is corrupted. Check sum error.</td>
<td>Create the point file again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7033</td>
<td>Specified point file cannot be found. Specify a valid point file name.</td>
<td>Check the name of the specified point file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7034</td>
<td>Cannot save the point file.</td>
<td>Failed to save the point file (create a temporary file). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7035</td>
<td>Cannot save the point file.</td>
<td>Failed to save the point file (file open). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7036</td>
<td>Cannot save the point file.</td>
<td>Failed to save the point file (renew the file header). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7037</td>
<td>Cannot save the point file.</td>
<td>Failed to save the point file (create the file name). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7038</td>
<td>Cannot save the point file.</td>
<td>Failed to save the point file (copy the file). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7039</td>
<td>Cannot save the point file.</td>
<td>Failed to save the point file (change the file name). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7040</td>
<td>The point label is not correct. Specify a valid point label.</td>
<td>The initial character of the point label name is improper. Correct the label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7041</td>
<td>The point label is not correct. Specify a valid point label.</td>
<td>Inadequate character is used. Correct the label name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7042</td>
<td>The pallet cannot be defined.</td>
<td>Undefined flag for pallet data is mixed. Check the point data. Correct the point data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7043</td>
<td>Invalid point file version.</td>
<td>The point file version is different. Re-create the point file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7044</td>
<td>The point file format version is unsupported.</td>
<td>The point file is not supported. Re-create the point file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7045</td>
<td>The specified work queue number is invalid.</td>
<td>Check the specified work queue number.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7046</td>
<td>The specified work queue is full.</td>
<td>The work queue is full. Delete the point data from the work queue and register.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7047</td>
<td>The specified work queue data does not exist.</td>
<td>Check the specified index.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7048</td>
<td>The work queue is not correctly initialized.</td>
<td>Failed to initialize the work queue (allocate memory). Reboot the controller. Initialize the controller firmware. Replace the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7049</td>
<td>The parameter of the work queue instruction is invalid.</td>
<td>Check the parameters of the commands related to the work queue.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7050</td>
<td>Cannot execute while work queue data is registered.</td>
<td>Cannot set the work queue since the point data is registered to the work queue. Empty the work queue before setting.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 7101 Fieldbus slave. An error occurred during I/O data transform.
- The fieldbus slave board is broken or the controller software is damaged. Restore the controller firmware.
- A communication data error was detected during communication. The communication cable has a problem. Check the communication cable and its related units.
- The fieldbus is broken or the controller software is damaged. Restore the controller firmware.
- The PLC is not running or not connected. Check the PLC, the communication cable, and peripherals. (If Code 1 is 22 when the CC-Link board is used.)

### 7103 Fieldbus slave. Timeout error occurred during I/O data transform.
- The fieldbus slave board is broken or the controller software is damaged. Restore the controller firmware.
- A communication data error was detected during communication. The communication cable has a problem. Check the communication cable and its related units.

### 7150 Fieldbus master. Bus is disconnected.
- Check the connection of the communication cable for the fieldbus.

### 7151 Fieldbus master. Bus power is off.
- Check whether the communication cable for the fieldbus is powered.

### 7152 Fieldbus master. Bus status error.
- Reboot the controller.
- Check the fieldbus master board.
- Replace the fieldbus master board.

### 7200 Invalid argument.
- Check the parameter.

### 7201 A system error occurred.
- |

### 7202 There is not enough memory.
- |
<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7203</td>
<td>Access is denied.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7210</td>
<td>Drive is not ready.</td>
<td>Set the device.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7211</td>
<td>The specified path is invalid.</td>
<td>Make sure the specified path exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7212</td>
<td>The specified path already exists.</td>
<td>If the specified directory or file already exists, you cannot execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7213</td>
<td>The file specified by path does not exist.</td>
<td>Make sure the specified file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7214</td>
<td>File size is too large.</td>
<td>Specify the file that is less than 2G bytes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7215</td>
<td>The specified file is open.</td>
<td>The specified file number is already existing. Use another file number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7216</td>
<td>The open mode is illegal.</td>
<td>Make sure you opened in reading or writing mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7217</td>
<td>There is no read data.</td>
<td>Make sure there are data to read.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7230</td>
<td>The specified connection is open.</td>
<td>The specified file number is already existing. Use another file number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7231</td>
<td>A connection-level error occurred while opening the connection.</td>
<td>Check the access right of database.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7232</td>
<td>The connection is closed.</td>
<td>Use OpenDB and open the database.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7233</td>
<td>An unsupported data type was used.</td>
<td>Convert the data into string or numeric value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7234</td>
<td>Data size is too large.</td>
<td>Too large data in a line. Specify the query so that necessary field are only retrieved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7235</td>
<td>The specified file type is not supported.</td>
<td>Check the type of Excel file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7236</td>
<td>There is no selected data.</td>
<td>Make sure the data you retrieved exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7250</td>
<td>No bytes were available to read.</td>
<td>There are no retrieved data. Check the send program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7251</td>
<td>The port is in an invalid state.</td>
<td>Check the device setting for the specified port.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7252</td>
<td>The specified port is open.</td>
<td>Check the port number to open.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7253</td>
<td>The port is closed</td>
<td>Check the port number to close.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7254</td>
<td>The specified port is not open.</td>
<td>Check the port number to open.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7255</td>
<td>Timeout reading from the port.</td>
<td>Check the port timeout period and update to the appropriate setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7256</td>
<td>Timeout writing to the port.</td>
<td>Check the port timeout period and update to the appropriate setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7260</td>
<td>The checksum in project file is invalid.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7261</td>
<td>Invalid function.</td>
<td>Check the function definition to call.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7262</td>
<td>Invalid parameters.</td>
<td>Check the function definition to call.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7263</td>
<td>Cannot execute while creating DLL.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7264</td>
<td>Failed to create DLL.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7265</td>
<td>DLL file cannot be found.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7300</td>
<td>Vision Communication. Server mode not supported.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7302</td>
<td>Vision Communication. Failed to read from the camera.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7303</td>
<td>Vision Communication. Read data overflow.</td>
<td>Data exceeding the receive buffer was received.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7304</td>
<td>Vision Communication. Failed to open the Ethernet port.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7305</td>
<td>Vision Communication. Invalid IP address of camera.</td>
<td>Rebuild the project. Check the camera configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7306</td>
<td>Vision Communication. No specification of Server or Client.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7307</td>
<td>Vision Communication. Failed to send to the camera.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7308</td>
<td>Vision Communication. Camera version is old.</td>
<td>The version of the connected camera is old. Update the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7321</td>
<td>Vision Communication. Camera setting has not been set.</td>
<td>Rebuild the project. Check the camera configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7322</td>
<td>Vision Communication. Read timeout.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7323</td>
<td>Vision Communication. Read invalid data.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7324</td>
<td>Vision Communication. Failed to send to the camera.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7325</td>
<td>Vision Communication. Connection is not completed.</td>
<td>Check the connection with the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7326</td>
<td>Vision Communication. Read data is too long.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7327</td>
<td>Vision Communication. Undefined vision sequence.</td>
<td>Check the sequence name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7328</td>
<td>Vision Communication. Camera setting has not been set.</td>
<td>Rebuild the project. Check the camera configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7329</td>
<td>Vision Communication. Vis file was not found.</td>
<td>Rebuild the project. Check the camera configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7330</td>
<td>Vision Communication. Failed to allocate memory.</td>
<td>Reduce the number of sequences, objects, and calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7341</td>
<td>Vision Communication. Out of max camera number.</td>
<td>Review the camera registration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7342</td>
<td>Vision Communication. Invalid camera number.</td>
<td>Review the camera registration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7343</td>
<td>Vision Communication. VSet parameter is too long.</td>
<td>Review the names and string variables of sequences, objects, and calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7344</td>
<td>Vision Communication: Too many parameters for VGet.</td>
<td>The number of specified variables is exceeding 32. Reduce the number of parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7345</td>
<td>Vision Communication. Not enough data for VGet statement variable assign.</td>
<td>Reboot the camera. Check the version of the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7346</td>
<td>Vision Communication. Cannot execute a Vision statement from the command window.</td>
<td>Execute the command from the program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7400</td>
<td>Matrix determinate too small.</td>
<td>If specifying the virtual camera, specify the real camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7402</td>
<td>Invalid value for maximum motion distance.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>-----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>7403</td>
<td>Invalid value for maximum pose difference angle.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7404</td>
<td>Invalid value for LJMMode.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7405</td>
<td>Command aborted by user.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7406</td>
<td>Joint 1 angle change exceeded the maximum allowed during calibration.</td>
<td>Adjust the start angle of Joint 1.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7407</td>
<td>Joint 2 angle change exceeded the maximum allowed during calibration.</td>
<td>Adjust the start angle of Joint 2.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7408</td>
<td>Joint 4 angle change exceeded the maximum allowed during calibration.</td>
<td>Adjust the start angle of Joint 4.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7409</td>
<td>Joint 6 angle change exceeded the maximum allowed during calibration.</td>
<td>Adjust the start angle of Joint 6.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7410</td>
<td>Network camera. Timeout during image file transfer from PC.</td>
<td>Check the connection of PC and camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7411</td>
<td>No upward camera sequence was specified for mobile calibration with upward reference.</td>
<td>Specify the existing sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7412</td>
<td>The specified upward camera sequence has no calibration.</td>
<td>Set the calibration for upward camera sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7413</td>
<td>The specified upward camera sequence calibration is not complete.</td>
<td>Complete the upward camera sequence calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7414</td>
<td>The target sequence cannot be used when RuntimeAcquire is Strobed.</td>
<td>Set the RuntimeAcquire of target sequence to Stationary.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7415</td>
<td>Invalid calibration reference type.</td>
<td>Selectable ReferenceType is different depending on CameraOrientation. Select again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7416</td>
<td>Invalid calibration data. Teach the calibration points again.</td>
<td>Need to teach the calibration points again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7417</td>
<td>Invalid calibration setup.</td>
<td>Try to perform point teach of calibration again. Or check the target sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7418</td>
<td>Invalid calibration target sequence.</td>
<td>Target sequence may not be selected or camera number of the target sequence differs from that of calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7419</td>
<td>The target sequence camera is not the same as the calibration camera.</td>
<td>Set the sequence of the same camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7420</td>
<td>The target sequence has no objects.</td>
<td>Add the detection object to target sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7421</td>
<td>Invalid last step for the target sequence.</td>
<td>Check the steps.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7422</td>
<td>Exception occurred when search for the calibration target.</td>
<td>Check the target sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7423</td>
<td>Invalid number of results for calibration target sequence.</td>
<td>Create a sequence to detect results of required number of targets.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7424</td>
<td>Cannot load the calibration points.</td>
<td>Perform calibration point teach again.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7425</td>
<td>Invalid camera orientation.</td>
<td>Check the CameraOrientation of calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7426</td>
<td>Distortion correction calibration is incomplete.</td>
<td>Perform distortion correction if it is set.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>7427</td>
<td>Invalid vision object was specified.</td>
<td>Invalid vision object is specified in Vision Guide commands such as VSet and VGet. Specify the valid vision object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7428</td>
<td>V and W coordinates must be zero for the type of robot used.</td>
<td>Set V and W of the Base to 0.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7429</td>
<td>Invalid robot speed specified for the current operation.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7430</td>
<td>Invalid robot acceleration specified for the current operation.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7431</td>
<td>Invalid ShowWarning parameter value.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7432</td>
<td>Cannot create the object using the camera specified in the sequence.</td>
<td>Check the Vision object type. Update the camera firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7433</td>
<td>Invalid model data.</td>
<td>Re-teach the model or use a different model if the error occurred during</td>
<td>The model being loaded may not be compatible with the current version of CV or RC+.</td>
<td></td>
</tr>
<tr>
<td>7500</td>
<td>Network camera. Out of memory.</td>
<td>Initialize the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7501</td>
<td>Network camera. Project does not exist.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7502</td>
<td>Network camera. Project has not been set.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7503</td>
<td>Network camera. Vision property or result not supported.</td>
<td>Update the camera firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7504</td>
<td>Network camera. Cannot open project file.</td>
<td>Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7505</td>
<td>Undefined vision sequence.</td>
<td>Check the sequence name.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>7506</td>
<td>Undefined vision object.</td>
<td>Check the object name.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>7507</td>
<td>Network camera. Critical error.</td>
<td>Initialize the camera.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>7508</td>
<td>Network camera. Invalid command.</td>
<td>Update the camera firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7509</td>
<td>Invalid vision property value.</td>
<td>Check the property value.</td>
<td>Update the camera firmware.</td>
<td></td>
</tr>
<tr>
<td>7510</td>
<td>Invalid vision property.</td>
<td>Check the property name.</td>
<td>Update the camera firmware.</td>
<td></td>
</tr>
<tr>
<td>7511</td>
<td>Vision model not trained.</td>
<td>Teach the model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7512</td>
<td>Undefined vision calibration.</td>
<td>Check the calibration name.</td>
<td>Rebuild the project.</td>
<td></td>
</tr>
<tr>
<td>7513</td>
<td>Vision model object not Self.</td>
<td>Check the property value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7514</td>
<td>Invalid vision result.</td>
<td>Check the result name.</td>
<td>Update the camera firmware.</td>
<td></td>
</tr>
<tr>
<td>7515</td>
<td>Vision object not found.</td>
<td>Check the Found result before obtaining the result.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7516</td>
<td>No vision calibration.</td>
<td>Check the calibration name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7517</td>
<td>Incomplete vision calibration.</td>
<td>Perform calibration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>7518</td>
<td>Network camera. Cannot connect with camera.</td>
<td>Check the camera connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7819</td>
<td>Network camera. Communication error.</td>
<td>Check the camera connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7520</td>
<td>Window out of bounds.</td>
<td>Set the window within the bounds.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7521</td>
<td>OCR font is invalid.</td>
<td>Register the OCR font.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7522</td>
<td>The specified vision calibration already exists.</td>
<td>Change the calibration name. Delete the existing calibration in advance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7523</td>
<td>The specified vision sequence already exists.</td>
<td>Change the sequence name. Delete the existing sequence in advance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7524</td>
<td>The specified vision object already exists.</td>
<td>Change the object name. Delete the existing sequence in advance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7525</td>
<td>Cannot load vision project.</td>
<td>The project folder may be corrupt. Restore backup data.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7526</td>
<td>Cannot save vision project.</td>
<td>The project folder may be write-protect. Check the access permission for the project folder.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7527</td>
<td>Vision processor. Critical error.</td>
<td>Initialize the camera. Rebuild the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7528</td>
<td>Image file not found.</td>
<td>Check the image file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7529</td>
<td>Camera does not exist.</td>
<td>Check the camera connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7530</td>
<td>Acquisition failed.</td>
<td>Check the camera connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7531</td>
<td>No objects to train.</td>
<td>Teach the model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7532</td>
<td>Cannot load image file.</td>
<td>Check the image file.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7533</td>
<td>Camera is not supported by RC+7.0. SC300/SC1200 is not supported by RC+7.0. Use CV1/CV2.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7534</td>
<td>Camera firmware does not support new functions of RC+7.0.</td>
<td>Update the camera firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7535</td>
<td>Invalid data from network camera.</td>
<td>Initialize the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7536</td>
<td>Network camera export status failed.</td>
<td>Initialize the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7537</td>
<td>Invalid ImageSize value. The specified value is not supported by the camera. ImageSize exceeding the camera resolution cannot be specified. Correct the property value.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7538</td>
<td>Invalid ZoomFactor. The specified value requires data outside of the image area. Settable values are from 0.1 to 10.0. Correct the property value.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7539</td>
<td>The camera does not support Code Reader.</td>
<td>Update the camera firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7540</td>
<td>The camera does not support OCR.</td>
<td>Update the camera firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7541</td>
<td>Insufficient data for teaching model. Black or white image cannot be registered as a model.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7542</td>
<td>Model window cannot be outside of image. Correct the position of the model window.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7543</td>
<td>Calibration points have not been taught.</td>
<td>Teach the calibration point.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7544</td>
<td>Calibration must be fixed upward. Sequence with the calibration data of the upward fixed camera is only settable.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7545</td>
<td>Point was not defined.</td>
<td>Teach the calibration point.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7546</td>
<td>RobotPlacePos has not been calibrated. Click CalRobotPlacePos and calibrate RobotPlacePos.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------</td>
<td>-----------------------------------------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>7547</td>
<td>Camera IP address is out of current subnet.</td>
<td>Correct the camera IP address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7548</td>
<td>Camera was not detected.</td>
<td>Check the wiring of the camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7549</td>
