# General Contents

## Chapter 1
**OVERVIEW** .................................................................................................................. 1-1
1-1 Features of the ERCX Series Controller ........................................................................ 1-2
1-2 Setting Up for Operation ............................................................................................... 1-3
1-3 External View and Part Names ...................................................................................... 1-4
  1-3-1 ERCX controller ..................................................................................................... 1-4
  1-3-2 TPB ....................................................................................................................... 1-6
1-4 System Configuration .................................................................................................. 1-7
  1-4-1 System configuration ............................................................................................. 1-7
1-5 Accessories and Options ............................................................................................. 1-8
  1-5-1 Accessories ......................................................................................................... 1-8
  1-5-2 Peripheral options ............................................................................................... 1-8

## Chapter 2
**INSTALLATION AND CONNECTION** .................................................................................. 2-1
2-1 Installing the ERCX Controller ...................................................................................... 2-2
  2-1-1 Installation method ............................................................................................... 2-2
2-2 Connecting the Power Supply ...................................................................................... 2-3
  2-2-1 Power supply ....................................................................................................... 2-3
  2-2-2 Connecting the power supply ............................................................................... 2-4
  2-2-3 Insulation resistance and voltage breakdown tests ................................................. 2-4
2-3 Grounding .................................................................................................................... 2-5
2-4 Connecting the ERCX to the Control Unit .................................................................... 2-5
2-5 Connecting to the Robot .............................................................................................. 2-6
  2-5-1 Robot I/O connector and signal table .................................................................... 2-6
2-6 Connecting to the I/O Connector .................................................................................. 2-7
2-7 Connecting the Absolute Battery ................................................................................ 2-8

## Chapter 3
**I/O INTERFACE** ............................................................................................................... 3-1
3-1 I/O Signals .................................................................................................................. 3-2
3-2 Input Signal Description .............................................................................................. 3-3
  3-2-1 Dedicated command input .................................................................................. 3-3
  3-2-2 General-purpose input (DI0 to DI15) .................................................................. 3-6
  3-2-3 SERVICE mode input (SVCE) .......................................................................... 3-7
  3-2-4 Interlock (LOCK) .............................................................................................. 3-7
  3-2-5 Emergency stop inputs 1, 2 (EMG1, EMG2) ...................................................... 3-7
3-3 Output Signal Description ........................................................................................... 3-8
  3-3-1 Dedicated output ................................................................................................ 3-8
  3-3-2 General-purpose output (DO0 to DO12) .............................................................. 3-9
3-4 I/O Circuits .................................................................................................................. 3-9
  3-4-1 I/O circuit specifications ..................................................................................... 3-9
  3-4-2 I/O circuit and connection example .................................................................... 3-10
3-5 I/O Connection Diagram ............................................................................................ 3-12
  3-5-1 General connections .......................................................................................... 3-12
  3-5-2 Connection to PLC output unit .......................................................................... 3-13
  3-5-3 Connection to PLC input unit ............................................................................ 3-14
3-6 I/O Control Timing Timing Charts ............................................................................ 3-15
  3-6-1 When turning the power on .............................................................................. 3-15
  3-6-2 When executing a dedicated input command ...................................................... 3-16
  3-6-3 When interlock signal is input .......................................................................... 3-20
  3-6-4 When emergency stop is input ............................................................................ 3-21
  3-6-5 When alarm is issued ....................................................................................... 3-21
  3-6-6 When executing a point movement command ................................................... 3-22
## Chapter 4 BASIC OPERATION OF THE TPB ...................................................... 4-1

4-1 Connecting and Disconnecting the TPB .................................................. 4-2
   4-1-1 Connecting the TPB to the ERCX controller ................................. 4-2
   4-1-2 Disconnecting the TPB from the ERCX controller ........................ 4-3
   4-1-3 Different points from SRCX and DRCX controllers .................... 4-3

4-2 Basic Key Operation ............................................................................. 4-4

4-3 Reading the Screen ............................................................................. 4-5
   4-3-1 Program execution screen ........................................................... 4-5
   4-3-2 Program edit screen (teaching playback) ..................................... 4-5
   4-3-3 Point edit screen ........................................................................ 4-5
   4-3-4 DIO monitor screen ................................................................. 4-6

4-4 Hierarchical Menu Structure ............................................................. 4-7

4-5 Restricting Key Operation by Access Level ........................................ 4-8
   4-5-1 Explanation of access level ......................................................... 4-8
   4-5-2 Changing an access level ......................................................... 4-9

## Chapter 5 PARAMETERS ............................................................................. 5-1

5-1 Setting the Parameters ......................................................................... 5-2

5-2 Parameter Description ......................................................................... 5-3

## Chapter 6 PROGRAMMING ...................................................................... 6-1

6-1 Basic Contents .................................................................................... 6-2
   6-1-1 Robot language and point data ..................................................... 6-2
   6-1-2 Using the TPB to enter the robot language ................................. 6-2
   6-1-3 Program specifications ............................................................... 6-2