<td>Invalid Radius. Radius must be ( \geq ) RadiusInner and ( \leq ) RadiusOuter.</td>
<td>Correct the property value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7550</td>
<td>OCR character does not exist.</td>
<td>Register the OCR character.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7551</td>
<td>OCR option is not active.</td>
<td>Enable the option.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7572</td>
<td>Invalid sequence name.</td>
<td>Specify a name that begins with an alphabet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7573</td>
<td>Invalid calibration name.</td>
<td>Specify a name that begins with an alphabet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7574</td>
<td>Sequence or calibration name already exists.</td>
<td>Specify another calibration name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7575</td>
<td>Invalid camera.</td>
<td>Specify valid camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7576</td>
<td>The vision target could not be found.</td>
<td>Check the vision sequence to detect the target.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7577</td>
<td>Failed to position the vision target within the specified tolerance.</td>
<td>Check the vision sequence to detect the target.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7578</td>
<td>No object with a search window was found in the sequence.</td>
<td>Add an object to detect the target in the vision sequence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7579</td>
<td>Invalid initial rotation angle.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7580</td>
<td>Invalid final rotation angle.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7581</td>
<td>Invalid target tolerance.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7582</td>
<td>Invalid tool definition type.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7583</td>
<td>Invalid rotation angle.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7584</td>
<td>Invalid local definition type.</td>
<td>Specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7585</td>
<td>Calibration plate detection failed.</td>
<td>Adjust the focus and exposure time of the lens to show the target clearly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7586</td>
<td>Focal length detection failed.</td>
<td>Narrow down a lens diaphragm.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7587</td>
<td>Local definition scale detection failed.</td>
<td>Adjust the focus and exposure time of the lens to show the target clearly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7588</td>
<td>Calibration plate pose detection failed.</td>
<td>Adjust the focus and exposure time of the lens to show the target clearly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7589</td>
<td>Invalid object name.</td>
<td>Specify a name that begins with an alphabet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7590</td>
<td>Maximum move distance exceeded the limit set by VDefSetMotionRange.</td>
<td>Adjust the start position. Or set the limit value widely.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7591</td>
<td>Maximum pose difference angle exceeded the limit set by VDefSetMotionRange.</td>
<td>Adjust the start position. Or set the limit value widely.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7592</td>
<td>Maximum joint angle difference exceeded the limit set by VDefSetMotionRange.</td>
<td>Adjust the start position. Or set the limit value widely.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7596</td>
<td>Local definition rough camera alignment failed.</td>
<td>Adjust the start position.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7597</td>
<td>Local definition plane could not be calculated.</td>
<td>Adjust the vision sequence to show the calibration plate clearly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7598</td>
<td>Calibration generates points move distance too small.</td>
<td>Make the search area bigger or the target smaller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7599</td>
<td>Calibration generate points camera to robot relation error.</td>
<td>If specifying the virtual camera, specify the real camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>7600</td>
<td>GUI Builder. Cannot execute a GUI Builder statement from the command window.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7602</td>
<td>GUI Builder. GSet parameter is too long.</td>
<td>Correct the parameter to the proper length.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7603</td>
<td>GUI Builder. Too many parameters for GGet.</td>
<td>Check the number of parameters.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7604</td>
<td>GUI Builder. Not enough data for GGet statement variable assignment.</td>
<td>Specify the variable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7610</td>
<td>GUI Builder. The event task cannot be executed. System in pause state and EventTaskType is Normal.</td>
<td>The system can be operated by changing EventTaskType to &quot;NoPause&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7611</td>
<td>GUI Builder. The event task cannot be executed. Safeguard is open and EventTaskType is Normal.</td>
<td>The system can be operated by changing EventTaskType to &quot;NoEmgAbort&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7612</td>
<td>GUI Builder. The event task cannot be executed. Estop is active and EventTaskType is not NoEmgAbort.</td>
<td>The system can be operated by changing EventTaskType to &quot;NoEmgAbort&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7613</td>
<td>GUI Builder. The event task cannot be executed. System in error state and EventTaskType is not NoEmgAbort.</td>
<td>The system can be operated by changing EventTaskType to &quot;NoEmgAbort&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7650</td>
<td>GUI Builder. Invalid property.</td>
<td>Specify the valid property.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7651</td>
<td>GUI Builder. Invalid form.</td>
<td>Specify the valid form.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7652</td>
<td>GUI Builder. Invalid control.</td>
<td>Specify the valid control.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7653</td>
<td>GUI Builder. The specified form is already open.</td>
<td>Modify the program to avoid double launch.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7654</td>
<td>GUI Builder. Event function does not exist.</td>
<td>Check the function name set for the event.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7655</td>
<td>GUI Builder. Item does not exist.</td>
<td>Specify the valid item.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7656</td>
<td>GUI Builder. Invalid property value.</td>
<td>Check the property value and specify the valid value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7700</td>
<td>Security. Invalid user.</td>
<td>Contact the administrator to register the user.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7701</td>
<td>Security. Invalid password.</td>
<td>Check the password.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7702</td>
<td>Security. Permission denied.</td>
<td>Contact the administrator to set authority.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7703</td>
<td>Security. Option not active.</td>
<td>Register the options.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7710</td>
<td>Source and destination cannot be the same.</td>
<td>Specify another destination.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7711</td>
<td>Point file name is used by another robot.</td>
<td>Check the point file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7712</td>
<td>Invalid axis specified.</td>
<td>Check whether the specified axis is valid. Check if the axis is specified correctly.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7713</td>
<td>Option not enabled</td>
<td>Enable the option.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7714</td>
<td>File not found.</td>
<td>Specify the correct file name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7715</td>
<td>Robot number is out of the available range.</td>
<td>Check the robot number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7716</td>
<td>Robot does not exist.</td>
<td>Check whether the robot is registered.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7717</td>
<td>File Error. Invalid folder.</td>
<td>Check the folder name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7718</td>
<td>Cannot write the file.</td>
<td>Check the storage or the write authority.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7751</td>
<td>Simulator. Failed to save the objects.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7752</td>
<td>Simulator. Failed to load the objects.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7753</td>
<td>Simulator. Failed to mapping of memory.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7754</td>
<td>Simulator. The virtual controller already exists.</td>
<td>Name of the virtual controller may be duplicated. Check the controller name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7755</td>
<td>Simulator. Failed to create the virtual controller connection information.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7756</td>
<td>Simulator. The copy source of the virtual controller does not exist.</td>
<td>Check the virtual controller name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7757</td>
<td>Simulator. The copy destination of the virtual controller already exists.</td>
<td>Name of the virtual controller may be duplicated. Check the controller name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7758</td>
<td>Simulator. Failed to copy the virtual controller connection information.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7759</td>
<td>Simulator. Failed to delete the virtual controller connection information.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7760</td>
<td>Simulator. Failed to delete the controller connection information.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7761</td>
<td>Simulator. Failed to rename the controller connection information.</td>
<td>Check the virtual controller name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7762</td>
<td>Simulator. The rename source of the virtual controller does not exist.</td>
<td>Check the virtual controller name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7763</td>
<td>Simulator. The rename destination of the virtual controller already exists.</td>
<td>Check the virtual controller name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7765</td>
<td>Simulator. Failed to read the Robot definition file.</td>
<td>Check if the definition file exists.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7766</td>
<td>Simulator. Failed to copy the layout objects.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7767</td>
<td>Simulator. Failed to cut the layout objects.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7768</td>
<td>Simulator. Failed to paste the layout objects.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7769</td>
<td>Simulator. Failed to remove the Robot.</td>
<td>Reboot RC+.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>7773</td>
<td>Simulator. Robot or Object was not specified.</td>
<td>Specify Robot or Object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7774</td>
<td>Simulator. Duplicated Robot name or Object name.</td>
<td>Change the Robot name or Object name so that it does not duplicate.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7775</td>
<td>Simulator. Could not find Robot.</td>
<td>Check whether the Robot is set or check the Robot name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7776</td>
<td>Simulator. Could not find Object.</td>
<td>Check whether the Object is set or check the Object name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7777</td>
<td>Simulator. Could not find Hand.</td>
<td>Check whether the Hand is set or check the Hand name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7778</td>
<td>Simulator. The specified object is already registered as a Part object.</td>
<td>Unregister the Part.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7779</td>
<td>Simulator. The specified object is not Part object</td>
<td>Specify the object set for the Part.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7780</td>
<td>Simulator. Could not find the specified Tool.</td>
<td>Specify the set Tool.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7781</td>
<td>Simulator. Child object can not be picked.</td>
<td>Change to parent object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7782</td>
<td>Simulator. Parent objects can not be specified for Part, Mounted Device, or Mobile Camera</td>
<td>Unregister as Part, Mounted Device or Mobile Camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7783</td>
<td>Simulator. Robot can not be specified.</td>
<td>Specify an Object other than Robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7784</td>
<td>Simulator. The same object can not be specified as a parent object.</td>
<td>Specify another object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7785</td>
<td>Simulator. Child object can not be specified as parent object.</td>
<td>Change to parent object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7786</td>
<td>Simulator. The specified object is already registered as a parent object.</td>
<td>Specify another object.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7787</td>
<td>Simulator. Specified value is invalid.</td>
<td>Check the set value.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7788</td>
<td>Simulator. Specified variable type is invalid.</td>
<td>Check the variable type.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7789</td>
<td>Simulator. Object can not be specified.</td>
<td>Specify the Robot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7790</td>
<td>Simulator. Hand can not be specified.</td>
<td>Specify an object other than Hand.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7791</td>
<td>Simulator. Camera can not be specified.</td>
<td>Specify an object other than Camera.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7800</td>
<td>Data cannot be changed, because it is not data of PG axis.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7801</td>
<td>Invalid joint number was specified.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7802</td>
<td>The robot type is invalid.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7803</td>
<td>The parameter is invalid.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7804</td>
<td>The robot number is invalid.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7805</td>
<td>MCD failure. Failed to open the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7806</td>
<td>MCD failure. Failed to read the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7807</td>
<td>MCD failure. Failed to save the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7808</td>
<td>MCD failure. Failed to create the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7809</td>
<td>MCD failure. Failed to write the MCD file.</td>
<td>Restore the controller configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>7810</td>
<td>MPL failure. Failed to open the MPL file.</td>
<td>Reinstall the firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7811</td>
<td>MPL failure. Failed to read the MPL file.</td>
<td>Update the firmware.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 7812 | MPL failure. Failed to write the MPL file. | 1. Reboot the controller.  
2. Reinstall the firmware. |        |        |
| 7815 | IFS failure. Failed to open the IFS file. | Restore the controller configuration. |        |        |
| 7816 | IFS failure. Failed to read the IFS file. | Restore the controller configuration. |        |        |
| 7817 | IFS failure. Failed to write the IFS file. | Restore the controller configuration. |        |        |
| 7822 | MTR failure. Failed to read the MTR file. | 1. Reboot the controller.  
2. Reinstall the MT. |        |        |
| 7824 | MTR failure. Failed to save the MTR file. | Reboot the controller. |        |        |
| 7825 | PRM failure. Failed to create the PRM file. | Restore the controller configuration. |        |        |
| 7827 | PRM failure. Failed to read the PRM file. | Restore the controller configuration. |        |        |
| 7829 | PRM failure. Failed to save the PRM file. | Restore the controller configuration. |        |        |
| 7830 | File failure. Cannot access the file. | 1. Reboot the controller.  
2. Reinstall the firmware. |        |        |
| 7831 | The motor type is invalid. | Check the motor amplifier. |        |        |
| 7840 | MCD failure. Area allocate error. | Reboot the controller. |        |        |
| 7845 | FGI failure. Failed to open the FGI file. | Reboot the controller.  
Reinstall the firmware. | - | - |
| 7847 | MDL failure. Failed to open the MDL file. | Reboot the controller.  
Reinstall the firmware. | - | - |
| 7848 | MDL failure. Failed to read the MDL file. | Reboot the controller.  
Reinstall the latest firmware version. | - | - |
| 7900 | Fieldbus not installed. | - |        |        |
| 7901 | Fieldbus invalid parameter. | - |        |        |
| 7902 | Fieldbus line defect. | Check the connection of the communication cable for the fieldbus.  
Check whether the communication cable for the fieldbus is powered. (if the fieldbus requires power supply)  
Check the connection of the fieldbus slave. |        |        |
| 7903 | Fieldbus device not configured. | Check that the fieldbus master board is installed.  
Reboot the computer where the fieldbus master board is installed.  
Replace the fieldbus master board. |        |        |
| 7904 | Fieldbus invalid board. | Check that the fieldbus master board is installed.  
Reboot the computer where the fieldbus master board is installed.  
Replace the fieldbus master board. |        |        |
<p>| 7905 | Fieldbus connection denied. | - |        |        |</p>
<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>7906</td>
<td>Fieldbus invalid device configuration.</td>
<td>Check that the fieldbus master board is installed. Reboot the computer where the fieldbus master board is installed. Replace the fieldbus master board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7907</td>
<td>Fieldbus general error.</td>
<td>Check that the fieldbus master board is installed. Reboot the computer where the fieldbus master board is installed. Replace the fieldbus master board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7908</td>
<td>Fieldbus configuration error.</td>
<td>Check the fieldbus master setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7909</td>
<td>Fieldbus slaves were not detected.</td>
<td>Register the slave to the fieldbus master by accompanying applicomIO Console application.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7910</td>
<td>Fieldbus configuration file not found.</td>
<td>Import the Fieldbus configuration file as described in the Fieldbus I/O manual.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7911</td>
<td>Fieldbus invalid configuration file.</td>
<td>Import the Fieldbus configuration file as described in the Fieldbus I/O manual.</td>
<td></td>
<td>If the problem persists, contact Epson.</td>
</tr>
<tr>
<td>7912</td>
<td>Fieldbus invalid device ID.</td>
<td>An invalid device ID parameter was used. Check your program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7913</td>
<td>Fieldbus invalid service was specified.</td>
<td>An invalid explicit messaging service number was used. Check your program.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7914</td>
<td>Fieldbus cannot open master board driver.</td>
<td>Check that the Fieldbus master board is installed correctly. Check that the drivers for the Fieldbus master board were installed correctly.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7915</td>
<td>Fieldbus cannot open master board channel.</td>
<td>Check that no other application is using the Fieldbus master board. Check that the correct firmware is installed (downloaded) on the board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7916</td>
<td>Fieldbus host ready timeout.</td>
<td>Check that the drivers for the Fieldbus master board were installed correctly. Reboot and try again. Replace the board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7917</td>
<td>Fieldbus bus communication timeout.</td>
<td>Check the bus power and slave connections.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7950</td>
<td>Force sensing. Invalid force sensor.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7951</td>
<td>Force sensing. Invalid force sensor axis.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7952</td>
<td>Force sensing. Sensor read failed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7953</td>
<td>Force sensing. Sensor initialization failed.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7954</td>
<td>Force sensing. Sensor not initialized.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7955</td>
<td>Force sensing. Force or torque exceeded saturation level.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7975</td>
<td>Force Guide. Value out of range.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7976</td>
<td>Force Guide. Invalid property value.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>9001</td>
<td>Emergency stop circuit failure was detected. Disconnection or other failure was found in one of the redundant inputs.</td>
<td>Check whether no disconnection, earth fault, or short-circuit of the emergency stop input signal exits. Then reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9002</td>
<td>Safeguard circuit failure was detected. Disconnection or other failure was found in one of the redundant inputs.</td>
<td>Check whether no disconnection, earth fault, or short-circuit of the safeguard input signal exits. Then reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9003</td>
<td>Initialization failure. Failed to initialize the firmware.</td>
<td>This is likely because of the controller hardware failure. Check the wiring is correct. If the error is not cleared after the controller is rebooted, contact us.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9004</td>
<td>Initialization failure. Failed to initialize the DU. Check the DU power and the connection.</td>
<td>The number of set Drive Unit(s) disagrees with the number of recognized Drive Unit(s). Check the wirings of power supply and between Control Unit and Drive Unit are correct. If the error is not cleared after the controller is rebooted, contact us.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9005</td>
<td>Initialization failure. Failed to initialize the DU. Check the connection.</td>
<td>This is likely because of the Drive Unit hardware failure. Check the wiring is correct. If the error is not cleared after the controller is rebooted, contact us.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9006</td>
<td>Initialization failure. Failed to initialize the Remote I/O. Check the Remote I/O setting.</td>
<td>Check the Remote I/O setting value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9007</td>
<td>Error of Force Sensor occurs. Note 1: Each error code See each error code to take a relevant countermeasure.</td>
<td>Error of Force Sensor has occurred. Please confirm Note 1 by the system history, and take a relevant countermeasure.</td>
<td>Each error code</td>
<td></td>
</tr>
<tr>
<td>9011</td>
<td>Battery voltage of the CPU board backup is lower than the specified voltage. Replace the CPU board battery.</td>
<td>Replace the battery for the CPU board immediately. Keep the controller ON as long as possible until the battery is replaced.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9012</td>
<td>5V input voltage for CPU board is lower than the specified voltage. If normal voltage is not generated by 5V power supply alone, replace the power supply.</td>
<td>If normal voltage is not generated by 5V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9013</td>
<td>24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage. If normal voltage is not generated by 24V power supply alone, replace the power supply.</td>
<td>If normal voltage is not generated by 24V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9014</td>
<td>Internal temperature of the Controller is higher than the specified temperature.</td>
<td>Stop the controller as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9015</td>
<td>Speed of the controller fan is below the specified speed. (FAN1)</td>
<td>Check whether the filter of the controller is not clogged up. Replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
</tbody>
</table>

---

**Note 1:** Each error code

See each error code to take a relevant countermeasure.

**Note 2:** Each error code
<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>9016</td>
<td>Speed of the controller fan is below the specified speed. (FAN2)</td>
<td>Check whether the filter of the controller is not clogged up. Replace the fan.</td>
<td>Current value</td>
<td>Boundary value</td>
</tr>
<tr>
<td>9017</td>
<td>Internal temperature of the Controller is higher than the specified temperature.</td>
<td>Stop the controller as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9018</td>
<td>54V input voltage for CPU board is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 54V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9019</td>
<td>3.3V input voltage for CPU board is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 3.3V power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9020</td>
<td>DC input voltage for CPU board is lower than or bigger than the specified voltage.</td>
<td>If normal voltage is not generated by DC power supply alone, replace the power supply.</td>
<td>100 times of current value</td>
<td>100 times of boundary value</td>
</tr>
<tr>
<td>9021</td>
<td>DU1 3.3V input voltage for the board is lower than the minimum allowed voltage.</td>
<td>If normal voltage is not generated by 3.3V of Drive Unit 1 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9022</td>
<td>DU1 5V input voltage for the board is lower than the minimum allowed voltage.</td>
<td>If normal voltage is not generated by 5V of Drive Unit 1 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9023</td>
<td>DU1 24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 24V of Drive Unit 1 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9024</td>
<td>DU1 Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop the Drive Unit 1 as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9025</td>
<td>DU1 Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter of the Drive Unit 1 is not clogged up. Replace the fan.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9026</td>
<td>DU1 Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter of the Drive Unit 1 is not clogged up. Replace the fan.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9031</td>
<td>DU2 3.3V input voltage for the board is lower than the minimum allowed voltage.</td>
<td>If normal voltage is not generated by 3.3V of Drive Unit 2 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9032</td>
<td>DU2 5V input voltage for the board is lower than the minimum allowed voltage.</td>
<td>If normal voltage is not generated by 5V of Drive Unit 2 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9033</td>
<td>DU2 24 V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 24V of Drive Unit 2 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>9034</td>
<td>DU2 Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop the Drive Unit 2 as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9035</td>
<td>DU2 Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter of the Drive Unit 2 is not clogged up. Replace the fan.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9036</td>
<td>DU2 Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter of the Drive Unit 2 is not clogged up. Replace the fan.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9041</td>
<td>DU3 3.3V input voltage for the board is lower than the minimum allowed voltage.</td>
<td>If normal voltage is not generated by 3.3V of Drive Unit 3 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9042</td>
<td>DU3 5V input voltage for the board is lower than the minimum allowed voltage.</td>
<td>If normal voltage is not generated by 5V of Drive Unit 3 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9043</td>
<td>DU3 24V input voltage for the motor brake, encoder and fan is lower than the specified voltage.</td>
<td>If normal voltage is not generated by 24V of Drive Unit 3 power supply alone, replace the power supply.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9044</td>
<td>DU3 Internal temperature of the Controller is higher than the allowed temperature.</td>
<td>Stop the Drive Unit 3 as soon as possible and check whether the ambient temperature of the controller is not high. Check whether the filter is not clogged up.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9045</td>
<td>DU3 Rotating speed of the controller fan is below the allowed speed. (FAN1)</td>
<td>Check whether the filter of the Drive Unit 3 is not clogged up. Replace the fan.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9046</td>
<td>DU3 Rotating speed of the controller fan is below the allowed speed. (FAN2)</td>
<td>Check whether the filter of the Drive Unit 3 is not clogged up. Replace the fan.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9100</td>
<td>Initialization failure. Failed to allocate memory.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9101</td>
<td>Message queue has become full.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9102</td>
<td>Initialization failure. Failed to initialize Modbus.</td>
<td>(When RTU is selected) Check whether the selected port is installed. (When TCP is selected) Check whether the selected port number is used by other</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9103</td>
<td>Initialization failure. Failed to initialize the user output.</td>
<td>If the manipulator is specified, check whether the specified manipulator is registered.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9104</td>
<td>Remote User Output failure. Specified command cannot be executed.</td>
<td>Check the condition expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9233</td>
<td>The Fieldbus I/O driver is in an abnormal state.</td>
<td>The module is broken or the controller software is damaged. Restore the controller firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9234</td>
<td>Fieldbus I/O driver initialization failure.</td>
<td>The module is broken or the controller software is damaged. Restore the controller firmware.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>9610</td>
<td>RAS circuit detected a servo system malfunction. Reboot the controller.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Check for noise. Replace the controller.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9611</td>
<td>Servo CPU internal RAM failure. Reboot the controller. Check for noise.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Replace the DMB.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9612</td>
<td>RAM for the main and servo CPU communication failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9613</td>
<td>Servo CPU internal RAM failure. Reboot the controller. Check for noise.</td>
<td>Reboot the controller. Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td>For T series, it is not DMB but CPU board.</td>
</tr>
<tr>
<td></td>
<td>Replace the DMB.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9614</td>
<td>Initialization communication of main CPU and servo CPU failure. Reboot the Controller. Check for noise. Replace DMB.</td>
<td>Reboot the controller. Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td>For T series, it is not DMB but CPU board.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9615</td>
<td>Initialization communication of the main and servo CPU failure. Reboot the controller. Check for noise. Replace DMB.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9616</td>
<td>Communication of the main and servo CPU failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller. Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td>For T series, it is not DMB but CPU board.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9617</td>
<td>Communication of the main and servo CPU failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller. Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td>For T series, it is not DMB but CPU board.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9618</td>
<td>Servo long time command overrun.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9619</td>
<td>Servo long time command check sum error.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9620</td>
<td>System watchdog timer detected a failure. Reboot the controller. Check for noise. Replace the DMB.</td>
<td>Reboot the controller. Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td>For T series, it is not DMB but CPU board.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9621</td>
<td>Drive unit check failure.</td>
<td>Check the noise countermeasures. Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>9622</td>
<td>RAM failure of the servo CPU.</td>
<td>Check the noise countermeasures.  Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9623</td>
<td>Failure of the redundant circuitry for the emergency stop or the safeguard. Check the wiring.</td>
<td>Check the noise countermeasures.  Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9624</td>
<td>Low voltage of the main circuit power supply was detected. Check the power supply voltage. Reboot the controller.</td>
<td>Check the noise countermeasures.  Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9625</td>
<td>Control relay contact of the main circuit power supply is welded closed. Replace the DPB.</td>
<td></td>
<td>Replace the DMB.</td>
<td></td>
</tr>
<tr>
<td>9626</td>
<td>Detect the recognition mismatch of the sub CPU and main CPU.</td>
<td>Reboot the controller, take the measure against noise, or replace the CPU board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9627</td>
<td>Temperature of regeneration resistor was higher than the specified temperature.</td>
<td>Specify the Weight/Inertia setting.  Check the load.  Check the robot. (Smoothness, backlash, non-smooth motion, loose belt tension, brake)  Check the interference with the peripheral equipment. (Collision, contact)  Check the model setting.  Check the power cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9628</td>
<td>Over voltage of the sub CPU.</td>
<td>Replace the DPB board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9630</td>
<td>Servo real time status failure. Check sum error.</td>
<td>Reboot the controller.  Replace the DMB.  Check the noise countermeasures.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9632</td>
<td>Servo real time status failure. Servo free running counter error</td>
<td>Reboot the controller.  Replace the DMB.  Check the noise countermeasures.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9633</td>
<td>Servo real time status failure. Servo CPU communication error.</td>
<td>Reboot the controller.  Replace the DMB.  Check the noise countermeasures.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9640</td>
<td>Irregular motion control interruption was detected. Interruption duplicate.</td>
<td>Reboot the controller.  Replace the DMB.  Check the noise countermeasures.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9691</td>
<td>Data sending failure in motion network.</td>
<td>Check the connection of the cable for Drive Unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9692</td>
<td>Data receiving failure in motion network.</td>
<td>Check the connection of the cable for Drive Unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>9697</td>
<td>Data sending failure of Force Sensor I/F board. Check connection of the Force Sensor I/F board and Force Sensor.</td>
<td>Check connection of the Force Sensor I/F board and Force Sensor.</td>
<td>Reboot the controller.</td>
<td>Please inquire with us if a similar error occurs even after the above countermeasures are taken.</td>
</tr>
<tr>
<td>9698</td>
<td>Data receiving failure of Force Sensor I/F board. Check connection of the Force Sensor I/F board and Force Sensor.</td>
<td>Check connection of the Force Sensor I/F board and Force Sensor.</td>
<td>Reboot the controller.</td>
<td>Please inquire with us if a similar error occurs even after the above countermeasures are taken.</td>
</tr>
<tr>
<td>9700</td>
<td>Servo control gate array failure. Check the DMB.</td>
<td>Check the short-circuit and improper connection of the peripheral equipment wiring. (Emergency and I/O connectors)</td>
<td>Replace the DMB.</td>
<td>Replace the additional axis unit. For T series, reboot the controller, take the measure against noise, and replace the CPU board and motor unit.</td>
</tr>
<tr>
<td>9701</td>
<td>Disconnection of the parallel encoder signal. Check the signal cable connection or the robot internal wiring.</td>
<td>Check the M/C cable signal. Check the robot signal wiring. (Missing pin, disconnection, short-circuit)</td>
<td>Replace the motor. (Encoder failure)</td>
<td>Replace the DMB. (Detection circuit failure)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the connector connection in the controller. (Loosening, connecting to the serial encoder terminal on the DMB)</td>
<td>Check the model setting. (Improperly setting of the parallel encoder)</td>
<td>Check the peripheral equipment wiring. (Emergency and I/O)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
</tr>
<tr>
<td>9702</td>
<td>Motor driver is not installed. Install the motor driver. Check the DMB or the motor driver.</td>
<td>Check whether the motor driver is mounted. Check the model setting and hardware setting.</td>
<td>Replace the motor driver.</td>
<td>Replace the DMB.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>For T series, check the wiring of the motor unit.</td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>9703</td>
<td>Initialization communication failure of incremental encoder. Check the signal cable connection and the robot setting.</td>
<td>Check the model setting. Replace the motor. (Encoder failure) Replace the DMB. For T series, reboot the controller, take the measure against noise, and replace the CPU board and motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9704</td>
<td>Initialization failure of absolute encoder. Check the signal cable connection or the robot setting.</td>
<td>Check the model setting. Replace the motor. (Encoder failure) Replace the DMB. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9705</td>
<td>Encoder division setting failure. Check the robot setting.</td>
<td>Check the model setting.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9706</td>
<td>Data failure at the absolute encoder initialization. Check the signal cable connection, the controller, or the motor.</td>
<td>Replace the motor. (Encoder failure) Replace the DMB. Check the noise countermeasures. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9707</td>
<td>Absolute encoder multi-turn is beyond the maximum range. Reset the encoder.</td>
<td>Reset the encoder. Replace the motor. (Encoder failure)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9708</td>
<td>Position is out of the range. Reset the encoder.</td>
<td>Reset the encoder. Replace the DMB. Replace the motor. (Encoder failure)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9709</td>
<td>No response from the serial encoder. Check the signal cable connection, the motor, the DMB, or the encoder I/F board.</td>
<td>Check the model setting. (Improperly setting of the parallel encoder model) Check the signal cable connection. Replace the DMB and encoder I/F board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9710</td>
<td>Serial encoder initialization failure. Reboot the controller. Check the motor, the DMB, or the encoder I/F board.</td>
<td>Check the robot configuration. Check the signal cable. Replace the DMB and encoder I/F board. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9711</td>
<td>Serial encoder communication failure. Reboot the controller. Check the motor, the DMB, or the encoder I/F board.</td>
<td>Check the robot configuration. Check the signal cable. Replace the DMB and encoder I/F board. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>9712</td>
<td>Servo CPU watchdog timer failure. Reboot the controller. Check the motor or the DMB.</td>
<td>Replace the DMB. Check the noise countermeasures. For T series, check the connection of the signal cable. Reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9713</td>
<td>Current control circuit WDT failure. Reboot the controller. Check the controller.</td>
<td>Check the power cable connection. Check the 15V power supply and cable connection. Replace the DMB. Check the noise countermeasures. For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9714</td>
<td>The DMB is not for this robot.</td>
<td>Check robot setting. Replace with the supported DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9715</td>
<td>Encoder is reset. Reboot the controller.</td>
<td>Reboot the controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9716</td>
<td>Power supply failure of the absolute encoder. Replace the battery to a new one. Check the robot internal wiring.</td>
<td>Reset the encoder. Check the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9717</td>
<td>Backup data failure of the absolute encoder. Reset the encoder.</td>
<td>Reset the encoder. Check the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9718</td>
<td>Absolute encoder battery alarm.</td>
<td>Replace the battery. Check the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9719</td>
<td>Position failure of the absolute encoder. Reset the encoder. Replace the motor.</td>
<td>Reset the encoder. Replace the motor. (Encoder failure) For T series, reboot the controller, take the measure against noise, and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9720</td>
<td>Speed is too high at controller power ON. Stop the robot and reboot the controller.</td>
<td>Reboot the controller. Reset the encoder. For T series, reboot the controller, take the measure against noise, and replace the motor unit. Check the interference with the other devices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9721</td>
<td>Absolute encoder over heat.</td>
<td>Lower the motion duty. Wait until the temperature of the encoder decreases.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9722</td>
<td>R/D transducer failure. Reset the encoder. Check resolver board or the robot internal wiring.</td>
<td>Resets the encoder. Check the signal wiring of the manipulator (loose pin, disconnection, short). Replace the resolver board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>9723</td>
<td>G sensor communication failure. Check the signal cable connection or the robot internal wiring.</td>
<td>Check for the signal cable connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the signal wiring of the manipulator (loose pin, disconnection, short).</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the noise countermeasure.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the control board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9724</td>
<td>G sensor data error. Check for the control board.</td>
<td>Replace the control board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9725</td>
<td>The multi rotational data and the R/D conversion data is different. Reset the encoder.</td>
<td>Reset the resolver.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the noise countermeasure.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the resolver board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9726</td>
<td>Disconnection of the resolver excitation signal. Reset the encoder. Check the resolver board or the robot internal wiring.</td>
<td>Check the signal wiring of the manipulator (loose pin, disconnection, short).</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the resolver board.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9727</td>
<td>S-DSP communication failure. Check the DMB.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the noise countermeasure.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9728</td>
<td>Current feedback data failure. Check the DMB.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the noise countermeasure.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For T series, check the short-circuit and earth fault of the power cable. Reboot the controller or replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9729</td>
<td>D-DSP communication failure. Check the DMB.</td>
<td>Reboot the Controller.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check the noise countermeasure.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the DMB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9730</td>
<td>Speed is too high at controller power OFF. Reset the encoder.</td>
<td>Reset the encoder.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the motor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9731</td>
<td>Speed is too high. Reset the encoder.</td>
<td>Reset the encoder.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Replace the motor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For T series, reboot the controller and replace the motor unit.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9732</td>
<td>Servo alarm A.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Message</td>
<td>Remedy</td>
<td>Note 1</td>
<td>Note 2</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>10000</td>
<td>Command aborted by user</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10001</td>
<td>Command timeout.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10002</td>
<td>Bad point file line syntax</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10003</td>
<td>Project could not be built.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10004</td>
<td>Cannot initialize Spel class instance.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10005</td>
<td>Cannot initialize parser.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10006</td>
<td>Cannot initialize wbproxy.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10007</td>
<td>Project does not exist.</td>
<td>Check whether the project name and the path are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10008</td>
<td>No project specified.</td>
<td>Specify the project.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10009</td>
<td>Cannot open file.</td>
<td>Check whether the project name and the path are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10010</td>
<td>Cannot create file.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10011</td>
<td>File not found</td>
<td>Check whether the project name and the path are correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10013</td>
<td>Cannot execute LoadPoints with Robot Manager open.</td>
<td>Close the robot manager and execute.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10014</td>
<td>Project cannot be locked.</td>
<td>Terminate other applications.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10015</td>
<td>Project could not be synchronized.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10016</td>
<td>Drive not ready</td>
<td>Check whether the drive designation is correct.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10017</td>
<td>Invalid IP address</td>
<td>Check the IP address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10018</td>
<td>Invalid IP mask</td>
<td>Check the IP mask.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10019</td>
<td>Invalid IP gateway</td>
<td>Check the IP gateway.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10020</td>
<td>IP address or gateway cannot be the subnet address.</td>
<td>Check the IP address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10021</td>
<td>IP address or gateway cannot be the broadcast address.</td>
<td>Check the IP address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10022</td>
<td>Invalid DNS address</td>
<td>Check the DNS.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10023</td>
<td>Commands cannot be executed because the project build is not complete.</td>
<td>Execute after the project build is completed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10024</td>
<td>Invalid task name</td>
<td>Check the task name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10100</td>
<td>Command already in cycle.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10101</td>
<td>Command aborted by user</td>
<td>Execute ResetAbort.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10102</td>
<td>Invalid server instance.</td>
<td>Specify the correct instance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10103</td>
<td>Invalid CommandTask value.</td>
<td>Specify the correct task number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10104</td>
<td>Cannot change ServerInstance after initialized.</td>
<td>Set ServerInstance before initialization.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10501</td>
<td>Connection aborted.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10502</td>
<td>Cannot connect with the controller.</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10503</td>
<td>Controller firmware is not compatible with this version of RC+.</td>
<td>Upgrade the RC+ version.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10504</td>
<td>USB connection of this system is reserved for the RC620 Controller and cannot be used for RC+7.0.</td>
<td>Install the RC+7.0 to another computer.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## SPEL+ Error Messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Remedy</th>
<th>Note 1</th>
<th>Note 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>10505</td>
<td>The specified connection does not exist.</td>
<td>Check the connection number.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10600</td>
<td>Frame grabber driver not installed.</td>
<td>Install the driver.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbortMotion</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Abs</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Accel</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AccelMax</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>AccelR</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AccelS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Acos</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Agl</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>AglToPls</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>AIO_In</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>AIO_InW</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>AIO_Out</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AIO_OutW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AIO_Set</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AIO_TrackingSet</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>AIO_TrackingStart</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AIO_TrackingEnd</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>AIO_TrackingOn</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Align</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>AlignECP</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>And</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>AOpen</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Arc</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Arc3</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Arch</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Arm</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ArmClr</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ArmDef</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ArmSet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Asc</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Asin</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Atan</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Atan2</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ATCLR</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>AtHome</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ATRQ</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>AutoLJM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AvoidSingularity</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Base</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BClr</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>BClr64</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>BGo</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>BMove</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Boolean</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>BOpen</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Box</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BoxCler</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>BoxDef</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Brake</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

*Notes:*
- ✓: Command can be used in the command window.
- ✓: Command can be used as a statement in the SPEL+ program.
- ✓: Command can be used as a function.
## Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RC+</td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>BSet</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BSet64</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BTst</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>BTst64</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Byte</td>
<td>-</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>C Calib</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Call</td>
<td>-</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>CalPls</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ChDir</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ChDisk</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ChDrive</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ChkCom</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ChkNet</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Chr$</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ClearPoints</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Close</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>CloseCom</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>CloseDB</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>CloseNet</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cls</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cnv AbortTrack</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cnv Accel</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Downstream</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Fine</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv LPulse</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Mode</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Name$</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Number</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv OffsetAngle</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Point</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv PosErr</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Pulse</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv QueAdd</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv QueGet</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv QueLen</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv QueList</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cnv QueMove</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cnv QueReject</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv QueRemove</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cnv QueUserData</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv RobotConveyor</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Speed</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Trigger</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cnv Upstream</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CollisionDetect</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cont</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Copy</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Cos</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ctr</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CTReset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>CtrlDev</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CtrlInfo</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CurDir$</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CurDisk$</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CurDrive$</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CurPos</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Curve</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
</tbody>
</table>
## Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVMove</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CP_Offset</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CR</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CT</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CU</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CX</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CY</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CZ</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Date</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Date$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Declare</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>DegToRad</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Del</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DeleteDB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DispDev</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Dist</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Do...Loop</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Double</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ECP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ECPClr</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ECPDef</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ECPSet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ElapsedTime</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Elbow</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Eof</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Era</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>EResume</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Err$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Err</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Errb</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ErrMsg$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Error</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ErrorOn</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Ert</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>EStopOn</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Eval</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Exit</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ExportPoints</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FbusIO_GetBusStatus</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>FbusIO_GetDeviceStatus</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>FbusIO_SendMsg</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FileDataTime$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>FileExists</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>FileLen</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Find</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FindPos</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Fine</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FineDist</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FineStatus</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Fix</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Flush</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FmtStr</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FmtStr$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>FolderExists</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Command</td>
<td>Command window</td>
<td>Program</td>
<td>Function</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>For...Next</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Force Calibrate</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Force ClearTrigger</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Force GetForce</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Force GetForces</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Force Sensor</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Force SetTrigger</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>FreeFile</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Function...Fend</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>GClose</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GetCurrentUser$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>GetRobotInsideBox</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>GetRobotInsidePlane</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>GGet</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Global</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Go</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Gosub...Return</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Goto</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>GSet</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>GShow</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>GShowDialog</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Halt</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Hand</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>HealthCalcPeriod</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthCtrlAlarmOn</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HealthCtrlInfo</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthCtrlRateOffset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthCtrlReset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthCtrlWarningEnable</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>HealthRateCtrlInfo</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRateRBInfo</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBAAlarmOn</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBAAnalysis</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBDistance</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBInfo</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HealthRBRateOffset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBReset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBSpeed</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HealthRBSStart</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBStop</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>HealthRBTRQ</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HealthRBWarningEnable</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Here</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Hex$</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Hof$</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Home</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HomeClr</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HomeDef</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>HomeSet</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Hordr</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Hour</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>If...Then...Else...EndIf</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ImportPoints</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>In</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>InBCD</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Inertia</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>InPos</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Input</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RC+</td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>Input #</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>InputBox</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>InReal</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>InsideBox</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>InsidePlane</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>InStr</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Int</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Int32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InW</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>IODef</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>IOLabel$</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>IONumber</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>J1Angle</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>J4Angle</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>J1Flag</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>J2Flag</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>J4Flag</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>J6Flag</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Joint</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>JRange</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JTran</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Jump</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump3</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump3CP</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JumpTLZ</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCase$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Len</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimitTorque</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimitTorqueLP</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimitTorqueStop</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimitTorqueStopLP</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimZ</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimZMargin</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Input</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Input #</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LJM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LoadPoints</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LocalClr</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LocalDef</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LogIn</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Long</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSet$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LShift</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LShift64</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LTrim$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M Mask</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCal</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCalComplete</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RC+</td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>M Cordr</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Mem In</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Mem In W</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Mem Off</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mem On</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Mem Out</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Mem Out W</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Mem Sw</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Mid $</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>M Hour</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Mk Dir</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Mod</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Motor</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Move</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>MsgBox</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>My Task</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>N Next</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Not</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>O Off</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>OL Accl</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>OL Rate</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>On</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>On Err</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Op BCD</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Open Com</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Open DB</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Open Net</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>O port</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Or</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Out</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Out Real</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Out W</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>P#</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>PA gl</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Pallet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Pallet Clr</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Parse Str</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Pass</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Pause</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Pause On</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>P Description</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>P Description $</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>P Def</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>P Del</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Perform Mode</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PG Fast Stop</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>PG L Speed</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PG Scan</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>PG Slow Stop</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>P Label</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>P Label $</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Plane</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Plane Clr</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Plane Def</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>PL list</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>PL Local</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Pls</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>P Number</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
</tbody>
</table>
## Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RC+</td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>PosFound</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Power</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PPIs</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Preserve</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Print</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Print #</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PTRCLR</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PTPBoost</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PTPBoostOK</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>PTPTime</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>PTran</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PTRQ</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Pulse</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Q</td>
<td>Ø</td>
<td>Ø</td>
<td>-</td>
</tr>
<tr>
<td>QP</td>
<td>Ø</td>
<td>Ø</td>
<td>-</td>
</tr>
<tr>
<td>QPDeceI</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>QPDeceS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Quit</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>R</td>
<td>RadToDeg</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Randmize</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Range</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Read</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ReadBin</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Real</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>RealAccel</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RealPls</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RealPos</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RealTorque</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Recover</td>
<td>✓</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RecoverPos</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Redim</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Rename</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>RenDir</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Reset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>ResetElapsedTime</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Restart</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Resume</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Return</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Right$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RmDir</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Rnd</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Robot</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RobotInfo</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RobotInfo$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RobotModel$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RobotName$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RobotSerial$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RobotType</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ROpen</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>RSet$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RShift64</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RShift</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RTrim$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>RunDialog</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>S</td>
<td>SafetyOn</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>SavePoints</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Seek</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Select...Send</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SelectDB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Command</td>
<td>Command window</td>
<td>Program</td>
<td>Function</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>Sense</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetCom</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetIn</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetInReal</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetInW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetLatch</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetNet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SetSw</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SFree</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Sgn</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Short</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Shutdown</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Signal</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SimGet</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>SimSet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Sin</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>SingularityAngle</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SingularityDist</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SingularitySpeed</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SLock</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SoftCP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Space$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Speed</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SpeedFactor</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SpeedR</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SpeedS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SPELCom_Event</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Sqr</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>ST</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>StartMain</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Stat</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Str$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>String</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Sw</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>SyncLock</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>SyncUnlock</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>SyncRobots</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SysConfig</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>SysErr</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>Tab$</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>Tan</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>TargetOK</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>TaskDone</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>TaskInfo</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>TaskInfo$</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>TaskState</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>T</td>
<td>TaskWait</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TCLim</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TCPSpeed</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>TCSpeed</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TeachOn</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>TGo</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Till</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TillOn</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Time</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Time$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>TLCln</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Command</td>
<td>Command window</td>
<td>Program</td>
<td>Function</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
<td>---------</td>
<td>----------</td>
</tr>
<tr>
<td>TLDef</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TLSet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TMOut</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TMove</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Tmr</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>TmReset</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Ton</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>Tool</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Trap</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Trim$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>TW</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>UBound</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>UByte</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>UCase$</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>UInt32</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UOpen</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>UpdateDB</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>UShort</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VCal</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VCalPoints</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VCls</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VCreateCalibration</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VCreateObject</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VCreateSequence</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VDefArm</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VDefGetMotionRange</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VDefLocal</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VDefSetMotionRange</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VDefTool</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VDeleteCalibration</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VDeleteObject</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VDeleteSequence</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VGGet</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VGioCenter</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VisCalib</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VisCallInfo</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VisCalLoad</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VisCalSave</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VisTrans</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VLoad</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VRUn</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VSave</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VSaveImage</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VSD</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VSet</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VShowModel</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>VStatsReset</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VStatsResetAll</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VStatsSave</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VStatsShow</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VTeach</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VTrain</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>VxCalib</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>VxCalDelete</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VxCalLoad</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VxCallInfo</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
</tbody>
</table>
## Appendix A: SPEL+ Command Use Condition List

<table>
<thead>
<tr>
<th>Command</th>
<th>Command window</th>
<th>Program</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RC+</td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>VxCalSave</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VxTrans</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>W Wait</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WaitNet</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WaitPos</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WaitSig</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Weight</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Where</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>WindowStatus</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_Add</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_AutoRemove</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_Get</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_Len</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_List</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>WorkQue_Reject</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_Remove</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_Sort</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WorkQue_UserData</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WOpen</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Wrist</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Write</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WriteBin</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>X Xor</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Xqt</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>XY</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>XYLim</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>XYLimClr</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>XYLimDef</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
</tbody>
</table>
Appendix B: Precaution of Compatibility

B-1: Precaution of EPSON RC+ 6.0 Compatibility

Overview

This section contains information for customers using EPSON RC+ 7.0 with RC700 Controller that have already used EPSON RC+ 6.0 with RC620.