6-2 Editing Programs ................................................................................ 6-3
   6-2-1 Creating programs after initialization ....................................... 6-4
   6-2-2 Creating a new program ............................................................ 6-6
   6-2-3 Adding a step ............................................................................. 6-7
   6-2-4 Correcting a step ....................................................................... 6-9
   6-2-5 Inserting a step ......................................................................... 6-10
   6-2-6 Deleting a step .......................................................................... 6-11

6-3 Program Utility .................................................................................. 6-12
   6-3-1 Copying a program ................................................................. 6-12
   6-3-2 Deleting a program ............................................................... 6-13
   6-3-3 Viewing the program information ............................................. 6-14

## Chapter 7 EDITING POINT DATA ............................................................. 7-1

7-1 Manual Data Input ............................................................................ 7-2

7-2 Teaching Playback ............................................................................. 7-3

7-3 Direct Teaching ................................................................................. 7-5

7-4 Manual Control of General-Purpose Output ..................................... 7-7

7-5 Manual Release of Holding Brake ..................................................... 7-8

7-6 Deleting Point Data ........................................................................ 7-9

7-7 Tracing Points (Moving to a registered data point) ......................... 7-10

## Chapter 8 ROBOT LANGUAGE ............................................................... 8-1

8-1 Robot Language Table ....................................................................... 8-2

8-2 Robot Language Syntax Rules .......................................................... 8-3
   8-2-1 Command statement format .................................................... 8-3
   8-2-2 Variables ................................................................................. 8-4

8-3 Program Function ............................................................................. 8-5
   8-3-1 Multi-task function ............................................................... 8-5
   8-3-2 Limitless movement function .............................................. 8-6
Chapter 9  OPERATING THE ROBOT ................................................................. 9-1

9-1 Performing Return-to-Origin ................................................................. 9-2
  9-1-1  Return-to-origin by the search method ............................................ 9-2
  9-1-2  Return-to-origin by the mark method .............................................. 9-4

9-2 Using Step Operation............................................................................. 9-6

9-3 Using Automatic Operation ................................................................. 9-9

9-4 Switching the Execution Program ....................................................... 9-11

9-5 Emergency Stop Function ................................................................. 9-12
  9-5-1  Initiating an emergency stop ......................................................... 9-12
  9-5-2  Recovering from an emergency stop ............................................. 9-12

9-6 Displaying the Memory I/O Status ...................................................... 9-14

9-7 Displaying the Variables ..................................................................... 9-15
# Chapter 10 OTHER OPERATIONS

10-1 Initialization ................................................................. 10-2
10-2 DIO Monitor Display .................................................. 10-4
   10-2-1 Display from the monitor menu ............................. 10-4
   10-2-2 Display from the DIO key operation ....................... 10-5
10-3 System Information Display ........................................ 10-5
10-4 SERVICE mode function ............................................. 10-6
   10-4-1 Safety settings for SERVICE mode ......................... 10-7
   10-4-2 Enabling/disabling the SERVICE mode function ........ 10-9
   10-4-3 Setting the SERVICE mode functions ...................... 10-11
10-5 System utilities .......................................................... 10-13
   10-5-1 Viewing hidden parameters ................................... 10-13
10-6 Using a Memory Card .................................................. 10-14
   10-6-1 Saving controller data to a memory card ................ 10-14
   10-6-2 Loading data from a memory card ......................... 10-16
   10-6-3 Formatting a memory card ................................. 10-18
   10-6-4 Viewing the ID number for memory card data ........... 10-19
10-7 Duty (load factor) monitor ........................................... 10-20
   10-7-1 Measuring the duty (load factor) ............................. 10-22

# Chapter 11 COMMUNICATION WITH PC

11-1 Communication Parameter Specifications .................... 11-2
11-2 Communication Cable Specifications ............................ 11-3
   11-2-1 Connecting to the computer with a 25-pin D-sub connector 11-3
   11-2-2 Connecting to the computer with a 9-pin D-sub connector 11-3
11-3 Communication Command Specifications ...................... 11-4
11-4 Communication Command List .................................... 11-5
11-5 Communication Command Description ......................... 11-8
   11-5-1 Robot movements ................................................ 11-8
   11-5-2 Data handling .................................................. 11-17
   11-5-3 Utilities .......................................................... 11-29

# Chapter 12 MESSAGE TABLES

12-1 Error Messages .......................................................... 12-2
   12-1-1 Error message specifications .............................. 12-2
   12-1-2 Command error message ...................................... 12-2
   12-1-3 Operation error message .................................... 12-3
   12-1-4 Program error message ...................................... 12-4
   12-1-5 System error message ....................................... 12-5
   12-1-6 Multi-task error message ................................... 12-5
12-2 TPB Error Messages ................................................... 12-6
12-3 Stop Messages .......................................................... 12-7
   12-3-1 Message specifications ...................................... 12-7
   12-3-2 Stop messages ................................................ 12-7
12-4 Displaying the Error History ...................................... 12-8

# Chapter 13 TROUBLESHOOTING

13-1 If A Trouble Occurs .................................................. 13-2
13-2 Alarm and Countermeasures ....................................... 13-3
   13-2-1 Alarm specifications ......................................... 13-3
   13-2-2 Alarm message list .......................................... 13-4
13-3 Troubleshooting for Specific Symptom ......................... 13-7
   13-3-1 Relating to the robot movement ......................... 13-7
   13-3-2 Relating to the I/O .......................................... 13-9
   13-3-3 Other ......................................................... 13-10
13-4 Displaying the Alarm History .................................... 13-11
Thank you for purchasing the YAMAHA single-axis robot controller ERCX series (hereafter called "ERCX controller" or simply "ERCX" or "this controller"). This manual describes ERCX controller features and operating procedures.