EPSON RC+ 7.0 and EPSON RC+ 6.0 differ in such as hardware, adaptable manipulators, number of joint allowance, and software execution environment. Please read this section and understand the contents for the safety use of the Robot system.

EPSON RC+ 7.0 is improved software that has compatibility with products before EPSON RC+ 7.0 and designed to innovate advanced software technologies. However, some parts do not have compatibility with EPSON RC+ 6.0 or have been deleted to specialize in the robot controller and for ease of use.

The following compatibility is indicated based on EPSON RC+ 6.0 compared to EPSON RC+ 7.0.

General Differences

General differences of EPSON RC+ 6.0 and EPSON RC+ 7.0 are as follows.

<table>
<thead>
<tr>
<th>Item</th>
<th>EPSON RC+ 7.0</th>
<th>EPSON RC+ 6.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of task</td>
<td>Up to 32 tasks (Background task : Up to 16 tasks)</td>
<td>Up to 32 tasks (Background task : Up to 16 tasks)</td>
</tr>
<tr>
<td>Type of task</td>
<td>Able to specify NoPouse task Able to specify NoEmgAbort task Able to specify Background task</td>
<td>Able to specify NoPouse task Able to specify NoEmgAbort task Able to specify Background task</td>
</tr>
<tr>
<td>Special TRAP such as TRAP ERROR</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Task starts by TRAP number</td>
<td>Dedicated task number</td>
<td>Dedicated task number</td>
</tr>
<tr>
<td>Multi Manipulator</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Robot number</td>
<td>1 to 16</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Number of significant figure for Real type</td>
<td>6 digits</td>
<td>6 digits</td>
</tr>
<tr>
<td>Number of significant figure for Double type</td>
<td>14 digits</td>
<td>14 digits</td>
</tr>
<tr>
<td>Array elements number</td>
<td>Other than string variable</td>
<td>Other than string variable</td>
</tr>
<tr>
<td></td>
<td>Local variable 2,000</td>
<td>Local variable 2,000</td>
</tr>
<tr>
<td></td>
<td>Global variable 100,000</td>
<td>Global variable 100,000</td>
</tr>
<tr>
<td></td>
<td>Module variable 100,000</td>
<td>Module variable 100,000</td>
</tr>
<tr>
<td></td>
<td>Global Preserve variable 4,000</td>
<td>Global Preserve variable 4,000</td>
</tr>
<tr>
<td>String variable</td>
<td>Other than string variable</td>
<td>Other than string variable</td>
</tr>
<tr>
<td></td>
<td>Local variable 200</td>
<td>Local variable 200</td>
</tr>
<tr>
<td></td>
<td>Global variable 10,000</td>
<td>Global variable 10,000</td>
</tr>
<tr>
<td></td>
<td>Module variable 10,000</td>
<td>Module variable 10,000</td>
</tr>
<tr>
<td></td>
<td>Global Preserve variable 400</td>
<td>Global Preserve variable 400</td>
</tr>
<tr>
<td>Device number</td>
<td>21:PC</td>
<td>21:PC</td>
</tr>
<tr>
<td></td>
<td>22:REMOTE</td>
<td>22:REMOTE</td>
</tr>
<tr>
<td></td>
<td>24:TP</td>
<td>24:TP</td>
</tr>
<tr>
<td></td>
<td>20:TP3</td>
<td>28:LCD</td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Item</th>
<th>EPSON RC+ 7.0</th>
<th>EPSON RC+ 6.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control device</td>
<td>Remote I/O</td>
<td>Remote I/O</td>
</tr>
<tr>
<td></td>
<td>PC</td>
<td>PC</td>
</tr>
<tr>
<td></td>
<td>REMOTE COM</td>
<td>REMOTE COM</td>
</tr>
<tr>
<td></td>
<td>REMOTE Ethernet</td>
<td>REMOTE Ethernet</td>
</tr>
<tr>
<td></td>
<td>TP3</td>
<td>TP3</td>
</tr>
<tr>
<td>Timer number range</td>
<td>0 to 63</td>
<td>0 to 63</td>
</tr>
<tr>
<td>Program capacity</td>
<td>8 MB</td>
<td>8 MB</td>
</tr>
<tr>
<td>Signal No range</td>
<td>0 to 63</td>
<td>0 to 63</td>
</tr>
<tr>
<td>for SyncLock, SyncUnlock</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory I/O port</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>I/O port number</td>
<td>Common with EPSON RC+ 6.0</td>
<td>Common with EPSON RC+ 6.0</td>
</tr>
<tr>
<td>Port No of Ethernet</td>
<td>201 to 216</td>
<td>201 to 216</td>
</tr>
<tr>
<td>Remote I/O assignment</td>
<td>Assigned as default</td>
<td>Default: --</td>
</tr>
<tr>
<td>Port No of RS-232C communication</td>
<td>1 to 8, 1001 to 1008</td>
<td>1 to 8, 1001, 1002</td>
</tr>
<tr>
<td>OpenCom execution of RS-232C communication</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Input/output to files</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>File number</td>
<td>30 to 63</td>
<td>30 to 63</td>
</tr>
<tr>
<td>Access number for the database</td>
<td>501 to 508</td>
<td>501 to 508</td>
</tr>
<tr>
<td>Vision Guide</td>
<td>Network camera type</td>
<td>Network camera type</td>
</tr>
<tr>
<td></td>
<td>Frame grubber type</td>
<td>Frame gruber type</td>
</tr>
<tr>
<td>Conveyor tracking</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PG robot</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>OCR</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Security</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>VBGuide 6.0 (RC+ API 7.0)</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Fieldbus I/O</td>
<td>Use normal I/O commands</td>
<td>Use normal I/O commands</td>
</tr>
<tr>
<td>Fieldbus master</td>
<td>Response is not guaranteed</td>
<td>Response is not guaranteed</td>
</tr>
<tr>
<td>Fieldbus slave</td>
<td>Response is guaranteed</td>
<td>Response is guaranteed</td>
</tr>
<tr>
<td>GUI Builder</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Error number</td>
<td>Common with EPSON RC+ 6.0</td>
<td>Common with EPSON RC+ 6.0</td>
</tr>
</tbody>
</table>
## Compatibility List of Commands

+ Function expansion / function changes have been made with upper compatibility.
− No changes.
! Pay attention. Function changes or syntax changes have been made.
!! Pay attention. Significant changes have been made.
× Deleted.

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Abs Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Accel Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Accel Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelMax Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelR Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelS Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelS Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Acos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AglToPls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Agl Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AlignECP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Align Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>And Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arc Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arc3 Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arch Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arch Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arm Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arm Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmDef Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmSet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Asc Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Asin Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Atan Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Atan2 Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ATCLR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ATRQ Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ATRQ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>B Base Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Base Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BClr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BGo Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>BMove Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Boolean Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Box Statement</td>
<td>+</td>
<td>Added the remote output logic designation</td>
</tr>
<tr>
<td>Box Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BoxClr Function</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoxDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brake Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Brake Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BSet Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BTst Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Call Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ChkCom Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ChkNet Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chr$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ClearPoints Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CloseCom Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CloseNet Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cis Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cos Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CP Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CP Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTReset Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ctr Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CtrlDev Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CtrlInfo Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CurPos Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Curve Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CVMove Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CX to CW Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CX to CW Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DegToRad Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DispDev Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DispDev Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dist Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do...Loop Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Double Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECP Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECP Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECPClr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EcpDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECPSet Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECPSet Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elbow Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elbow Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Era Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Erase Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EResume Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Erf$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Compatibility</td>
<td>Note</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Erl Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Err Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ErrMsg$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ErrorOn Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ert Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EStopOn Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exit Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F Find Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FindPos Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fine Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fine Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fix Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FmtStr$ Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For...Next</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function...Fend</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G Global Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>Go Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gosub...Return</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Goto Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H Halt Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hand Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hand Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Here Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Here Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hex$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Home Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HomeClr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HomeDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HomeSet Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HomeSet Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOOrdr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOOrdr Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hour Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hour Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I If...EndIf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InBCD Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inertia Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inertia Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InPos Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input# Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InsideBox Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InsidePlane Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InStr Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Compatibility</td>
<td>Note</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Int Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Integer Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>InW Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOLabel$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IONumber Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IONumber Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J J1Flag Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J1Flag Function</td>
<td></td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>J2Flag Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J2Flag Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J4Flag Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J4Flag Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J6Flag Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>J6Flag Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JA Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Joint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRange Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JRange Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JS Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JT Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JTran Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump3 Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jump3CP Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L LCase$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Len Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimZ Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LimZ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Input Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line Input# Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LJM Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LoadPoints</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LocalClr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LocalDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lof Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Long Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSet$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LShift Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LTrim$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M Mask Operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MemIn Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MemInW Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MemOff Statement</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>MemOn Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MemOut Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MemOutW Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MemSw Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mid$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mod Operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motor Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motor Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MyTask Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N Not Operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>O Off Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLAccel Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLAccel Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLRate Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OLRate Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>On Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OnErr</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpBCD Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenCom Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenNet Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oport Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Or Operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Out Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Out Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OutW Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OutW Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P PAgI Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pallet Statement</td>
<td>+ Added coordinate value designation</td>
<td></td>
</tr>
<tr>
<td>Pallet Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ParsStr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ParsStr Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pass Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pause Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PauseOn Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PDel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLabel$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLabel Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plane Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plane Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PlaneClr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PlaneDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLList Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLLabel Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLDel</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This table lists various commands and their compatibility details. The plus symbol (+) indicates an added coordinate value designation.
<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIs Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PNumber Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PosFound Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Power Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Power Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PPls Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Print Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Print# Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTCLR Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTPBoost Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTPBoost Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTPBoostOK Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTPTimer Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTTran Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTRQ Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>PTRQ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Pulse Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Pulse Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>QP Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Quit Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RadToDeg Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Randnize Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Range Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Read Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>ReadBin Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Real Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RealPls Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RealPos Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RealTorque Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Redim Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Reset Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Resume Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Return Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RobotInfo Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RobotInfo$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RobotModel$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RobotName$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RobotSerial$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RobotType Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RSet$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RShift Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>RTrim$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>SafetyOn Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>SavePoints Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Select...Send Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Sense Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Compatibility</td>
<td>Note</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>SetCom Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SetInW Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SetIn Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SetNet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SetSw Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SFree Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SFree Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sgn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Signal Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sin Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SLock Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SoftCP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SoftCP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Space$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Speed Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Speed Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedR Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedS Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedS Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SPELCom_Event Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sqr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Stat Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Str$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>String Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sw Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SyncLock Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SyncUnlock Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SysConfig Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SysErr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tab$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tan Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TargetOK Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskDone Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskInfo Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskInfo$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskState Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskState Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskWait Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TC Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCLim Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCLim Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCSpeed Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCSpeed Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TGo Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>TillOn Function</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Command</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Time$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLClr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIDef Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLSet Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLSet Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMOut Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMove Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tmr Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TmReset Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toff Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ton Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tool Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tool Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trap Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trim$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tw Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>U UBound Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCase$ Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>V Val Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W Wait Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WaitNet Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WaitPos Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WaitSig Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weight Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weight Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Where Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wrist Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wrist Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WriteBin Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X Xor Operator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xqt Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XY Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XYLim Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XYLim Function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XYLimClr Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XYLimDef Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XYLimDef Function</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix B: Precaution of Compatibility

B-2: Precaution of EPSON RC+ 5.0 Compatibility

Overview

This section contains information for customers using EPSON RC+ 7.0 with RC700 and RC90 Controllers that have already used EPSON RC+ 5.0 with RC180.

EPSON RC+ 7.0 and EPSON RC+ 5.0 differ in such as hardware, adaptable manipulators, number of joint allowance, and software execution environment. Please read this section and understand the contents for the safety use of the Robot system.

EPSON RC+ 7.0 is improved software that has compatibility with products before EPSON RC+ 7.0 and designed to innovate advanced software technologies. However, some parts do not have compatibility with EPSON RC+ 5.0 or have been deleted to specialize in the robot controller and for ease of use.

The following compatibility is indicated based on EPSON RC+ 5.0 compared to EPSON RC+ 7.0.

General Differences

General differences of EPSON RC+ 5.0 and EPSON RC+ 7.0 are as follows.