When used with a YAMAHA single-axis FLIP-X series robot, the ERCX controller performs positioning and pick-and-place tasks of various mechanical parts and devices.

This first chapter explains basic information you should know before using the ERCX controller such as names and functions of the various parts, steps necessary to prepare the robot for operation, and the architecture of the system itself. Please read this chapter carefully for a basic overview of the ERCX controller.
1-1 Features of the ERCX Series Controller

The ERCX series is a high-performance robot controller using a 32-bit RISC chip CPU. When used with a YAMAHA single-axis FLIP-X series robot, the ERCX controller performs positioning tasks of various mechanical parts and devices. The ERCX controller also performs I/O control of solenoid valves and sensors, and controls communication with a PC (personal computer). Using only one ERCX controller allows configuring a complete system for simple applications such as pick-and-place tasks.

The ERCX series has the following features:

- A high-performance 32-bit RISC chip CPU is used for high-speed, high-precision software servo control.
- Absolute method is used as a standard feature. This eliminates return-to-origin operation which has been necessary each time the power is turned on, allowing you to begin actual robot tasks immediately after power-on.
- The program assets that were created with the previous SRC, SRCA, ERC and SRCH series can be used without any modifications. The robot language and I/O control operations are the same as using the SRCX, DRCX and TRCX series. Options such as the TPB programming box, POPCOM support software, memory card and I/O checker can also be directly used as is.
- Ideal acceleration and deceleration speeds can be obtained by simply entering the number of the robot to control and the payload parameter. No troublesome servo adjustments are required.
- The I/O interface provides 16 input and 13 output points for general-purpose user wiring as a standard feature.
- The TPB programming box (option) allows interactive user operation by simple menus that permit immediate use. The robot can also be operated from a personal computer (PC) just the same as TPB when the POPCOM software (option) is installed in the PC.
- Programs for robot operation can be written with an easy-to-learn robot language that closely resembles BASIC. Even first-time users will find it easy to use.
- Users not accustomed to robot language can use a PLC (programmable logic controller) to directly move the robot by specifying the operation points.
- Users can create programs and control the robot on a personal computer (PC). Communication with the PC is performed with an easy-to-learn robot language similar to BASIC. Even first-time users will find it easy to use.
- A built-in multi-task function allows efficiently creating the programs.

NOTE

The ERCX controller can be operated from either a TPB (programming box) or a PC running with communication software such as POPCOM. This user’s manual mainly describes operations using the TPB. For details on operation with POPCOM, refer to the POPCOM manual. If you want to use your own methods to operate the ERCX controller from a PC, refer to Chapter 11 “Communications with PC” for pertinent information.
## 1-2 Setting Up for Operation

The chart below illustrates the basic steps to follow from the time of purchase of this controller until it is ready for use. The chapters of this user’s manual are organized according to the operation procedures, and allow first time users to proceed one step at a time.

### Basic steps

<table>
<thead>
<tr>
<th>Operation</th>
<th>Information to be familiar with</th>
<th>Refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installation</td>
<td>• Installing the controller</td>
<td>2-1</td>
</tr>
<tr>
<td></td>
<td>Wiring and connection</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Connecting the power supply</td>
<td>2-2</td>
</tr>
<tr>
<td></td>
<td>• Grounding</td>
<td>2-3</td>
</tr>
<tr>
<td></td>
<td>• Connecting peripheral equipment</td>
<td>2-4 to 2-7</td>
</tr>
<tr>
<td></td>
<td>• Understanding the I/O interface</td>
<td>Chapter 3</td>
</tr>
<tr>
<td>Setting parameters</td>
<td>• Understanding basic TPB operations</td>
<td>Chapter 4</td>
</tr>
<tr>
<td></td>
<td>• Setting the various parameters</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>Programming</td>
<td>• Inputting or editing programs</td>
<td>Chapter 6</td>
</tr>
<tr>
<td></td>
<td>• Editing point data</td>
<td>Chapter 7</td>
</tr>
<tr>
<td></td>
<td>• Robot language</td>
<td>Chapter 8</td>
</tr>
<tr>
<td>Running the robot</td>
<td>• Return-to-origin</td>
<td>Chapter 9</td>
</tr>
<tr>
<td></td>
<td>• Various operation steps</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Emergency stop</td>
<td></td>
</tr>
</tbody>
</table>
1-3 External View and Part Names

This section explains part names of the ERCX controller and TPB along with their functions. Note that the external view and specifications are subject to change without prior notice to the user.

1-3-1 ERCX controller

1. Status Display Lamp
   This lamp indicates the operating status of the robot and controller.
   Refer to "15-1-3 LED display" for information on controller status and the matching LED display.

2. TPB Connector
   This is used to connect the TPB or DPB programming box, or the RS-232C terminal of a PC (personal computer).

3. COM Connector
   This is used to connect a network system when the optional network card is installed. (This is covered when the option is not in use.)

4. BAT Connector
   This is the connector for the absolute battery.

5. Robot I/O Connector
   This is used for input and output from robot peripheral devices such as resolver and brake signals, and also used to connect the power line for the servo motor.

6. I/O Connector
   This is used to connect external equipment such as a PLC.

7. Power connector (24V, N, ↓)
   This is the connector for supplying DC 24V power to the ERCX controller. The ground terminal must be properly grounded to prevent electrical shock to the human body and to maintain equipment reliability.
Fig. 1-1 Exterior of the ERCX controller

Fig. 1-2 Three-side view of the ERCX controller
1-3-2 TPB

1. Liquid Crystal Display (LCD) Screen
   This display has four lines of twenty characters each and is used as a program console.

2. Memory Card Slot
   An IC memory card can be inserted here. Be careful not to insert the card upside-down.

3. Control Keys
   The TPB can be operated in interactive data entry mode. Instructions are input through the
   control keys while reading the contents on the LCD screen.

4. Connection Cable
   This cable connects the TPB to the ERCX controller.

5. DC Power Input Terminal
   Not used.

6. Emergency Stop Button
   This is the emergency stop button. When pressed, it locks in the depressed position. To
   release this button, turn it clockwise.
   To cancel emergency stop, first release this button and then use the servo recovery command
   via the I/O interface or the servo recovery operation from the TPB.
1-4 System Configuration

1-4-1 System configuration

The ERCX controller can be combined with various peripheral units and optional products to configure a robot system as shown below.

**Fig. 1-5 System configuration diagram**
1-5 Accessories and Options

1-5-1 Accessories

The ERCX controller comes with the following accessories. After unpacking, check that all items are included.

1. Power connector
   MC1.5/3-ST-3,5 made by Phoenix Contact 1 piece
2. I/O connector
   Connector : FCN-361P048-AU made by Fujitsu 1 piece
   Connector cover : FCN-360C048-E made by Fujitsu 1 piece
3. RS-232C dust cover
   XM2T-2501 made by OMRON 1 piece
4. Absolute battery unit (B1, B2)
   Ni-Cd battery : (Either of the following types is supplied according to the user's order.)
   B1 type (3.6V/700mAh) made by Sanyo Electric 1 piece
   B2 type (3.6V/2000mAh) made by Sanyo Electric 1 piece
   Cable tie : T30R made by Tyton 2 pieces
   Binding strap : A TMS-30 made by Kitagawa Industries 2 pieces

1-5-2 Peripheral options

The following options are available for the ERCX controller:

1. TPB
   This is a hand-held programming box that connects to the ERCX controller for teaching point data, editing robot programs and operating the robot. The TPB allows interactive user operation by simple menus so that even first-time users can easily operate the robot with the TPB.

2. IC memory card
   An IC memory card can be used with the TPB to back up programs, point data and parameter data.

3. POPCOM
   The POPCOM is support software that runs on a PC (personal computer) connected to the ERCX controller. The POPCOM software allows easy editing of robot programs and operation of a robot just the same as with a TPB.

4. I/O checker
   The I/O checker connects to the I/O connector and can be used as an I/O status monitor (with LED indicators) or as a simulated input device by toggle switches.
This chapter contains precautions that should be observed when installing the controller, as well as procedures and precautions for wiring the controller to the robot and to external equipment.
2-1 Installing the ERCX Controller

2-1-1 Installation method
Using the L-shaped brackets attached to the top and bottom of the controller, install the controller from the front or rear position. (See Fig.1-2 Three-side view of the ERCX controller.)

2-1-2 Installation location
- Install the controller in locations where the ambient temperature is between 0 to 40°C and the humidity is between 35 to 85% without condensation.
- Do not install the controller upside down or at an angle.
- Install the controller in locations with sufficient space (at least 20mm away from the wall or other object) for good ventilation and air flow.
- Do not install the controller in locations where corrosive gases such as sulfuric acid or hydrochloric acid gas are present, or in atmosphere containing flammable gases and liquids.
- Install the controller in locations with a minimal amount of dust.
- Avoid installing the controller in locations subject to cutting chips, oil or water from other machines.
- Avoid installing the controller in locations where electromagnetic noise or electrostatic noise is generated.
- Avoid installing the controller in locations subject to shock or large vibration.
2-2 Connecting the Power Supply

2-2-1 Power supply

■ Power requirements

<table>
<thead>
<tr>
<th>Power supply voltage</th>
<th>DC24V ±10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power supply current</td>
<td>3 to 4.5A (depends on robot type)</td>
</tr>
</tbody>
</table>