<table>
<thead>
<tr>
<th>Item</th>
<th>EPSON RC+ 7.0</th>
<th>EPSON RC+ 5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of task</td>
<td>Up to 32 tasks</td>
<td>Up to 16 tasks</td>
</tr>
<tr>
<td>(Background task : Up to 16 tasks)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type of task</td>
<td>Able to specify NoPouse task</td>
<td>Able to specify NoPouse task</td>
</tr>
<tr>
<td></td>
<td>Able to specify NoEmgAbort task</td>
<td>Able to specify Background task</td>
</tr>
<tr>
<td>Special TRAP</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>such as TRAP ERROR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task starts by TRAP number</td>
<td>Dedicated task number</td>
<td>Dedicated task number</td>
</tr>
<tr>
<td>Multi Manipulator</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Robot number</td>
<td>1 to 16</td>
<td>1</td>
</tr>
<tr>
<td>Number of significant figure for</td>
<td>6 digits</td>
<td>6 digits</td>
</tr>
<tr>
<td>Real type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of significant figure for</td>
<td>14 digits</td>
<td>14 digits</td>
</tr>
<tr>
<td>Double type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array elements number</td>
<td>Other than string variable</td>
<td>Other than string variable</td>
</tr>
<tr>
<td></td>
<td>Local variable</td>
<td>Local variable</td>
</tr>
<tr>
<td></td>
<td>2,000</td>
<td>1,000</td>
</tr>
<tr>
<td></td>
<td>Global variable</td>
<td>Global variable</td>
</tr>
<tr>
<td></td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td></td>
<td>Module variable</td>
<td>Module variable</td>
</tr>
<tr>
<td></td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td></td>
<td>Global Preserve variable</td>
<td>Global Preserve variable</td>
</tr>
<tr>
<td></td>
<td>4,000</td>
<td>1,000</td>
</tr>
<tr>
<td></td>
<td>String variable</td>
<td>String variable</td>
</tr>
<tr>
<td></td>
<td>Local variable</td>
<td>Local variable</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>Global variable</td>
<td>Global variable</td>
</tr>
<tr>
<td></td>
<td>10,000</td>
<td>1,000</td>
</tr>
<tr>
<td></td>
<td>Module variable</td>
<td>Module variable</td>
</tr>
<tr>
<td></td>
<td>10,000</td>
<td>1,000</td>
</tr>
<tr>
<td></td>
<td>Global Preserve variable</td>
<td>Global Preserve variable</td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>100</td>
</tr>
<tr>
<td>Device number</td>
<td>21:PC</td>
<td>21:PC</td>
</tr>
<tr>
<td></td>
<td>22:REMOTE</td>
<td>22:REMOTE</td>
</tr>
<tr>
<td></td>
<td>24:TP</td>
<td>23:OP</td>
</tr>
<tr>
<td></td>
<td>20:TP3</td>
<td>24:TP</td>
</tr>
<tr>
<td>Control device</td>
<td>Remote I/O</td>
<td>Remote I/O</td>
</tr>
<tr>
<td></td>
<td>PC</td>
<td>PC</td>
</tr>
<tr>
<td></td>
<td>REMOTE COM</td>
<td>OP1</td>
</tr>
<tr>
<td></td>
<td>REMOTE Ethernet</td>
<td>REMOTE Ethernet</td>
</tr>
<tr>
<td></td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>Timer number range</td>
<td>0 to 63</td>
<td>0 to 15</td>
</tr>
<tr>
<td>Program capacity</td>
<td>8 MB</td>
<td>4 MB</td>
</tr>
<tr>
<td>Item</td>
<td>EPSON RC+ 7.0</td>
<td>EPSON RC+ 5.0</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>---------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Signal No range for SyncLock, SyncUnlock</td>
<td>0 to 63</td>
<td>0 to 15</td>
</tr>
<tr>
<td>Signal No range for WaitSig, Signal</td>
<td>0 to 63</td>
<td>0 to 5</td>
</tr>
<tr>
<td>Memory I/O port</td>
<td>1024</td>
<td>256</td>
</tr>
<tr>
<td>I/O port number</td>
<td>Common with EPSON RC+ 5.0</td>
<td></td>
</tr>
<tr>
<td>Port No of Ethernet</td>
<td>201 to 216</td>
<td>201 to 208</td>
</tr>
<tr>
<td>Remote I/O assignment</td>
<td>Assigned as default</td>
<td>Assigned as default</td>
</tr>
<tr>
<td>Port No of RS-232C communication</td>
<td>1 to 8, 1001 to 1008</td>
<td>1 to 8</td>
</tr>
<tr>
<td>OpenCom execution of RS-232C communication port</td>
<td>Mandatory</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Input/output to files</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>File number</td>
<td>30 to 63</td>
<td>Not supported</td>
</tr>
<tr>
<td>Access number for the database</td>
<td>501 to 508</td>
<td>Not supported</td>
</tr>
<tr>
<td>Vision Guide</td>
<td>Network camera type</td>
<td>Network camera type</td>
</tr>
<tr>
<td>Conveyor tracking</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>PG robot</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>OCR</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Security</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>VBGuide 5.0 (RC+ API 7.0)</td>
<td>Supported</td>
<td>VBGuide Lite is supported</td>
</tr>
<tr>
<td>Fieldbus I/O</td>
<td>Use normal I/O commands</td>
<td>Use normal I/O commands</td>
</tr>
<tr>
<td>Fieldbus master</td>
<td>Response is not guaranteed</td>
<td>Not supported</td>
</tr>
<tr>
<td>Fieldbus slave</td>
<td>Response is guaranteed</td>
<td>Response is guaranteed</td>
</tr>
<tr>
<td>GUI Builder</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Error number</td>
<td>Common with EPSON RC+ 5.0</td>
<td>Not supported</td>
</tr>
</tbody>
</table>
## Compatibility List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Abs Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Accel Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Accel Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AccelMax Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AccelR Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AccelR Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AccelIS Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AccelIS Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Acos Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AglToPls Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Agl Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>AlignECP Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Align Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>And Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Arc Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Arc3 Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Arch Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Arch Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Arm Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Arm Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ArmClr Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ArmDef Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ArmSet Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ArmSet Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Asc Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Asin Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Atan Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Atan2 Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ATCLR Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ATRQ Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>ATRQ Function</td>
<td>−</td>
</tr>
<tr>
<td>B</td>
<td>Base Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Base Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>BClr Function</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>BGo Statement</td>
<td>+ Added PerformMode parameter.</td>
</tr>
<tr>
<td></td>
<td>BMove Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Boolean Statement</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>Box Statement</td>
<td>+ Added the robot number designation</td>
</tr>
<tr>
<td></td>
<td>Box Function</td>
<td>+ Added the robot number designation</td>
</tr>
<tr>
<td></td>
<td>BoxClr Function</td>
<td>+ Added the robot number designation</td>
</tr>
<tr>
<td></td>
<td>BoxDef Function</td>
<td>+ Added the robot number designation</td>
</tr>
<tr>
<td></td>
<td>Brake Statement</td>
<td>−</td>
</tr>
<tr>
<td>Command</td>
<td>Compatibility</td>
<td>Note</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>Brake Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BTst Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Byte Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>C Call Statement</td>
<td>+</td>
<td>DLL function Call is supported</td>
</tr>
<tr>
<td>ChkCom Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ChkNet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Chr$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ClearPoints Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CloseCom Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CloseNet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Cls Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Cos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CTRReset Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Ctr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CtrlDev Function</td>
<td>!</td>
<td>Changed device ID</td>
</tr>
<tr>
<td>CtrlInfo Function</td>
<td>−</td>
<td>Changed the obtaining contents</td>
</tr>
<tr>
<td>CurPos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Curve Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CVMove Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CX to CW Statement</td>
<td>+</td>
<td>Added CR, CS, CT</td>
</tr>
<tr>
<td>CX to CW Function</td>
<td>+</td>
<td>Added CR, CS, CT</td>
</tr>
<tr>
<td>D Date Statement</td>
<td>!</td>
<td>Only displays</td>
</tr>
<tr>
<td>Date$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>DegToRad Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>DispDev Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>DispDev Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Dist Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Do...Loop Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Double Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>E ECP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECPClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>EcpDef Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECPSet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECPSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ElapsedTime Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Elbow Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Elbow Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Era Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erase Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>EResume Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erf$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erl Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Err Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ErrMsg$ Function</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ErrorOn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Ert Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>EStopOn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Exit Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Find Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FindPos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Fine Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Fine Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Fix Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FmtStr$ Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>For...Next</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Function...Fend</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Go Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>Gosub...Return</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Goto Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Halt Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hand Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hand Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Here Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Here Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hex$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Home Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HomeClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HomeDef Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HomeSet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HomeSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HOOrdr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HOOrdr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hour Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hour Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>If...EndIf</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>In Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InBCD Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Inertia Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Inertia Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InPos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Input Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Input# Statement</td>
<td>+</td>
<td>Added the device number</td>
</tr>
<tr>
<td>InsideBox Function</td>
<td>!</td>
<td>Added the designation of robot number and All Cannot use with Wait statement</td>
</tr>
<tr>
<td>InsidePlane Function</td>
<td>!</td>
<td>Added the designation of robot number and All Cannot use with Wait statement</td>
</tr>
<tr>
<td>InStr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Int Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Integer Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InW Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Compatibility</td>
<td>Note</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>IOLabel$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>IONumber Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>IONumber Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J J1Flag Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J1Flag Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J2Flag Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J2Flag Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J4Flag Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J4Flag Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J6Flag Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>J6Flag Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>JA Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Joint</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>JRange Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>JRange Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>JS Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>JT Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>JTran Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Jump Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>Jump3 Statement</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>Jump3CP Statement</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>L LCase$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Left$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Len Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LimZ Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LimZ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Line Input Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Line Input# Statement</td>
<td>+</td>
<td>Added the device number</td>
</tr>
<tr>
<td>LJM Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LoadPoints</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Local Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Local Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LocalClr Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LocalDef Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Lof Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Long Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LSet$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LShift Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>LTrim$ Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>M Mask Operator</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemIn Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemInW Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemOff Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemOn Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemOut Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemOutW Statement</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>MemSw Function</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>Mid$ Function</td>
<td>—</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mod Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Motor Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Motor Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Move Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>MyTask Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>N Not Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>O Off Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OLAccel Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OLAccel Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OLRate Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OLRate Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>On Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OnErr</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OpBCD Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OpenCom Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OpenNet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Oport Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Or Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Out Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Out Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OutW Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OutW Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PAgl Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pallet Statement</td>
<td>−</td>
<td>Added the coordinate designation</td>
</tr>
<tr>
<td>Pallet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ParsStr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ParsStr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pass Statement</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>Pause Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PauseOn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PDef Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PDel</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PLabel$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PLabel Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Plane Statement</td>
<td>+</td>
<td>Added the robot number designation</td>
</tr>
<tr>
<td>Plane Function</td>
<td>+</td>
<td>Added the robot number designation</td>
</tr>
<tr>
<td>PlaneClr Statement</td>
<td>+</td>
<td>Added the robot number designation</td>
</tr>
<tr>
<td>PlaneDef Function</td>
<td>+</td>
<td>Added the robot number designation</td>
</tr>
<tr>
<td>PLList Statement</td>
<td>!</td>
<td>Changed the display type</td>
</tr>
<tr>
<td>PLocal Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PLocal Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PNumber Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PosFound Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Power Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Power Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PPPls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Print Statement</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print# Statement</td>
<td>+</td>
<td>Changed the device number</td>
</tr>
<tr>
<td>PTCLR Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTPBoost Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTPBoost Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTPBoostOK Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTPTime Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTran Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTRQ Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PTRQ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Pulse Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Pulse Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>QP Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Quit Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>R RadToDeg Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Randmize Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Range Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Read Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ReadBin Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Real Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RealPts Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RealPos Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RealTorque Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Redim Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Reset Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ResetElapsedTime Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Resume Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Return Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RobotInfo Function</td>
<td>+</td>
<td>Added the information</td>
</tr>
<tr>
<td>RobotInfo$ Function</td>
<td>+</td>
<td>Added the display of default point file name</td>
</tr>
<tr>
<td>RobotModel$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RobotName$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RobotSerial$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RobotType Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RSet$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RShift Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RTrim$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>S SafetyOn Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SavePoints Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Select...Send Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Sense Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SetCom Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SetInW Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SetIn Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SetNet Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SetSw Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SFree Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SFree Function</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>
Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sgn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Signal Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sin Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SLock Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SoftCP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SoftCP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Space$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Speed Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Speed Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedR Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedS Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SpeedS Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SPELCom_Event Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sqr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Stat Function</td>
<td>+</td>
<td>Added the information</td>
</tr>
<tr>
<td>Str$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>String Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Sw Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SyncLock Statement</td>
<td>!</td>
<td>Error occurs by executing SyncLock repeatedly</td>
</tr>
<tr>
<td>SyncUnlock Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>SysConfig Statement</td>
<td>+</td>
<td>Added the information</td>
</tr>
<tr>
<td>SysErr Function</td>
<td>+</td>
<td>Added the function to retrieve the warnings</td>
</tr>
<tr>
<td>Tab$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tan Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TargetOK Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskDone Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskInfo Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskInfo$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskState Statement</td>
<td>+</td>
<td>Added the display of background task</td>
</tr>
<tr>
<td>TaskState Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TaskWait Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TC Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCLim Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCLim Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCSpeed Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TCSpeed Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TGo Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>TillOn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Time Command</td>
<td>!</td>
<td>Only displays</td>
</tr>
<tr>
<td>Time Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Time$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TLClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TlDef Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TLSet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TLSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TMOut Statement</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMove Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tmr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>TmReset Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Toff Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Ton Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tool Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tool Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Trap Statement</td>
<td>!</td>
<td>Added the Trap that interrupts the controller status</td>
</tr>
<tr>
<td>Trim$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Tw Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>UBound Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>UCase$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Val Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Wait Statement</td>
<td>!</td>
<td>Added the global variables and others as the wait condition</td>
</tr>
<tr>
<td>WaitNet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>WaitPos Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>WaitSig Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Weight Statement</td>
<td>+</td>
<td>Added the designation of S, T</td>
</tr>
<tr>
<td>Weight Function</td>
<td>+</td>
<td>Added the designation of S, T</td>
</tr>
<tr>
<td>Where Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Wrist Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Wrist Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Write Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>WriteBin Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Xor Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Xqt Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XY Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLim Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLim Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLimClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLimDef Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLimDef Function</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
## Commands from EPSON RC+ Ver.4.* (Not supported in EPSON RC+ 5.0)

<table>
<thead>
<tr>
<th>EPSON RC+ Statement</th>
<th>EPSON RC+ Function</th>
<th>EPSON RC+ 5.0 Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aopen Statement</td>
<td>Cnv_QueUserData Function</td>
<td>Hofs Function</td>
</tr>
<tr>
<td>BOpen Statement</td>
<td>Cnv_RobotConveyor Function</td>
<td>ImportPoints Statement</td>
</tr>
<tr>
<td>Calib Statement</td>
<td>Cnv_Speed Function</td>
<td>InputBox Statement</td>
</tr>
<tr>
<td>CalPls Statement</td>
<td>Cnv_Trigger Statement</td>
<td>LogIn Function</td>
</tr>
<tr>
<td>ChDir Statement</td>
<td>Cnv_Upstream Function</td>
<td>MCalComplete Function</td>
</tr>
<tr>
<td>ChDrive Statement</td>
<td>Cont Statement</td>
<td>MCal Statement</td>
</tr>
<tr>
<td>Close Statement</td>
<td>Copy Statement</td>
<td>MCordr Statement</td>
</tr>
<tr>
<td>Cnv_AbortTrack Statement</td>
<td>CurDir$ Function</td>
<td>Recovery Function</td>
</tr>
<tr>
<td>Cnv_Downstream Statement</td>
<td>CurDrive$ Function</td>
<td>MKDir Statement</td>
</tr>
<tr>
<td>Cnv_Fine Statement</td>
<td>Declare Statement</td>
<td>MsgBox Statement</td>
</tr>
<tr>
<td>Cnv_Fine Function</td>
<td>Del Statement</td>
<td>Rename Statement</td>
</tr>
<tr>
<td>Cnv_Name$ Function</td>
<td>Eof Function</td>
<td>Rename Statement</td>
</tr>
<tr>
<td>Cnv_Number Function</td>
<td>Eval Function</td>
<td>RenDir Statement</td>
</tr>
<tr>
<td>Cnv_Point Function</td>
<td>FbusIO_GetBusStatus Function</td>
<td>Restart Statement</td>
</tr>
<tr>
<td>Cnv_PosErr Function</td>
<td>FbusIO_GetDeviceStatus Function</td>
<td>Restart Statement</td>
</tr>
<tr>
<td>Cnv_Pulse Function</td>
<td>FbusIO_SendMsg Statement</td>
<td>Robot Statement</td>
</tr>
<tr>
<td>Cnv_QueAdd Statement</td>
<td>FileDateTime$ Function</td>
<td>Robot Function</td>
</tr>
<tr>
<td>Cnv_QueGet Function</td>
<td>FileExists Function</td>
<td>ROpen Statement</td>
</tr>
<tr>
<td>Cnv_QueLen Function</td>
<td>FileLen Function</td>
<td>RunDialog Statement</td>
</tr>
<tr>
<td>Cnv_QueList Statement</td>
<td>FolderExists Function</td>
<td>Seek Statement</td>
</tr>
<tr>
<td>Cnv_QueMove Statement</td>
<td>FreeFile Function</td>
<td>Shutdown Statement</td>
</tr>
<tr>
<td>Cnv_QueReject Statement</td>
<td>GetCurrentUser$ Statement</td>
<td>UOpen Statement</td>
</tr>
<tr>
<td>Cnv_QueReject Function</td>
<td>Hofs Statement</td>
<td>UOpen Statement</td>
</tr>
<tr>
<td>Cnv_QueRemove Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cnv_QueUserData Function</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix B: Precaution of Compatibility

B-3: Precaution of EPSON RC+ Ver.4.* Compatibility

Overview

This section contains information for customers using EPSON RC+ 7.0 with RC700 Controller that have already used EPSON RC+ Ver.4.* with RC520 or RC420. EPSON RC+ 7.0 and EPSON RC+ Ver.4.* differs in such as hardware, adaptable manipulators, number of joint allowance, and software execution environment. Please read this section and understand the contents for the safety use of the Robot system. EPSON RC+ 7.0 is improved software that has compatibility with products before EPSON RC+ 7.0 and designed to innovate advanced software technologies. However, some parts do not have compatibility with EPSON RC+ Ver.4.* or have been deleted to specialize in the robot controller and for ease of use. The following compatibility is indicated based on EPSON RC+ Ver.4.* compared to EPSON RC+ 7.0.

General Differences

General differences of EPSON RC+ Ver.4.* and EPSON RC+ 7.0 are as follows.

<table>
<thead>
<tr>
<th>Item</th>
<th>EPSON RC+ 7.0</th>
<th>EPSON RC+ Ver.4.*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of task</td>
<td>Up to 32 tasks</td>
<td>Up to 32 tasks</td>
</tr>
<tr>
<td></td>
<td>(Background task : Up to 16 tasks)</td>
<td></td>
</tr>
<tr>
<td>Type of task</td>
<td>Able to specify NoPouse task</td>
<td>Able to specify NoPouse task</td>
</tr>
<tr>
<td></td>
<td>Able to specify NoEmgAbort task</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Able to specify Background task</td>
<td></td>
</tr>
<tr>
<td>Special TRAP such as TRAP ERROR</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Task starts by TRAP number</td>
<td>Dedicated task number</td>
<td>Task number only using 1 to 32</td>
</tr>
<tr>
<td>Multi manipulator</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Robot number</td>
<td>1 to 16</td>
<td>1 to 16</td>
</tr>
<tr>
<td>Number of significant figure for Real type</td>
<td>6 digits</td>
<td>7 digits</td>
</tr>
<tr>
<td>Number of significant figure for Double type</td>
<td>14 digits</td>
<td>15 digits</td>
</tr>
<tr>
<td>Array elements number</td>
<td>Other than string variable</td>
<td>As far as the memory remains</td>
</tr>
<tr>
<td></td>
<td>Local variable 2000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Global variable 1,000,000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Module variable 1,000,000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Global Preserve variable 4,000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>String variable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Local variable 200</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Global variable 10,000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Module variable 10,000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Global Preserve variable 400</td>
<td></td>
</tr>
<tr>
<td>Line number</td>
<td>Not supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Device number</td>
<td>21:PC</td>
<td>1:Controller</td>
</tr>
<tr>
<td></td>
<td>22:REMOTE</td>
<td>2:REMOTE</td>
</tr>
<tr>
<td></td>
<td>24:TP</td>
<td>3:OP</td>
</tr>
<tr>
<td></td>
<td>20:TP3</td>
<td></td>
</tr>
<tr>
<td>Control device</td>
<td>Remote I/O PC</td>
<td>Remote I/O PC</td>
</tr>
<tr>
<td></td>
<td>Remote COM</td>
<td>PC</td>
</tr>
<tr>
<td></td>
<td>REMOTE Ethernet</td>
<td>OP500RC</td>
</tr>
<tr>
<td></td>
<td>TP3</td>
<td></td>
</tr>
<tr>
<td>Timer number range</td>
<td>0 to 63</td>
<td>0 to 63</td>
</tr>
<tr>
<td>Program capacity</td>
<td>8 MB</td>
<td>4 MB</td>
</tr>
</tbody>
</table>

EPSON RC+ 7.0 (Ver.7.4) SPEL+ Language Reference Rev.9
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Item</th>
<th>EPSON RC+ 7.0</th>
<th>EPSON RC+ Ver.4.*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal No range for SyncLock, SyncUnlock</td>
<td>0 to 63</td>
<td>1 to 32</td>
</tr>
<tr>
<td>Signal No range for WaitSig, Signal</td>
<td>0 to 63</td>
<td>0 to 127</td>
</tr>
<tr>
<td>Memory I/O port</td>
<td>1024</td>
<td>512</td>
</tr>
<tr>
<td>I/O port number</td>
<td>Different with EPSON RC+4.0</td>
<td></td>
</tr>
<tr>
<td>Port No of Ethernet</td>
<td>201 to 216</td>
<td>128 to 147</td>
</tr>
<tr>
<td>Remote I/O assignment</td>
<td>Assigned as default</td>
<td>Default: --</td>
</tr>
<tr>
<td>Port No of RS-232C communication</td>
<td>1 to 8, 1001, 1008</td>
<td>1 to 16</td>
</tr>
<tr>
<td>OpenCom execution of RS-232C communication port</td>
<td>Mandatory</td>
<td>Optional</td>
</tr>
<tr>
<td>Input/output to files</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>File number for the file access</td>
<td>30 to 63</td>
<td>30 to 63</td>
</tr>
<tr>
<td>Access number for the database</td>
<td>501 to 508</td>
<td>Not supported</td>
</tr>
<tr>
<td>Vision Guide</td>
<td>Network camera type</td>
<td>Frame grubber type</td>
</tr>
<tr>
<td>Conveyor tracking</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>PG robot</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>OCR</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Security</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>VBGuide (RC+ API 7.0)</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Fieldbus I/O</td>
<td>Use normal I/O commands</td>
<td>Use special commands</td>
</tr>
<tr>
<td>Fieldbus master</td>
<td>Response is not guaranteed</td>
<td>Response is not guaranteed</td>
</tr>
<tr>
<td>Fieldbus slave</td>
<td>Response is guaranteed</td>
<td>Response is not guaranteed</td>
</tr>
<tr>
<td>GUI Builder</td>
<td>Supported</td>
<td>Not supported</td>
</tr>
<tr>
<td>Group in the project</td>
<td>Not supported</td>
<td>Supported</td>
</tr>
<tr>
<td>Error number</td>
<td>Different with EPSON RC+ Ver.4.*</td>
<td></td>
</tr>
</tbody>
</table>
Compatibility List of Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Abs Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Accel Statement</td>
<td>+</td>
<td>Able to specify more than 100 for some robots</td>
</tr>
<tr>
<td>Accel Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelR Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelIS Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AccelIS Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Acos Function</td>
<td>+</td>
<td>Argument range check has been added</td>
</tr>
<tr>
<td>Agl Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AglToPls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>And Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>AOpen Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Arc Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arc3 Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arch Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arch Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arm Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Arm Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmSet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ArmSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Asin Function</td>
<td>+</td>
<td>Argument range check has been added</td>
</tr>
<tr>
<td>Atan Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Atan2 Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ATCLR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ATRQ Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ATRQ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>B Base Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BClr Function</td>
<td>+</td>
<td>Argument range check has been added</td>
</tr>
<tr>
<td>Beep Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>BGo Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>BMove Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Boolean Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BOpen Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Brake Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>BSet Function</td>
<td>+</td>
<td>Argument range check has been added</td>
</tr>
<tr>
<td>BTst Function</td>
<td>+</td>
<td>Argument range check has been added</td>
</tr>
<tr>
<td>Byte Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Calib Statement</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CalPls Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CalPls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Chain Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>ChDir Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ChDrive Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ChkCom Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ChkNet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Chr$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Clear Statement</td>
<td>!</td>
<td>Renamed to ClearPoints</td>
</tr>
<tr>
<td>Close Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CloseCom Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CloseNet Statement</td>
<td>+</td>
<td>Able to specify All</td>
</tr>
<tr>
<td>ClrScr Statement</td>
<td>!</td>
<td>Renamed toCls</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Device ID can be specified for arguments</td>
</tr>
<tr>
<td>Conv **</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Cont Statement</td>
<td>!</td>
<td>Able to execute by the setting</td>
</tr>
<tr>
<td>Copy Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Cos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Ctr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CTRest Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CtrlDev Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>CtrlDev Function</td>
<td>!</td>
<td>Changed device ID</td>
</tr>
<tr>
<td>CtrlInfo Function</td>
<td>!!</td>
<td>Changed the obtaining contents</td>
</tr>
<tr>
<td>CurDir$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CurDrive$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CurPos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Curve Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CVMove Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>CX to CW Statement</td>
<td>+</td>
<td>Added CR, CS, CT</td>
</tr>
<tr>
<td>CX to CW Function</td>
<td>+</td>
<td>Added CR, CS, CT</td>
</tr>
<tr>
<td>Date Statement</td>
<td>!</td>
<td>Only displays</td>
</tr>
<tr>
<td>Date$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Declare Statement</td>
<td>!</td>
<td>The processing is slow</td>
</tr>
<tr>
<td>DegToRad Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Del Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Dir Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Dist Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Do...Loop Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Double Statement</td>
<td>!</td>
<td>Significant figure is 14 digits</td>
</tr>
<tr>
<td>EClr Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>ECP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECP Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECPClr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ECPSet Statement</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECPSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Elbow Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Elbow Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ENetIO_****</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Eof Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>EPrint Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Era Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erase Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>EResume Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erf$ Function</td>
<td>+</td>
<td>Able to omit the task number</td>
</tr>
<tr>
<td>EPrint Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Era Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erase Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>EResume Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Erf$ Function</td>
<td>+</td>
<td>Able to omit the task number</td>
</tr>
<tr>
<td>Error Statement</td>
<td>+</td>
<td>Able to specify task number for arguments</td>
</tr>
<tr>
<td>Ert Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>EStopOn Function</td>
<td>+</td>
<td>Able to specify Wait</td>
</tr>
<tr>
<td>Eval Function</td>
<td>!</td>
<td>Differences in the error output</td>
</tr>
<tr>
<td>Exit Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FbusIO_****</td>
<td>×</td>
<td>Normal I/O command available</td>
</tr>
<tr>
<td>FileDateTime$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FileExists Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FileLen Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Find Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FindPos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Fine Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Fine Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Fix Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FmtStr$ Statement</td>
<td>!!</td>
<td>Function is limited significantly</td>
</tr>
<tr>
<td>FoldrExist Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>For...Next</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>FreeFile Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Function...Fend</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>GetCurrentUser$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Global Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Go Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>Gosub...Return</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Goto Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Halt Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hand Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hand Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Here Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Here Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hex$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hofs Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hofs Function</td>
<td>−</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HomeSet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HomeSet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HORdr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HORdr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hour Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Hour Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>HTest Statement</td>
<td>×</td>
<td>Extension “.pnt” has changed to “.pts”</td>
</tr>
<tr>
<td>HTest Function</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>If...EndIf</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ImportPoints Statement</td>
<td>!</td>
<td>Extension “.pnt” has changed to “.pts”</td>
</tr>
<tr>
<td>In Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>In($n) Statement</td>
<td>×</td>
<td>Replaced to MemIn</td>
</tr>
<tr>
<td>InBCD Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Inertia Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Inertia Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InPos Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Input Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Input# Statement</td>
<td>+</td>
<td>Input is available from devices</td>
</tr>
<tr>
<td>InputBox Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InStr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Int Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Integer Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InW Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>InW($n) Statement</td>
<td>×</td>
<td>Replaced to MemInW</td>
</tr>
<tr>
<td>IONumber Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>J4Flag Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>J4Flag Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>J6Flag Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>J6Flag Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>JA Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>JRange Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>JRange Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>JS Function</td>
<td>!</td>
<td>Returns True/False</td>
</tr>
<tr>
<td>JT Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>JTran Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Jump Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>Jump3 Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Jump3CP Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Kill Statement</td>
<td>×</td>
<td>Replaced with Del</td>
</tr>
<tr>
<td>LCase$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Left$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Len Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>LimZ Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>LimZ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Line Input Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Line Input# Statement</td>
<td>+</td>
<td>Input is available from devices</td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadPoints</td>
<td>!</td>
<td>Extension “.pnt” has changed to “.pts”</td>
</tr>
<tr>
<td>Local Statement</td>
<td>!</td>
<td>Local number “0” is an error</td>
</tr>
<tr>
<td>Local Function</td>
<td>–</td>
<td>Local number “0” is an error</td>
</tr>
<tr>
<td>LocalClr Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Lof Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>LogIn Statement</td>
<td>!</td>
<td>Changed from a statement to a function</td>
</tr>
<tr>
<td>Long Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>LPrint Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>LSet$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>LShift Function</td>
<td>+</td>
<td>Argument range check has been added</td>
</tr>
<tr>
<td>LTrim$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Mask Operator</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MCal Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MCalComplete Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MCoFs Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>MCoFs Function</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>MCordr Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MCordr Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MCorrg Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>MemIn Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MemInW Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MemOff Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MemOn Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MemOut Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MemOutW Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MemSw Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Mid$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MKDir Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Mod Operator</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Motor Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Motor Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Move Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MsgBox Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>MyTask Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>NNot Operator</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Off Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Off$ Statement</td>
<td>×</td>
<td>Replaced to MemOff</td>
</tr>
<tr>
<td>OLRate Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>OLRate Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>On Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>On$ Statement</td>
<td>×</td>
<td>Replaced to MemOn</td>
</tr>
<tr>
<td>OnErr</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>OP *</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>OpBCD Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>OpenCom Statement</td>
<td>!</td>
<td>OpenCom is mandatory</td>
</tr>
<tr>
<td>OpenNet Statement</td>
<td>–</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oport Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Or Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Out Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Out Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Out$ Statement</td>
<td>×</td>
<td>Replaced to MemOut</td>
</tr>
<tr>
<td>OutW Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OutW Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>OutWS Statement</td>
<td>×</td>
<td>Replaced to MemOutW</td>
</tr>
<tr>
<td>PPAgl Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pallet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pallet Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ParsStr Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>ParsStr Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pass Statement</td>
<td>+</td>
<td>Able to specify continuous point</td>
</tr>
<tr>
<td>Pause Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PauseOn Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PDef Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PDel</td>
<td>+</td>
<td>Argument check has been added</td>
</tr>
<tr>
<td>PLabel$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PLabel Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PList</td>
<td>!!</td>
<td>Changed the display type; Argument check has been added; Function of Plist* has been deleted</td>
</tr>
<tr>
<td>PLocal Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PLocal Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Pls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PNumber Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Point Assignment</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Point Expression</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>POrient Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>POrient Function</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>PosFound Function</td>
<td>!</td>
<td>Returns True/False</td>
</tr>
<tr>
<td>Power Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Power Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PPls Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Print Statement</td>
<td>!</td>
<td>Outputs all flags at point output; Sets the output digit number of Double type and Real type to significant figure</td>
</tr>
<tr>
<td>Print# Statement</td>
<td>!</td>
<td>Same as Print Statement; Enables Print to each devices</td>
</tr>
<tr>
<td>PTCCLR Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PTPBoost Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PTPBoost Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PTPBoostOK Function</td>
<td>!</td>
<td>Returns True/False</td>
</tr>
<tr>
<td>PTPTime Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PTRQ Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>PTRQ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Compatibility</td>
<td>Note</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Pulse Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Pulse Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>QP Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Quit Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RadToDeg Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Randmize Statement</td>
<td>+</td>
<td>Seed value can be specified</td>
</tr>
<tr>
<td>Range Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Read Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ReadBin Statement</td>
<td>+</td>
<td>Able to read multiple bytes to array variable</td>
</tr>
<tr>
<td>Real Statement</td>
<td>!</td>
<td>6 digit significant figure</td>
</tr>
<tr>
<td>Recover Statement</td>
<td>!</td>
<td>Able to execute by the setting</td>
</tr>
<tr>
<td>Redim Statement</td>
<td>!</td>
<td>Element number is limited Array called by reference cannot be executed</td>
</tr>
<tr>
<td>Rename Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RenDir Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Reset Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Resume Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Restart Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Reset Statement</td>
<td>+</td>
<td>Added Reset Error</td>
</tr>
<tr>
<td>Return Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Right$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RmDir Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Rnd Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Robot Statement</td>
<td>+</td>
<td>Added the RS series</td>
</tr>
<tr>
<td>Robot Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RobotModel$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RobotType Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ROpen Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>RSet$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RShift Function</td>
<td>+</td>
<td>Argument check has been added</td>
</tr>
<tr>
<td>RTrim$ Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RunDialog Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>+</td>
<td>Able to specify Wait</td>
</tr>
<tr>
<td>SafetyOn Function</td>
<td>+</td>
<td>Able to specify Wait</td>
</tr>
<tr>
<td>SavePoints Statement</td>
<td>!</td>
<td>Extension (.pnt) has changed to (.pts)</td>
</tr>
<tr>
<td>Seek Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Select...Send</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Sense</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SetCom Statement</td>
<td>!</td>
<td>Cannot specify “56000” for the transfer rate</td>
</tr>
<tr>
<td>SetNet Statement</td>
<td>-</td>
<td>Port with OpenCom cannot be executed</td>
</tr>
<tr>
<td>SFree Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SFree Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Sgn Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Shutdown Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Signal Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Sin Function</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SLock Statement</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Space$ Function</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Speed Function</td>
<td>+</td>
<td>Argument optional</td>
</tr>
<tr>
<td>SpeedR Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>SpeedR Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>SpeedS Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>SpeedS Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>SPELCom_Event Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>SPELCom_Return Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Sqr Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Stat Function</td>
<td>!</td>
<td>Some information cannot be retrieved</td>
</tr>
<tr>
<td>Str$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>String Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Sw Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Sw($) Function</td>
<td>×</td>
<td>Replaced to MemSw</td>
</tr>
<tr>
<td>SyncLock Statement</td>
<td>!</td>
<td>Error occurs by executing SyncLock repeatedly</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lock is released when the task is completed</td>
</tr>
<tr>
<td>SyncUnlock Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>T Tab$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Tan Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TargetOK Function</td>
<td>!</td>
<td>Returns True/False</td>
</tr>
<tr>
<td>TaskDone Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TaskState Function</td>
<td>!</td>
<td>6 specified tasks do not return while Wait statement execution</td>
</tr>
<tr>
<td>TaskWait Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TGo Statement</td>
<td>+</td>
<td>Added PerformMode parameter.</td>
</tr>
<tr>
<td>TillOn Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Time Command</td>
<td>!</td>
<td>Only displays</td>
</tr>
<tr>
<td>Time Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Time$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TLClr Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TLSet Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TLSet Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TMOut Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TMove Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Tmr Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>TmReset Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Tool Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Tool Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Trap Statement</td>
<td>!!</td>
<td>Compatibility with Trap Goto</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Trap Gosub abolished and replaced to Trap Call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Trap Call is renamed to Trap Xqt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added Trap Finish</td>
</tr>
<tr>
<td>Trim$ Function</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Tw Function</td>
<td>!</td>
<td>Returns True/False</td>
</tr>
<tr>
<td>Type Statement</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>U UBound Function</td>
<td>–</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix B: Precaution of Compatibility

<table>
<thead>
<tr>
<th>Command</th>
<th>Compatibility</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>UCase$ Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>UOpen Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>V Val Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Ver Statement</td>
<td>×</td>
<td>Replaced to SysConfig</td>
</tr>
<tr>
<td>Verinit Statement</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>W Wait Statement</td>
<td>+</td>
<td>Added the global variables and others as the wait condition</td>
</tr>
<tr>
<td>WaitNet Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>WaitPos Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>WaitSig Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Weight Statement</td>
<td>+</td>
<td>Added the designation of S, T</td>
</tr>
<tr>
<td>Weight Function</td>
<td>+</td>
<td>Added the designation of S, T</td>
</tr>
<tr>
<td>Where Statement</td>
<td>!</td>
<td>Coordinate value always displays 6-axis</td>
</tr>
<tr>
<td>While..Wend</td>
<td>×</td>
<td>Replaced to Do...Loop</td>
</tr>
<tr>
<td>WOpen Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Wrist Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Wrist Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Write Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>WriteBin Statement</td>
<td>+</td>
<td>Multiple bytes can be listed from the array variable</td>
</tr>
<tr>
<td>X Xor Operator</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Xqt Statement</td>
<td>+</td>
<td>Able to specify NoEmgAbort</td>
</tr>
<tr>
<td>XY Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLim Statement</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>XYLim Function</td>
<td>−</td>
<td></td>
</tr>
<tr>
<td>Z ZeroFlg Function</td>
<td>×</td>
<td></td>
</tr>
</tbody>
</table>
## Appendix C: Commands of EPSON RC+7.0