■ Power supply current

<table>
<thead>
<tr>
<th></th>
<th>T4</th>
<th>T5</th>
<th>C4</th>
<th>C5</th>
<th>YMS45</th>
<th>YMS55</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard model (horizontal use)</td>
<td>3A</td>
<td>3A</td>
<td>3A</td>
<td>3A</td>
<td>3A</td>
<td>4.5A</td>
</tr>
<tr>
<td>-BK model (equipped with brake for vertical use)</td>
<td>3A</td>
<td>3A</td>
<td>3A</td>
<td>3A</td>
<td>3A</td>
<td>4.5A</td>
</tr>
</tbody>
</table>

⚠️ CAUTION
If the current supplied to the ERCX controller is insufficient, alarm stop or abnormal operation may occur. Use caution when selecting a 24V power supply.

If the power supply voltage drops below the above range during operation, the alarm circuit will work and return the controller to the initial state the same as just after power-on, or stop operation. To avoid this problem, use a regulated power supply with voltage fluctuations of less than ±10%.

Since the controller uses a capacitor input type power supply circuit, a large inrush current flows when the power is turned on. Do not use fast-blow circuit breakers and fuses. For the same reason, avoid turning the power off and on again repeatedly in intervals of less than 10 seconds. This could harm the main circuit elements in the controller.

* The robot performance may be enhanced if you have a 24V power supply with an adequately large capacity. Please consult us for more details.

⚠️ CAUTION
The power supply mentioned above is for operating the controller. In addition to this power supply, you will need to provide power for the mechanical brake and I/O control through the I/O connector.

Power supply capacity required by I/O control:
50mA for emergency stop circuit + current capacity for I/O drive (depends on user application)

Power supply capacity required by brake control:
300mA for robots with brake
2-2-2 Connecting the power supply

Using the power plug (supplied), connect the power supply to the POWER connector of the ERCX controller. Make correct connections while referring to the figure below. Misconnections may result in serious danger such as fire. Securely screw the end of each wire onto the terminal so that it will not come loose. If the wires pick up noise and the controller becomes unstable, we recommend using a ferrite core with the power wires as shown below.

---

**Fig. 2-1 Power supply connections**

![Diagram of power supply connections]

1. 24V
2. N(0V)
3. Ground

---

**CAUTION**

The ERCX series controller does not have a power switch. Be sure to provide a power supply breaker (insulation) of the correct specifications that will turn the power on or off to the entire system including the robot controller.

**WARNING**

Before beginning the wiring work, make sure that the power supply for the entire system is turned off. Doing the wiring work while power is still turned on may cause electrical shocks.

2-2-3 Insulation resistance and voltage breakdown tests

Never attempt insulation resistance tests or voltage breakdown tests on the ERCX controller. Since capacitive grounding is provided between the controller body and 0V, these tests may mistakenly detect excess leakage current or damage the internal circuitry. If these tests are required, please consult your YAMAHA sales office or representative.
2-3  Grounding

The ERCX controller must be grounded to prevent danger to personnel from electrical shocks in case of electrical leakage and prevent equipment malfunctions due to electrical noise. We strongly recommend that Class D (grounding resistance of 100 ohms or less) or higher grounding be provided. For grounding the controller, use the ground terminal on the power supply terminal block.

* Class D grounding is the same as Class 3 grounding previously used.

2-4  Connecting the ERCX to the Control Unit

The ERCX controller can be operated either through the TPB programming box or through a PC (personal computer) equipped with an RS-232C terminal. When using the TPB, plug the TPB cable connector into the TPB connector of the ERCX controller. (Refer to "4-1-1 Connecting the TPB to the ERCX controller".) When using a PC, plug the RS-232C interface cable connector (25 pins) into the TPB connector of the ERCX controller. (Refer to "11-2 Communication Cable Specifications".) To prevent equipment malfunction due to noise, we strongly recommend that Class D (grounding resistance of 100 ohms or less) or higher grounding be provided.
2-5 Connecting to the Robot

First make sure that the power to the ERCX controller is turned off, and then connect the robot cable to the robot I/O connector on the front panel of the ERCX controller. Fully insert the robot I/O cable until it clicks in position.

* When the robot cable is disconnected from the controller, an alarm (15: FEEDBACK ERROR 2) is issued. Since the controller is shipped with the robot cable disconnected, an alarm is always issued when the controller is first turned on. But this is not an equipment problem.

2-5-1 Robot I/O connector and signal table

<table>
<thead>
<tr>
<th>Terminal No.</th>
<th>Signal name</th>
<th>Description</th>
<th>Terminal No.</th>
<th>Signal name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BK1+</td>
<td>Brake signal 1 (+)</td>
<td>9</td>
<td>BK1-</td>
<td>Brake signal 1 (-)</td>
</tr>
<tr>
<td>2</td>
<td>MW</td>
<td>Motor W-phase output</td>
<td>10</td>
<td>MV</td>
<td>Motor V-phase output</td>
</tr>
<tr>
<td>3</td>
<td>MU</td>
<td>Motor U-phase output</td>
<td>11</td>
<td>FG</td>
<td>Frame ground</td>
</tr>
<tr>
<td>4</td>
<td>DG</td>
<td>Digital ground</td>
<td>12</td>
<td>NC</td>
<td>No connection</td>
</tr>
<tr>
<td>5</td>
<td>WCK1+</td>
<td>Wire breakage detection 1 (+)</td>
<td>13</td>
<td>WCK1-</td>
<td>Wire breakage detection 1 (-)</td>
</tr>
<tr>
<td>6</td>
<td>R1+</td>
<td>Resolver excitation output 1 (+)</td>
<td>14</td>
<td>R1-</td>
<td>Resolver excitation output 1 (-)</td>
</tr>
<tr>
<td>7</td>
<td>PC1+</td>
<td>Resolver COS input 1 (+)</td>
<td>15</td>
<td>PC1-</td>
<td>Resolver COS input 1 (-)</td>
</tr>
<tr>
<td>8</td>
<td>PS1+</td>
<td>Resolver SIN input 1 (+)</td>
<td>16</td>
<td>PS1-</td>
<td>Resolver SIN input 1 (-)</td>
</tr>
</tbody>
</table>
2-6 Connecting to the I/O Connector

The I/O connector is used for connecting the ERCX controller to external equipment such as a PLC. When using external equipment for I/O control, connect the wiring to the I/O connector supplied as an accessory and then plug it into the I/O connector on the ERCX controller. The signals assigned to each of the I/O connector terminals and their functions are described in detail in Chapter 3.

The I/O connector that is compatible with the ERCX controller is listed below.

- Connector type No.: FCN-361P048-AU (Fujitsu)
- Connector cover type No.: FCN-360C48-E

**CAUTION**

Even if not using I/O control, the I/O connector should be plugged in after completing the following wiring.

1. Short pin numbers A24 (EMG1) and B24 (EMG2).
2. Short pin numbers B4 (LOCK) and A15, B15 (0V).
3. Connect pin numbers A13 and B13 (+IN, COM) to an external 24V power supply.

If step 1 is not completed, an emergency stop is triggered. If step 2 is not completed, an interlock occurs. In either case, the controller cannot be operated (see Chapter 3). Note that 24V power will not be supplied to the I/O circuit unless shorted as in 3. An alarm is issued (06:24V POWER OFF) when power is not supplied and the operation disabled.
2-7 Connecting the Absolute Battery

Connect the absolute battery to the controller as shown below. Use a cable tie and binding strap (supplied) to secure the battery to the side of the controller or at the proper position in the system. A "B1 type" or "B2 type" battery is supplied with the controller depending on your order.

**CAUTION**
Do not modify the battery wire or extend it. Modification and extended wire may cause troubles or malfunction of the robot.

- When the absolute battery is disconnected from the controller, an alarm (24: ABS. DATA ERROR) is issued. Since the controller is shipped with the absolute battery disconnected, an alarm is always issued when the controller is first turned on. But this is not an equipment problem. (An alarm "23: ABS. BAT. L-VOLTAGE" might occur in some cases.)

- When the controller is first used or is kept turned off for a period in excess of the data backup time, the battery must be recharged. The battery is automatically charged while the controller is turned on. Keep the battery charged for longer than the time listed in the table below. Since the battery charging time does not affect robot operation, the controller can be used to perform teaching, program editing and robot operation while the battery is still being charged.

<table>
<thead>
<tr>
<th>Type</th>
<th>Backup time *2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1 type</td>
<td>120h</td>
</tr>
<tr>
<td>B2 type</td>
<td>340h</td>
</tr>
</tbody>
</table>

*1) At ambient temperature of 20°C
*2) After power is off with absolute battery fully charged.

- If the absolute backup function is unnecessary, the controller can be used with the absolute battery left removed. (See "16-1 Operation When Not Using Absolute Function").
Chapter 3  I/O INTERFACE

The ERCX series has an I/O interface consisting of emergency stop inputs, interlock, 7 dedicated command inputs, 3 dedicated outputs, 16 general-purpose inputs, 13 general-purpose outputs, etc. This I/O interface allows exchanging commands and data between the ERCX series and external equipment. This I/O interface can also directly connect to and control actuators such as valves and sensors. To construct a system utilizing the features of the ERCX series, you must understand the signals assigned to each terminal on the I/O connector and how they work. This chapter 3 covers this fundamental information.

This chapter also provides examples of I/O circuit connections and timing charts for expanding the system by using a PLC or similar devices. Refer to these diagrams and examples when creating sequence programs. Terms "ON" and "OFF" used in this chapter mean "on" and "off" of switches connected to the input terminal when referring to input signals. They also mean "on" and "off" of output transistors when referring to output signals.
### 3-1 I/O Signals

The standard I/O connector of the ERCX controller has 48 pins, with an individual signal assigned to each pin. The following table shows the pin number as well as the name and description of each signal assigned to each pin. For a more detailed description of each signal, refer to "3-2 Input Signal Description" and onwards.