### C-1: List of Commands Added EPSON RC+4.0 or Later

<table>
<thead>
<tr>
<th>Command</th>
<th>Syntax</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbortMotion Statement</td>
<td>CS Statement</td>
<td>HealthCalePeriod Statement</td>
</tr>
<tr>
<td>AccelMax Function</td>
<td>CS Function</td>
<td>HealthCalePeriod Function</td>
</tr>
<tr>
<td>AglToPls Function</td>
<td>CT Statement</td>
<td>HealthCtrlAlarmOn Function</td>
</tr>
<tr>
<td>AIO_Out Statement</td>
<td>CT Function</td>
<td>HealthCtrlInfo Statement</td>
</tr>
<tr>
<td>AIO_Out Function</td>
<td>CtrlDev Function</td>
<td>HealthCtrlWarningEnable Function</td>
</tr>
<tr>
<td>AIO_OutW Statement</td>
<td>Curve Statement</td>
<td>HealthCtrlRateOffset Statement</td>
</tr>
<tr>
<td>AIO_OutW Function</td>
<td>CVMove Statement</td>
<td>HealthCtrlWarningEnd Function</td>
</tr>
<tr>
<td>AIO_Set Statement</td>
<td>Cnv_Accel Statement</td>
<td>HealthRateCtrlInfo Function</td>
</tr>
<tr>
<td>AIO_Set Function</td>
<td>Cnv_Accel Function</td>
<td>HealthRateRBInfo Function</td>
</tr>
<tr>
<td>AIO_TrackingSet Statement</td>
<td>Cnv_DownStream Statement</td>
<td>HealthRateRBAlarmOn Function</td>
</tr>
<tr>
<td>AIO_TrackingStart Statement</td>
<td>Cnv_Mode Statement</td>
<td>HealthRateRBAnalysis Function</td>
</tr>
<tr>
<td>AIO_TrackingEnd Statement</td>
<td>Cnv_Mode Function</td>
<td>HealthRateRBAnalysis Function</td>
</tr>
<tr>
<td>AIO_TrackingOn Function</td>
<td>Cnv_OffsetAngle Statement</td>
<td>HealthRBDistance Statement</td>
</tr>
<tr>
<td>AIO_In Function</td>
<td>Cnv_OffsetAngle Function</td>
<td>HealthRBDistance Function</td>
</tr>
<tr>
<td>AIO_InW Function</td>
<td>Cnv_Steamupstream Statement</td>
<td>HealthRBInfo Function</td>
</tr>
<tr>
<td>Align Function</td>
<td>CollisionDetect Statement</td>
<td>HealthRBInfo Function</td>
</tr>
<tr>
<td>AlignECP Function</td>
<td>CollisionDetect Function</td>
<td>HealthRBRateOffset Statement</td>
</tr>
<tr>
<td>ArmDef Function</td>
<td>DegToRad Function</td>
<td>HealthRBReset Function</td>
</tr>
<tr>
<td>ATCLR Statement</td>
<td>DeleteDB Statement</td>
<td>HealthRBSpeed Statement</td>
</tr>
<tr>
<td>AtHome Function</td>
<td>DispDev Statement</td>
<td>HealthRBStart Statement</td>
</tr>
<tr>
<td>ATRQ Statement</td>
<td>DispDev Function</td>
<td>HealthRBStop Statement</td>
</tr>
<tr>
<td>ATRQ Function</td>
<td>Dist Function</td>
<td>HealthRBTRQ Statement</td>
</tr>
<tr>
<td>AutoLJM Statement</td>
<td>EcpDef Function</td>
<td>HealthRBTRQ Function</td>
</tr>
<tr>
<td>AutoLJM Function</td>
<td>ElapsedTime Function</td>
<td>HealthRBWarningEnable Statement</td>
</tr>
<tr>
<td>AvoidSingularity Statement</td>
<td>HealthCalcPeriod Function</td>
<td>HealthRBWarningEnable Function</td>
</tr>
<tr>
<td>AvoidSingularity Function</td>
<td>HealthCtrlAlarmOn Function</td>
<td></td>
</tr>
<tr>
<td>BClr Function</td>
<td>HealthCtrlInfo Statement</td>
<td></td>
</tr>
<tr>
<td>BClr64 Function</td>
<td>HealthCtrlRateOffset Statement</td>
<td></td>
</tr>
<tr>
<td>Box Statement</td>
<td>HealthCtrlWarningEnable Function</td>
<td></td>
</tr>
<tr>
<td>Box Function</td>
<td>Here Function</td>
<td></td>
</tr>
<tr>
<td>BoxCtri Function</td>
<td>Hex$ Function</td>
<td></td>
</tr>
<tr>
<td>BoxDef Function</td>
<td>HomeClr Statement</td>
<td></td>
</tr>
<tr>
<td>Brake Function</td>
<td>HomeDef Function</td>
<td></td>
</tr>
<tr>
<td>BSet Function</td>
<td>FindPos Function</td>
<td></td>
</tr>
<tr>
<td>BSet64 Function</td>
<td>Find Statement</td>
<td></td>
</tr>
<tr>
<td>BTst Function</td>
<td>FineDist Statement</td>
<td></td>
</tr>
<tr>
<td>BTst64 Function</td>
<td>FineDist Function</td>
<td></td>
</tr>
<tr>
<td>ChDisk Statement</td>
<td>FineStatus Function</td>
<td></td>
</tr>
<tr>
<td>ChkCom Function</td>
<td>Fix Function</td>
<td></td>
</tr>
<tr>
<td>ChkNet Function</td>
<td>Flush Statement</td>
<td></td>
</tr>
<tr>
<td>CloseCom Statement</td>
<td>Fmtstr Statement</td>
<td></td>
</tr>
<tr>
<td>CloseDB Statement</td>
<td>FunctionHere Statement</td>
<td></td>
</tr>
<tr>
<td>CloseNet Statement</td>
<td>GetRobotInsideBox Function</td>
<td></td>
</tr>
<tr>
<td>Cls Statement</td>
<td>GetRobotInsidePlane Statement</td>
<td></td>
</tr>
<tr>
<td>CP Statement</td>
<td>InReal Function</td>
<td></td>
</tr>
<tr>
<td>CP Function</td>
<td>InsideBox Function</td>
<td></td>
</tr>
<tr>
<td>CP_Offset Statement</td>
<td>InsidePlane Function</td>
<td></td>
</tr>
<tr>
<td>CP_Offset Function</td>
<td>InStr Function</td>
<td></td>
</tr>
<tr>
<td>CR Statement</td>
<td>IODef Function</td>
<td></td>
</tr>
<tr>
<td>CR Function</td>
<td>IOLabel$ Function</td>
<td></td>
</tr>
<tr>
<td>Cnv_Accel Statement</td>
<td>IONumber Function</td>
<td></td>
</tr>
<tr>
<td>Cnv_OffsetAngle Statement</td>
<td>J1Angle Statement</td>
<td></td>
</tr>
<tr>
<td>Cnv_UpstreamStatement</td>
<td>J1Angle Function</td>
<td></td>
</tr>
<tr>
<td>Cnv_Mode Statement</td>
<td>J4Angle Statement</td>
<td></td>
</tr>
<tr>
<td>Cnv_Mode Function</td>
<td>JA Function</td>
<td></td>
</tr>
<tr>
<td>Cnv_OffsetAngle Statement</td>
<td>Joint Statement</td>
<td></td>
</tr>
<tr>
<td>Cnv_OffsetAngle Function</td>
<td>JumpTLZ Statement</td>
<td></td>
</tr>
<tr>
<td>Cnv_OffsetAngle Function</td>
<td>JTran Statement</td>
<td></td>
</tr>
<tr>
<td>CVMove Statement</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------</td>
<td></td>
</tr>
<tr>
<td>LatchEnable</td>
<td>Enable latch</td>
<td></td>
</tr>
<tr>
<td>LatchState</td>
<td>Read latch state</td>
<td></td>
</tr>
<tr>
<td>LatchPos</td>
<td>Read latch position</td>
<td></td>
</tr>
<tr>
<td>LimZMargin</td>
<td>Read Z margin</td>
<td></td>
</tr>
<tr>
<td>LimZMargin Function</td>
<td>Read Z margin function</td>
<td></td>
</tr>
<tr>
<td>LimitTorque</td>
<td>Read torque limit</td>
<td></td>
</tr>
<tr>
<td>LimitTorque Function</td>
<td>Read torque limit function</td>
<td></td>
</tr>
<tr>
<td>LimitTorqueLP</td>
<td>Read torque limit LP</td>
<td></td>
</tr>
<tr>
<td>LimitTorqueLP Function</td>
<td>Read torque limit LP function</td>
<td></td>
</tr>
<tr>
<td>LimitTorqueStop</td>
<td>Read torque limit stop</td>
<td></td>
</tr>
<tr>
<td>LimitTorqueStop Function</td>
<td>Read torque limit stop function</td>
<td></td>
</tr>
<tr>
<td>LimitTorqueStopLP</td>
<td>Read torque limit stop LP</td>
<td></td>
</tr>
<tr>
<td>LimitTorqueStopLP Function</td>
<td>Read torque limit stop LP function</td>
<td></td>
</tr>
<tr>
<td>LJM Function</td>
<td>Read LJM</td>
<td></td>
</tr>
<tr>
<td>LocalDef</td>
<td>Local definition</td>
<td></td>
</tr>
<tr>
<td>LShift64</td>
<td>Left shift 64-bit</td>
<td></td>
</tr>
<tr>
<td>MemInW</td>
<td>Read memory in 16-bit</td>
<td></td>
</tr>
<tr>
<td>MemOutW</td>
<td>Write memory 16-bit</td>
<td></td>
</tr>
<tr>
<td>MemOutW Statement</td>
<td>Write memory 16-bit statement</td>
<td></td>
</tr>
<tr>
<td>MHour</td>
<td>Read M hour</td>
<td></td>
</tr>
<tr>
<td>OLAccel</td>
<td>Read OLA accel</td>
<td></td>
</tr>
<tr>
<td>OLAccel Function</td>
<td>Read OLA accel function</td>
<td></td>
</tr>
<tr>
<td>OpenCom</td>
<td>Read open communication</td>
<td></td>
</tr>
<tr>
<td>OpenCom Statement</td>
<td>Read open communication statement</td>
<td></td>
</tr>
<tr>
<td>OpenDB</td>
<td>Read open database</td>
<td></td>
</tr>
<tr>
<td>OpenNet</td>
<td>Read open network</td>
<td></td>
</tr>
<tr>
<td>OpenNet Function</td>
<td>Read open network function</td>
<td></td>
</tr>
<tr>
<td>OutReal</td>
<td>Read output real</td>
<td></td>
</tr>
<tr>
<td>OutReal Function</td>
<td>Read output real function</td>
<td></td>
</tr>
<tr>
<td>P# Statement</td>
<td>Set point</td>
<td></td>
</tr>
<tr>
<td>PalletClr</td>
<td>Clear pallet</td>
<td></td>
</tr>
<tr>
<td>PauseOn</td>
<td>Pause on</td>
<td></td>
</tr>
<tr>
<td>PDef Function</td>
<td>Set definition</td>
<td></td>
</tr>
<tr>
<td>PDef Statement</td>
<td>Set definition statement</td>
<td></td>
</tr>
<tr>
<td>PDescription</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>PDescription Function</td>
<td>Description function</td>
<td></td>
</tr>
<tr>
<td>PerformMode</td>
<td>Perform mode</td>
<td></td>
</tr>
<tr>
<td>PerformMode Function</td>
<td>Perform mode function</td>
<td></td>
</tr>
<tr>
<td>PG_FastStop</td>
<td>Fast stop</td>
<td></td>
</tr>
<tr>
<td>PG_LSpeed</td>
<td>Linear speed</td>
<td></td>
</tr>
<tr>
<td>PG_LSpeed Function</td>
<td>Linear speed function</td>
<td></td>
</tr>
<tr>
<td>PG_Scan</td>
<td>Scan</td>
<td></td>
</tr>
<tr>
<td>PG_Scan Statement</td>
<td>Scan statement</td>
<td></td>
</tr>
<tr>
<td>PLabel</td>
<td>Set label</td>
<td></td>
</tr>
<tr>
<td>PLabel$</td>
<td>Set label$</td>
<td></td>
</tr>
<tr>
<td>PlaneClr</td>
<td>Clear plane</td>
<td></td>
</tr>
<tr>
<td>PlaneDef</td>
<td>Define plane</td>
<td></td>
</tr>
<tr>
<td>Plane Statement</td>
<td>Set plane</td>
<td></td>
</tr>
<tr>
<td>Plane Function</td>
<td>Plane function</td>
<td></td>
</tr>
<tr>
<td>PListStatement</td>
<td>Set list</td>
<td></td>
</tr>
<tr>
<td>PList Statement</td>
<td>Set list statement</td>
<td></td>
</tr>
<tr>
<td>PLocal</td>
<td>Read local</td>
<td></td>
</tr>
<tr>
<td>PLocal Statement</td>
<td>Read local statement</td>
<td></td>
</tr>
<tr>
<td>SpeedFactor</td>
<td>Set speed factor</td>
<td></td>
</tr>
<tr>
<td>SpeedFactor Function</td>
<td>Set speed factor function</td>
<td></td>
</tr>
<tr>
<td>StartMain</td>
<td>Start main</td>
<td></td>
</tr>
<tr>
<td>StartMain Function</td>
<td>Start main function</td>
<td></td>
</tr>
<tr>
<td>SyncRobots</td>
<td>Sync robots</td>
<td></td>
</tr>
<tr>
<td>SyncRobots Function</td>
<td>Sync robots function</td>
<td></td>
</tr>
<tr>
<td>TaskDone</td>
<td>Task done</td>
<td></td>
</tr>
<tr>
<td>TaskDone Function</td>
<td>Task done function</td>
<td></td>
</tr>
<tr>
<td>TaskWait</td>
<td>Task wait</td>
<td></td>
</tr>
<tr>
<td>TaskWait Function</td>
<td>Task wait function</td>
<td></td>
</tr>
<tr>
<td>Tab$ Function</td>
<td>Table $</td>
<td></td>
</tr>
<tr>
<td>TargetOK</td>
<td>Target OK</td>
<td></td>
</tr>
<tr>
<td>TargetOK Function</td>
<td>Target OK function</td>
<td></td>
</tr>
<tr>
<td>TC Statement</td>
<td>TC statement</td>
<td></td>
</tr>
<tr>
<td>TCLim Statement</td>
<td>TCLim statement</td>
<td></td>
</tr>
<tr>
<td>TCLim Function</td>
<td>TCLim function</td>
<td></td>
</tr>
<tr>
<td>TeachOn</td>
<td>Teach on</td>
<td></td>
</tr>
<tr>
<td>TeachOn Function</td>
<td>Teach on function</td>
<td></td>
</tr>
<tr>
<td>TIDef Function</td>
<td>TIDef function</td>
<td></td>
</tr>
<tr>
<td>Toff Statement</td>
<td>Toff statement</td>
<td></td>
</tr>
<tr>
<td>Toff Statement</td>
<td>Toff statement</td>
<td></td>
</tr>
<tr>
<td>Toff Function</td>
<td>Toff function</td>
<td></td>
</tr>
<tr>
<td>UBounded</td>
<td>Unbound</td>
<td></td>
</tr>
<tr>
<td>UBound Function</td>
<td>Unbound function</td>
<td></td>
</tr>
<tr>
<td>ResetElapsedTime Statement</td>
<td>Reset elapsed time statement</td>
<td></td>
</tr>
<tr>
<td>ResetElapsedTime Function</td>
<td>Reset elapsed time function</td>
<td></td>
</tr>
<tr>
<td>RadToDeg</td>
<td>Rad to deg</td>
<td></td>
</tr>
<tr>
<td>ReadBin</td>
<td>Read bin</td>
<td></td>
</tr>
<tr>
<td>Read Bin Statement</td>
<td>Read bin statement</td>
<td></td>
</tr>
<tr>
<td>Read Statement</td>
<td>Read statement</td>
<td></td>
</tr>
<tr>
<td>Read Statement</td>
<td>Read statement</td>
<td></td>
</tr>
<tr>
<td>RealAccel</td>
<td>Real accel</td>
<td></td>
</tr>
<tr>
<td>RealAccel Function</td>
<td>Real accel function</td>
<td></td>
</tr>
<tr>
<td>RealPLs Function</td>
<td>Real PLs function</td>
<td></td>
</tr>
<tr>
<td>RealPos Function</td>
<td>Read position</td>
<td></td>
</tr>
<tr>
<td>RealPos Function</td>
<td>Read position function</td>
<td></td>
</tr>
<tr>
<td>RealTorque</td>
<td>Read torque</td>
<td></td>
</tr>
<tr>
<td>RealTorque Function</td>
<td>Read torque function</td>
<td></td>
</tr>
<tr>
<td>RecoverPos</td>
<td>Recover position</td>
<td></td>
</tr>
<tr>
<td>RecoverPos Function</td>
<td>Recover position function</td>
<td></td>
</tr>
<tr>
<td>Redim Statement</td>
<td>Redimension</td>
<td></td>
</tr>
<tr>
<td>Redim Statement</td>
<td>Redimension statement</td>
<td></td>
</tr>
<tr>
<td>Recover Function</td>
<td>Recover function</td>
<td></td>
</tr>
<tr>
<td>Recover Statement</td>
<td>Recover statement</td>
<td></td>
</tr>
<tr>
<td>Rnd Function</td>
<td>Round function</td>
<td></td>
</tr>
<tr>
<td>Rnd Function</td>
<td>Round function</td>
<td></td>
</tr>
<tr>
<td>RobotInfo</td>
<td>Read robot info</td>
<td></td>
</tr>
<tr>
<td>RobotInfo Function</td>
<td>Read robot info function</td>
<td></td>
</tr>
<tr>
<td>RobotInfo$ Function</td>
<td>Read robot info $ function</td>
<td></td>
</tr>
<tr>
<td>RobotModel$ Function</td>
<td>Read robot model $ function</td>
<td></td>
</tr>
<tr>
<td>RobotName$ Function</td>
<td>Read robot name $ function</td>
<td></td>
</tr>
<tr>
<td>RobotSerial$ Function</td>
<td>Read robot serial $ function</td>
<td></td>
</tr>
<tr>
<td>RobotType Function</td>
<td>Read robot type</td>
<td></td>
</tr>
<tr>
<td>RobotType Function</td>
<td>Read robot type function</td>
<td></td>
</tr>
<tr>
<td>RShift64 Function</td>
<td>Right shift 64-bit</td>
<td></td>
</tr>
<tr>
<td>SafetyOn</td>
<td>Safety on</td>
<td></td>
</tr>
<tr>
<td>SafetyOn Function</td>
<td>Safety on function</td>
<td></td>
</tr>
<tr>
<td>SelectDB</td>
<td>Select DB</td>
<td></td>
</tr>
<tr>
<td>SelectDB Statement</td>
<td>Select DB statement</td>
<td></td>
</tr>
<tr>
<td>SetCom</td>
<td>Set communication</td>
<td></td>
</tr>
<tr>
<td>SetCom Statement</td>
<td>Set communication statement</td>
<td></td>
</tr>
<tr>
<td>SetInW</td>
<td>Set in W</td>
<td></td>
</tr>
<tr>
<td>SetInW Statement</td>
<td>Set in W statement</td>
<td></td>
</tr>
<tr>
<td>SetIn Statement</td>
<td>Set in statement</td>
<td></td>
</tr>
<tr>
<td>SetIn Statement</td>
<td>Set in statement</td>
<td></td>
</tr>
<tr>
<td>SetNet</td>
<td>Set net</td>
<td></td>
</tr>
<tr>
<td>SetNet Statement</td>
<td>Set net statement</td>
<td></td>
</tr>
<tr>
<td>Shutdown</td>
<td>Shutdown</td>
<td></td>
</tr>
<tr>
<td>Shutdown Function</td>
<td>Shutdown function</td>
<td></td>
</tr>
<tr>
<td>SimGet</td>
<td>Simulation get</td>
<td></td>
</tr>
<tr>
<td>SimGet Statement</td>
<td>Simulation get statement</td>
<td></td>
</tr>
<tr>
<td>SimSet</td>
<td>Simulation set</td>
<td></td>
</tr>
<tr>
<td>SimSet Statement</td>
<td>Simulation set statement</td>
<td></td>
</tr>
<tr>
<td>SingularityAngle Statement</td>
<td>Set singularity angle</td>
<td></td>
</tr>
<tr>
<td>SingularityAngle Statement</td>
<td>Set singularity angle statement</td>
<td></td>
</tr>
<tr>
<td>SingularityDist Statement</td>
<td>Set singularity distance</td>
<td></td>
</tr>
<tr>
<td>SingularityDist Statement</td>
<td>Set singularity distance statement</td>
<td></td>
</tr>
<tr>
<td>SingularitySpeed Statement</td>
<td>Set singularity speed</td>
<td></td>
</tr>
<tr>
<td>SingularitySpeed Statement</td>
<td>Set singularity speed statement</td>
<td></td>
</tr>
<tr>
<td>SingularitySpeed Function</td>
<td>Set singularity speed function</td>
<td></td>
</tr>
<tr>
<td>VDefArm</td>
<td>VDef arm</td>
<td></td>
</tr>
<tr>
<td>VDefArm Statement</td>
<td>VDef arm statement</td>
<td></td>
</tr>
<tr>
<td>VDefLocal</td>
<td>VDef local</td>
<td></td>
</tr>
<tr>
<td>VDefLocal Statement</td>
<td>VDef local statement</td>
<td></td>
</tr>
<tr>
<td>VDefSetMotionRange Statement</td>
<td>VDef set motion range statement</td>
<td></td>
</tr>
<tr>
<td>VDefSetMotionRange Statement</td>
<td>VDef set motion range statement</td>
<td></td>
</tr>
<tr>
<td>VDefTool</td>
<td>VDef tool</td>
<td></td>
</tr>
<tr>
<td>VDefTool Statement</td>
<td>VDef tool statement</td>
<td></td>
</tr>
<tr>
<td>VGCenter</td>
<td>VG center</td>
<td></td>
</tr>
<tr>
<td>VGCenter Statement</td>
<td>VG center statement</td>
<td></td>
</tr>
<tr>
<td>VXCalib</td>
<td>VX calibrate</td>
<td></td>
</tr>
<tr>
<td>VXCalib Statement</td>
<td>VX calibrate statement</td>
<td></td>
</tr>
<tr>
<td>VXCalDelete</td>
<td>VX cal delete</td>
<td></td>
</tr>
<tr>
<td>VXCalDelete Statement</td>
<td>VX cal delete statement</td>
<td></td>
</tr>
<tr>
<td>VXCalLoad</td>
<td>VX cal load</td>
<td></td>
</tr>
<tr>
<td>VXCalLoad Statement</td>
<td>VX cal load statement</td>
<td></td>
</tr>
<tr>
<td>VXCalInfo</td>
<td>VX cal info</td>
<td></td>
</tr>
<tr>
<td>VXCalInfo Statement</td>
<td>VX cal info statement</td>
<td></td>
</tr>
<tr>
<td>VXCalSave</td>
<td>VX cal save</td>
<td></td>
</tr>
<tr>
<td>VXCalSave Statement</td>
<td>VX cal save statement</td>
<td></td>
</tr>
<tr>
<td>VXTrans</td>
<td>VX trans</td>
<td></td>
</tr>
<tr>
<td>VXTrans Statement</td>
<td>VX trans statement</td>
<td></td>
</tr>
</tbody>
</table>
WaitNet Statement
WaitPos Statement
Where Statement
WindosStatus Function
WriteBin Statement
Write Statement
WorkQue_Add Statement
WorkQue_AutoRemove Statement
WorkQue_AutoRemove Function
WorkQue_Get Function
WorkQue_Len Function
WorkQue_List Statement
WorkQue_Reject Statement
WorkQue_Reject Function
WorkQue_Remove Statement
WorkQue_Sort Statement
WorkQue_Sort Function
WorkQue_UserData Statement
WorkQue_UserData Function

XYLimClr Statement
XYLimDef Statement
XY Function
## C-2: List of Commands Added for Each Version of EPSON RC+ 7.0

<table>
<thead>
<tr>
<th>Version of EPSON RC+7.0</th>
<th>New Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common with EPSON RC+ 6.0, 5.0, and 4.0</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Ver.7.4.3</strong></td>
<td>AIO_TrackingSet</td>
</tr>
<tr>
<td></td>
<td>AIO_TrackingStart</td>
</tr>
<tr>
<td><strong>Ver.7.4.1</strong></td>
<td>AutoOrientationFlag</td>
</tr>
<tr>
<td></td>
<td>AutoOrientationFlag Function</td>
</tr>
<tr>
<td><strong>Ver.7.3.4</strong></td>
<td>SimGet</td>
</tr>
<tr>
<td></td>
<td>SimSet</td>
</tr>
<tr>
<td><strong>Ver.7.3.3</strong></td>
<td>HealthCtrlWarningEnable Statement</td>
</tr>
<tr>
<td></td>
<td>HealthCtrlWarningEnable Function</td>
</tr>
<tr>
<td><strong>Ver.7.3.2</strong></td>
<td>PDescription Statement</td>
</tr>
<tr>
<td><strong>Ver.7.3.1</strong></td>
<td>AIO_Out Statement</td>
</tr>
<tr>
<td></td>
<td>AIO_Out Function</td>
</tr>
<tr>
<td></td>
<td>AIO_OutW Statement</td>
</tr>
<tr>
<td></td>
<td>AIO_OutW Function</td>
</tr>
<tr>
<td></td>
<td>AIO_Set Statement</td>
</tr>
<tr>
<td><strong>Ver.7.3.0</strong></td>
<td>VDefArm Statement</td>
</tr>
<tr>
<td></td>
<td>VDefGetMotionRange Statement</td>
</tr>
<tr>
<td></td>
<td>VDefLocal Statement</td>
</tr>
<tr>
<td><strong>Ver.7.2.0</strong></td>
<td>CP_Offset Function</td>
</tr>
<tr>
<td></td>
<td>CP_Offset Statement</td>
</tr>
<tr>
<td></td>
<td>HealthCtrlAlarmOn Statement</td>
</tr>
<tr>
<td></td>
<td>HealthCtrlInfo Function</td>
</tr>
<tr>
<td></td>
<td>HealthCtrlInfo Statement</td>
</tr>
<tr>
<td></td>
<td>HealthCtrlRateOffset Function</td>
</tr>
<tr>
<td></td>
<td>HealthCtrlReset Function</td>
</tr>
<tr>
<td></td>
<td>HealthRateCtrlInfo Statement</td>
</tr>
<tr>
<td></td>
<td>HealthRateRBInfo Statement</td>
</tr>
<tr>
<td></td>
<td>HealthRBAlarmOn Statement</td>
</tr>
<tr>
<td></td>
<td>HealthRBAnalysis Function</td>
</tr>
<tr>
<td></td>
<td>HealthRBAnalysis Statement</td>
</tr>
<tr>
<td></td>
<td>HealthRBDistance Function</td>
</tr>
<tr>
<td></td>
<td>HealthRBDistance Statement</td>
</tr>
<tr>
<td></td>
<td>HealthRBInfo Function</td>
</tr>
<tr>
<td></td>
<td>HealthRBInfo Statement</td>
</tr>
<tr>
<td></td>
<td>HealthRBRateOffset Function</td>
</tr>
<tr>
<td><strong>Ver.7.1.4</strong></td>
<td>CollisionDetectStatement Function</td>
</tr>
<tr>
<td><strong>Ver.7.1.3</strong></td>
<td>CollisionDetect Function</td>
</tr>
<tr>
<td></td>
<td>MHourStatement Function</td>
</tr>
<tr>
<td><strong>Ver.7.1.2</strong></td>
<td>SingularityDist Function</td>
</tr>
<tr>
<td></td>
<td>SingularityDist Statement</td>
</tr>
<tr>
<td><strong>Ver.7.1.0</strong></td>
<td>BClr64 Statement</td>
</tr>
<tr>
<td></td>
<td>BSet64 Statement</td>
</tr>
<tr>
<td></td>
<td>BTst64 Statement</td>
</tr>
<tr>
<td></td>
<td>FineDist Function</td>
</tr>
<tr>
<td></td>
<td>FineDist Statement</td>
</tr>
<tr>
<td></td>
<td>FineStatus Statement</td>
</tr>
<tr>
<td></td>
<td>Fmtstr Function</td>
</tr>
<tr>
<td></td>
<td>IDefStatement</td>
</tr>
<tr>
<td></td>
<td>LShift64 Statement</td>
</tr>
<tr>
<td></td>
<td>RealAccel Statement</td>
</tr>
<tr>
<td></td>
<td>RShift64 Statement</td>
</tr>
<tr>
<td></td>
<td>WorkQue.Add Function</td>
</tr>
<tr>
<td>Version</td>
<td>Function</td>
</tr>
<tr>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>Ver.7.0.3</td>
<td>PerformMode</td>
</tr>
</tbody>
</table>