<table>
<thead>
<tr>
<th>No.</th>
<th>Pin No.</th>
<th>Signal name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A1</td>
<td>ABS-PT</td>
<td>Absolute point movement command</td>
</tr>
<tr>
<td>2</td>
<td>B1</td>
<td>INC-PT</td>
<td>Relative point movement command</td>
</tr>
<tr>
<td>3</td>
<td>A2</td>
<td>AUTO-R</td>
<td>Automatic operation start command</td>
</tr>
<tr>
<td>4</td>
<td>B2</td>
<td>STEP-R</td>
<td>Step operation start command</td>
</tr>
<tr>
<td>5</td>
<td>A3</td>
<td>ORG-S</td>
<td>Return-to-origin command</td>
</tr>
<tr>
<td>6</td>
<td>B3</td>
<td>RESET</td>
<td>Reset command</td>
</tr>
<tr>
<td>7</td>
<td>A4</td>
<td>SERVO</td>
<td>Servo recovery command</td>
</tr>
<tr>
<td>8</td>
<td>B4</td>
<td>LOCK</td>
<td>Interlock</td>
</tr>
<tr>
<td>9</td>
<td>A5</td>
<td>DI 0</td>
<td>General-purpose input 0</td>
</tr>
<tr>
<td>10</td>
<td>B5</td>
<td>DI 1</td>
<td>General-purpose input 1</td>
</tr>
<tr>
<td>11</td>
<td>A6</td>
<td>DI 2</td>
<td>General-purpose input 2</td>
</tr>
<tr>
<td>12</td>
<td>B6</td>
<td>DI 3</td>
<td>General-purpose input 3</td>
</tr>
<tr>
<td>13</td>
<td>A7</td>
<td>DI 4</td>
<td>General-purpose input 4</td>
</tr>
<tr>
<td>14</td>
<td>B7</td>
<td>DI 5</td>
<td>General-purpose input 5</td>
</tr>
<tr>
<td>15</td>
<td>A8</td>
<td>DI 6</td>
<td>General-purpose input 6</td>
</tr>
<tr>
<td>16</td>
<td>B8</td>
<td>DI 7</td>
<td>General-purpose input 7</td>
</tr>
<tr>
<td>17</td>
<td>A9</td>
<td>DI 8</td>
<td>General-purpose input 8</td>
</tr>
<tr>
<td>18</td>
<td>B9</td>
<td>DI 9</td>
<td>General-purpose input 9</td>
</tr>
<tr>
<td>19</td>
<td>A10</td>
<td>DI 10</td>
<td>General-purpose input 10</td>
</tr>
<tr>
<td>20</td>
<td>B10</td>
<td>DI 11</td>
<td>General-purpose input 11</td>
</tr>
<tr>
<td>21</td>
<td>A11</td>
<td>DI 12</td>
<td>General-purpose input 12</td>
</tr>
<tr>
<td>22</td>
<td>B11</td>
<td>DI 13</td>
<td>General-purpose input 13</td>
</tr>
<tr>
<td>23</td>
<td>A12</td>
<td>DI 14</td>
<td>General-purpose input 14</td>
</tr>
<tr>
<td>24</td>
<td>B12</td>
<td>DI 15/SVCE</td>
<td>General-purpose input 15/SERVICE mode input</td>
</tr>
<tr>
<td>25</td>
<td>A13</td>
<td>+IN COM</td>
<td>External +24V power supply input for controller</td>
</tr>
<tr>
<td>26</td>
<td>B13</td>
<td>+IN COM</td>
<td>External +24V power supply input for controller</td>
</tr>
<tr>
<td>27</td>
<td>A14</td>
<td>RESERVE</td>
<td>To be used at factory (Do not use)</td>
</tr>
<tr>
<td>28</td>
<td>B14</td>
<td>RESERVE</td>
<td>To be used at factory (Do not use)</td>
</tr>
<tr>
<td>29</td>
<td>A15</td>
<td>0V</td>
<td>Reference 0V for input/output</td>
</tr>
<tr>
<td>30</td>
<td>B15</td>
<td>0V</td>
<td>Reference 0V for input/output</td>
</tr>
<tr>
<td>31</td>
<td>A16</td>
<td>DO 0</td>
<td>General-purpose output 0</td>
</tr>
<tr>
<td>32</td>
<td>B16</td>
<td>DO 1</td>
<td>General-purpose output 1</td>
</tr>
<tr>
<td>33</td>
<td>A17</td>
<td>DO 2</td>
<td>General-purpose output 2</td>
</tr>
<tr>
<td>34</td>
<td>B17</td>
<td>DO 3</td>
<td>General-purpose output 3</td>
</tr>
<tr>
<td>35</td>
<td>A18</td>
<td>DO 4</td>
<td>General-purpose output 4</td>
</tr>
<tr>
<td>36</td>
<td>B18</td>
<td>END</td>
<td>End-of-run output</td>
</tr>
<tr>
<td>37</td>
<td>A19</td>
<td>BUSY</td>
<td>Command-in-progress output</td>
</tr>
<tr>
<td>38</td>
<td>B19</td>
<td>READY</td>
<td>Ready-to-operate output</td>
</tr>
<tr>
<td>39</td>
<td>A20</td>
<td>DO 5</td>
<td>General-purpose output 5</td>
</tr>
<tr>
<td>40</td>
<td>B20</td>
<td>DO 6</td>
<td>General-purpose output 6</td>
</tr>
<tr>
<td>41</td>
<td>A21</td>
<td>DO 7</td>
<td>General-purpose output 7</td>
</tr>
<tr>
<td>42</td>
<td>B21</td>
<td>DO 8</td>
<td>General-purpose output 8</td>
</tr>
<tr>
<td>43</td>
<td>A22</td>
<td>DO 9</td>
<td>General-purpose output 9</td>
</tr>
<tr>
<td>44</td>
<td>B22</td>
<td>DO 10</td>
<td>General-purpose output 10</td>
</tr>
<tr>
<td>45</td>
<td>A23</td>
<td>DO 11</td>
<td>General-purpose output 11</td>
</tr>
<tr>
<td>46</td>
<td>B23</td>
<td>DO 12</td>
<td>General-purpose output 12</td>
</tr>
<tr>
<td>47</td>
<td>A24</td>
<td>EMG 1</td>
<td>Emergency stop input 1 (used with EMG2)</td>
</tr>
<tr>
<td>48</td>
<td>B24</td>
<td>EMG 2</td>
<td>Emergency stop input 2 (used with EMG1)</td>
</tr>
</tbody>
</table>