**For EPSON RC+7.0 Ver.7.0.0, new commands are different from EPSON RC+ 6.0 and EPSON RC+ 5.0.**

**Added commands for EPSON RC+ 6.0, 5.0.**

<table>
<thead>
<tr>
<th>Version of EPSON RC+7.0</th>
<th>EPSON RC+ 6.0</th>
<th>EPSON RC+ 5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver.7.0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AutoLJM Function</td>
<td>AutoLJM Function</td>
<td>AutoLJM Function</td>
</tr>
<tr>
<td>AutoLJM Statement</td>
<td>AutoLJM Statement</td>
<td>AutoLJM Statement</td>
</tr>
<tr>
<td>AvoidSingularity Function</td>
<td>AvoidSingularity Function</td>
<td>AvoidSingularity Statement</td>
</tr>
<tr>
<td>AvoidSingularity Statement</td>
<td></td>
<td>ChDisk Function</td>
</tr>
<tr>
<td>Cnv_Accel Function</td>
<td>Cnv_Accel Function</td>
<td>Cnv_Accel Function</td>
</tr>
<tr>
<td>Cnv_Accel Statement</td>
<td>Cnv_Accel Statement</td>
<td>Cnv_Accel Statement</td>
</tr>
<tr>
<td>Cnv_DownStream Function</td>
<td>Cnv_DownStream Function</td>
<td>Cnv_DownStream Function</td>
</tr>
<tr>
<td>Cnv_Mode Function</td>
<td>Cnv_Mode Function</td>
<td>Cnv_Mode Function</td>
</tr>
<tr>
<td>Cnv_Mode Statement</td>
<td>Cnv_Mode Statement</td>
<td>Cnv_Mode Statement</td>
</tr>
<tr>
<td>Cnv_Upstream Function</td>
<td>Cnv_Upstream Function</td>
<td>Cnv_Upstream Function</td>
</tr>
<tr>
<td>DeleteDB Function</td>
<td>DeleteDB Function</td>
<td></td>
</tr>
<tr>
<td>elapsedTime Statement</td>
<td>Erbb Statement</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Flush Function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GetRobotInsideBox Statement</td>
<td>GetRobotInsidePlane Statement</td>
</tr>
<tr>
<td></td>
<td>J1Angle Function</td>
<td>J1Angle Statement</td>
</tr>
<tr>
<td></td>
<td>J1Angle Statement</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LimZMargin Function</td>
<td>LimZMargin Function</td>
</tr>
<tr>
<td></td>
<td>LimZMargin Statement</td>
<td>LimZMargin Statement</td>
</tr>
<tr>
<td></td>
<td>limitTorque Function</td>
<td>limitTorque Function</td>
</tr>
<tr>
<td></td>
<td>limitTorqueStatement</td>
<td>limitTorqueFunction</td>
</tr>
<tr>
<td></td>
<td>OpenDB Function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PalletClr Function</td>
<td>PalletClr Function</td>
</tr>
<tr>
<td></td>
<td>PG_FastStop Function</td>
<td>PG_FastStop Function</td>
</tr>
<tr>
<td></td>
<td>PG_LSpeed Function</td>
<td>PG_LSpeed Function</td>
</tr>
<tr>
<td></td>
<td>PG_LSpeed Statement</td>
<td>PG_LSpeed Statement</td>
</tr>
<tr>
<td></td>
<td>PG_SlowStop Function</td>
<td>PG_SlowStop Function</td>
</tr>
<tr>
<td></td>
<td>qpDeceler Function</td>
<td>qpDeceler Function</td>
</tr>
<tr>
<td></td>
<td>qpDeceler Statement</td>
<td>qpDeceler Statement</td>
</tr>
<tr>
<td></td>
<td>qpDecelerLS Function</td>
<td>qpDecelerLS Statement</td>
</tr>
<tr>
<td></td>
<td>RecoverPos Statement</td>
<td>RecoverPos Statement</td>
</tr>
<tr>
<td></td>
<td>Recover Function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SelectDB Function</td>
<td>Shutdown Statement</td>
</tr>
</tbody>
</table>
### Appendix C: Commands of EPSON RC+7.0

<table>
<thead>
<tr>
<th>Version of EPSON RC+7.0</th>
<th>EPSON RC+ 6.0</th>
<th>EPSON RC+ 5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver.7.0.0</td>
<td>SingularityAngle Function</td>
<td>SingularityAngle Function</td>
</tr>
<tr>
<td></td>
<td>SingularityAngle Statement</td>
<td>SingularityAngle Statement</td>
</tr>
<tr>
<td></td>
<td>Singularity/Speed Function</td>
<td>Singularity/Speed Function</td>
</tr>
<tr>
<td></td>
<td>Singularity/Speed Statement</td>
<td>Singularity/Speed Statement</td>
</tr>
<tr>
<td></td>
<td>SpeedFactor Function</td>
<td>SpeedFactor Function</td>
</tr>
<tr>
<td></td>
<td>SpeedFactor Statement</td>
<td>SpeedFactor Statement</td>
</tr>
<tr>
<td></td>
<td>UpdateDB Function</td>
<td>StartMain Function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SyncRobots Function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SyncRobots Statement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TeachOnStatement Function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WindosStatus Statement</td>
</tr>
</tbody>
</table>

**Added commands for EPSON RC+ 4.0.**

<table>
<thead>
<tr>
<th>Version of EPSON RC+7.0</th>
<th>EPSON RC+ 4.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver.7.0.0</td>
<td>AbortMotion Function</td>
</tr>
<tr>
<td></td>
<td>AccelMax Statement</td>
</tr>
<tr>
<td></td>
<td>AglToPls Statement</td>
</tr>
<tr>
<td></td>
<td>Align Statement</td>
</tr>
<tr>
<td></td>
<td>AlignECP Statement</td>
</tr>
<tr>
<td></td>
<td>ArmDef Statement</td>
</tr>
<tr>
<td></td>
<td>ATCLR Function</td>
</tr>
<tr>
<td></td>
<td>AtHome Statement</td>
</tr>
<tr>
<td></td>
<td>ATRQ Function</td>
</tr>
<tr>
<td></td>
<td>ATRQ Statement</td>
</tr>
<tr>
<td></td>
<td>AutoLJM Function</td>
</tr>
<tr>
<td></td>
<td>AutoLJM Statement</td>
</tr>
<tr>
<td></td>
<td>AvoidSingularity Function</td>
</tr>
<tr>
<td></td>
<td>AvoidSingularity Statement</td>
</tr>
<tr>
<td></td>
<td>BClr Statement</td>
</tr>
<tr>
<td></td>
<td>Box Function</td>
</tr>
<tr>
<td></td>
<td>Box Statement</td>
</tr>
<tr>
<td></td>
<td>BoxClr Statement</td>
</tr>
<tr>
<td></td>
<td>BoxDef Statement</td>
</tr>
<tr>
<td></td>
<td>Brake Statement</td>
</tr>
<tr>
<td></td>
<td>BSet Statement</td>
</tr>
<tr>
<td></td>
<td>BTst Statement</td>
</tr>
<tr>
<td></td>
<td>ChDisk Function</td>
</tr>
<tr>
<td></td>
<td>ChkCom Statement</td>
</tr>
<tr>
<td></td>
<td>ChkNet Statement</td>
</tr>
<tr>
<td></td>
<td>CloseCom Function</td>
</tr>
<tr>
<td></td>
<td>CloseDB Function</td>
</tr>
<tr>
<td></td>
<td>CloseNet Function</td>
</tr>
<tr>
<td></td>
<td>Cls Function</td>
</tr>
<tr>
<td></td>
<td>CP Function</td>
</tr>
<tr>
<td></td>
<td>CP Statement</td>
</tr>
<tr>
<td></td>
<td>CR Function</td>
</tr>
<tr>
<td></td>
<td>CR Statement</td>
</tr>
<tr>
<td></td>
<td>CS Function</td>
</tr>
<tr>
<td></td>
<td>CS Statement</td>
</tr>
<tr>
<td></td>
<td>CT Function</td>
</tr>
<tr>
<td></td>
<td>CT Statement</td>
</tr>
<tr>
<td></td>
<td>CtrlDev Statement</td>
</tr>
<tr>
<td></td>
<td>Curve Function</td>
</tr>
<tr>
<td></td>
<td>CVMove Function</td>
</tr>
<tr>
<td></td>
<td>Cnv_Accel Function</td>
</tr>
<tr>
<td></td>
<td>Cnv_Accel Statement</td>
</tr>
<tr>
<td></td>
<td>Cnv_DownStream Function</td>
</tr>
<tr>
<td></td>
<td>Cnv_Mode Function</td>
</tr>
<tr>
<td></td>
<td>Cnv_Mode Statement</td>
</tr>
<tr>
<td></td>
<td>Cnv_OffsetAngle Function</td>
</tr>
<tr>
<td></td>
<td>Cnv_OffsetAngle Statement</td>
</tr>
<tr>
<td></td>
<td>Cnv_Upstream Function</td>
</tr>
<tr>
<td></td>
<td>DegToRad Statement</td>
</tr>
<tr>
<td></td>
<td>DeleteDB Function</td>
</tr>
<tr>
<td></td>
<td>DispDev Function</td>
</tr>
<tr>
<td></td>
<td>DispDev Statement</td>
</tr>
<tr>
<td></td>
<td>Dist Statement</td>
</tr>
<tr>
<td></td>
<td>EcpDef Statement</td>
</tr>
<tr>
<td></td>
<td>EResume Function</td>
</tr>
<tr>
<td></td>
<td>Errb Statement</td>
</tr>
<tr>
<td></td>
<td>ErrorOn Statement</td>
</tr>
<tr>
<td></td>
<td>Error Function</td>
</tr>
<tr>
<td></td>
<td>EStopOn Statement</td>
</tr>
<tr>
<td></td>
<td>Exit Function</td>
</tr>
<tr>
<td>Version of EPSON RC+7.0</td>
<td>EPSON RC+ 4.0</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Ver.7.0.0</td>
<td></td>
</tr>
<tr>
<td>FindPos Statement</td>
<td>P# Function</td>
</tr>
<tr>
<td>Find Function</td>
<td>PalletClr Function</td>
</tr>
<tr>
<td>FineStatus Statement</td>
<td>PauseOn Statement</td>
</tr>
<tr>
<td>Fix Statement</td>
<td>PDef Statement</td>
</tr>
<tr>
<td>Flush Function</td>
<td>PDef Function</td>
</tr>
<tr>
<td>GetRobotInsideBox</td>
<td>PG_FastStop Function</td>
</tr>
<tr>
<td>GetRobotInsidePlane</td>
<td>PG_LSspeed Function</td>
</tr>
<tr>
<td>FineStatus Statement</td>
<td>PG_LSspeed Statement</td>
</tr>
<tr>
<td>Flush Function</td>
<td>PG_Scan Function</td>
</tr>
<tr>
<td>GetRobotInsideBox</td>
<td></td>
</tr>
<tr>
<td>GetRobotInsidePlane</td>
<td></td>
</tr>
<tr>
<td>Here Function</td>
<td>PG_SlowStop Function</td>
</tr>
<tr>
<td>Here Statement</td>
<td>PLabel Function</td>
</tr>
<tr>
<td>Hex$ Statement</td>
<td>PLabel$ Statement</td>
</tr>
<tr>
<td>HomeClr Function</td>
<td>PlaneClr Function</td>
</tr>
<tr>
<td>HomeDef Statement</td>
<td>PlaneDef Function</td>
</tr>
<tr>
<td>InReal Statement</td>
<td>Plane Function</td>
</tr>
<tr>
<td>InsideBox Statement</td>
<td>Plane Statement</td>
</tr>
<tr>
<td>InsidePlane Statement</td>
<td>PLabel Function</td>
</tr>
<tr>
<td>InStr Statement</td>
<td>PLabel Statement</td>
</tr>
<tr>
<td>IOLabel$ Statement</td>
<td>PlaneDef Statement</td>
</tr>
<tr>
<td>IONumber Statement</td>
<td>Plane Function</td>
</tr>
<tr>
<td>J1Angle Function</td>
<td>PTPBoostOK Statement</td>
</tr>
<tr>
<td>J1Angle Statement</td>
<td>PTPTime Statement</td>
</tr>
<tr>
<td>JA Statement</td>
<td>PTran Function</td>
</tr>
<tr>
<td>Joint Function</td>
<td>PTRQ Function</td>
</tr>
<tr>
<td>JTran Function</td>
<td>PTPRQ Statement</td>
</tr>
<tr>
<td>LatchEnable Function</td>
<td>QPDECELR Function</td>
</tr>
<tr>
<td>LatchState Statement</td>
<td>QPDECELR Statement</td>
</tr>
<tr>
<td>LatchPos Statement</td>
<td>QPDECELS Function</td>
</tr>
<tr>
<td>LimZMargin Function</td>
<td>QPDECELS Statement</td>
</tr>
<tr>
<td>LimZMargin Statement</td>
<td></td>
</tr>
<tr>
<td>LimitTorque Function</td>
<td>RadToDeg Statement</td>
</tr>
<tr>
<td>LimitTorque Statement</td>
<td>Randomize Function</td>
</tr>
<tr>
<td>LJM Statement</td>
<td>ReadBin Function</td>
</tr>
<tr>
<td>LocalDef Statement</td>
<td>Read Function</td>
</tr>
<tr>
<td>MemInW Statement</td>
<td>RealPls Statement</td>
</tr>
<tr>
<td>MemOutW Function</td>
<td>RealPos Statement</td>
</tr>
<tr>
<td>OLAccel Function</td>
<td>RealTorque Statement</td>
</tr>
<tr>
<td>OLAccel Statement</td>
<td>RecoverPos Statement</td>
</tr>
<tr>
<td>OpenCom Function</td>
<td>Recover Function</td>
</tr>
<tr>
<td>OpenCom Statement</td>
<td></td>
</tr>
<tr>
<td>OpenDB Function</td>
<td></td>
</tr>
<tr>
<td>OpenNet Function</td>
<td></td>
</tr>
<tr>
<td>OpenNet Statement</td>
<td></td>
</tr>
<tr>
<td>OutReal Function</td>
<td></td>
</tr>
<tr>
<td>OutReal Statement</td>
<td></td>
</tr>
<tr>
<td>Version of EPSON RC+7.0</td>
<td>EPSON RC+ 4.0</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Ver.7.0.0</td>
<td></td>
</tr>
<tr>
<td>Redim Function</td>
<td>VxCalib Function</td>
</tr>
<tr>
<td>RobotInfo Statement</td>
<td>VxCalDelete Function</td>
</tr>
<tr>
<td>RobotInfo$ Statement</td>
<td>VxCalLoad Function</td>
</tr>
<tr>
<td>RobotModel$ Statement</td>
<td>VxCalInfo Statement</td>
</tr>
<tr>
<td>RobotName$ Statement</td>
<td>VxCalSave Function</td>
</tr>
<tr>
<td>RobotSerial$ Statement</td>
<td>VxTrans Statement</td>
</tr>
<tr>
<td>RobotType Statement</td>
<td></td>
</tr>
<tr>
<td>SafetyOn Statement</td>
<td>WaitNet Function</td>
</tr>
<tr>
<td>SelectDB Function</td>
<td>WaitPos Function</td>
</tr>
<tr>
<td>SetCom Function</td>
<td>Where Function</td>
</tr>
<tr>
<td>SetInW Function</td>
<td>WindosStatus Statement</td>
</tr>
<tr>
<td>SetIn Function</td>
<td>WriteBin Function</td>
</tr>
<tr>
<td>SetNet Function</td>
<td>Write Function</td>
</tr>
<tr>
<td>SetSw Function</td>
<td></td>
</tr>
<tr>
<td>Shutdown Statement</td>
<td>XYLimClr Function</td>
</tr>
<tr>
<td>SingularityAngle Function</td>
<td>XYLimDef Function</td>
</tr>
<tr>
<td>SingularityAngle Statement</td>
<td>XY Statement</td>
</tr>
<tr>
<td>SingularitySpeed Function</td>
<td></td>
</tr>
<tr>
<td>SingularitySpeed Statement</td>
<td></td>
</tr>
<tr>
<td>SoftCP Function</td>
<td></td>
</tr>
<tr>
<td>SoftCP Statement</td>
<td></td>
</tr>
<tr>
<td>SpeedFactor Function</td>
<td></td>
</tr>
<tr>
<td>SpeedFactor Statement</td>
<td></td>
</tr>
<tr>
<td>StartMain Function</td>
<td></td>
</tr>
<tr>
<td>SyncRobots Function</td>
<td></td>
</tr>
<tr>
<td>SyncRobots Statement</td>
<td></td>
</tr>
<tr>
<td>SysErr Statement</td>
<td></td>
</tr>
<tr>
<td>Tab$ Function</td>
<td></td>
</tr>
<tr>
<td>TargetOK Statement</td>
<td></td>
</tr>
<tr>
<td>TaskDone Statement</td>
<td></td>
</tr>
<tr>
<td>TaskInfo Statement</td>
<td></td>
</tr>
<tr>
<td>TaskInfo$ Statement</td>
<td></td>
</tr>
<tr>
<td>TaskState Function</td>
<td></td>
</tr>
<tr>
<td>TaskState Statement</td>
<td></td>
</tr>
<tr>
<td>TaskWait Function</td>
<td></td>
</tr>
<tr>
<td>TC Function</td>
<td></td>
</tr>
<tr>
<td>TCLim Function</td>
<td></td>
</tr>
<tr>
<td>TCLim Statement</td>
<td></td>
</tr>
<tr>
<td>TCSpeed Function</td>
<td></td>
</tr>
<tr>
<td>TCSpeed Statement</td>
<td></td>
</tr>
<tr>
<td>TeachOn Statement</td>
<td></td>
</tr>
<tr>
<td>TillOn Statement</td>
<td></td>
</tr>
<tr>
<td>TIDef Statement</td>
<td></td>
</tr>
<tr>
<td>Toff Function</td>
<td></td>
</tr>
<tr>
<td>Ton Function</td>
<td></td>
</tr>
<tr>
<td>UBound Statement</td>
<td></td>
</tr>
<tr>
<td>UpdateDB Function</td>
<td></td>
</tr>
</tbody>
</table>
## C-3: Deletion Commands (Sort by Version)

Deletion commands of EPSON RC+ 6.0, 5.0, and 4.0.

<table>
<thead>
<tr>
<th>Version of EPSON RC+7.0</th>
<th>EPSON RC+ 6.0</th>
<th>EPSON RC+ 5.0</th>
<th>EPSON RC+ 4.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver.7.1.2</td>
<td>SetLCD Function</td>
<td>SetLCD Function</td>
<td>SetLCD Function</td>
</tr>
<tr>
<td>Ver.7.0.0</td>
<td>Dir Function</td>
<td>-</td>
<td>Dir Function</td>
</tr>
<tr>
<td></td>
<td>Type Function</td>
<td></td>
<td>Type Function</td>
</tr>
</tbody>
</table>