**CAUTION**

Pin numbers A14 and B14 are to be used at our factory. Do not make any connection to these terminals. Otherwise malfunction might result.

**NOTE**

Pin number B12 functions as the SERVICE mode input terminal only when the SERVICE mode function is enabled.
3-2 Input Signal Description

Input signals consist of 7 dedicated command inputs, 16 general-purpose inputs, interlock signals and an emergency stop input.

* DI15 functions as the SERVICE mode input when the SERVICE mode function is enabled. In this case, 15 general-purpose inputs are available.

All input circuits other than the emergency stop input use photocoupler-isolated input circuit specs. Only the emergency stop input circuit uses contact point input circuit specs. This contact point is directly connected to the relay coil that turns the internal motor power supply on and off.

3-2-1 Dedicated command input

The dedicated command input is used to control the ERCX controller from a PLC or other external equipment. To accept this input, the READY, BUSY and LOCK signals must be set as follows.

- READY signal: ON
- BUSY signal: OFF
- LOCK signal: ON

If the above conditions are not satisfied, then dedicated command inputs cannot be accepted even if they are input from external equipment. For example, when the BUSY signal is on, this means that the controller is already executing a dedicated command, so other dedicated commands are ignored even if they are input. When the LOCK signal is off, no other commands can be accepted since an interlock is active. (One exception is the reset and servo recovery commands that can be executed even when the LOCK signal is off as long as the READY and BUSY signals meet the above conditions.)

A dedicated command input is accepted when the dedicated command input is switched from "off" to "on" (at the instant the contact point closes). Whether the controller accepts the command or not can be checked by monitoring the BUSY signal.

Note that dedicated command inputs cannot be used as data in a program.

⚠️ CAUTION

The dedicated command inputs explained below must always be pulse inputs. In other words, they must be turned off (contact open) after the BUSY signal turns on. If a dedicated command input is not turned off, then the BUSY signal will remain on even when the command has ended normally. So the next command will not be accepted.

⚠️ CAUTION

When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function" for more details.)

- No dedicated commands can be executed in "SERVICE mode state" when command input from other than the TPB is prohibited.
■ Absolute point movement command (ABS-PT)
This command moves the robot to an absolute position of a point number specified by DI0 to DI9 along an axis coordinate whose origin is defined as 0, at a speed selected by DI10 or DI11. (See "3-2-2 General-purpose input (DI0 to DI15)").

⚠️ **CAUTION**
The DI0 to DI11 status must be confirmed before ABS-PT is executed. (See "3-6-6 When executing a point movement command").

■ Relative point movement command (INC-PT)
This command moves the robot a distance defined by a point number specified by DI0 to DI9 from the current position at a speed selected by DI10 or DI11.

⚠️ **NOTE**
Current position does not always indicate the actual robot position. More accurately, it is the current position data stored in the controller. Each time a movement command is executed correctly, the current position data in the controller is replaced with the target position data of the movement command. Therefore, if the robot is stopped by an interlock while executing a relative movement command, re-executing the same relative movement command moves the robot to the target position. (The robot does not move a relative distance from the stopped position by the interlock.) Similarly, after a robot movement command is executed, the controller still retains the target position data of that movement command as the current position data even if you move the robot to another position by manual operation. When a relative movement command is executed under this condition, the robot moves the specified distance from the target position of the movement command that was previously executed, rather than the actual robot position, so use caution.

Current position data differs from the actual robot position when:
- Emergency stop or interlock (LOCK) was activated while the robot was moving.
- A communication command ^C (movement interruption) was transmitted while the robot was moving.
- The SERVICE mode input was changed while the robot was moving.
- The robot was moved by manual operation.
- The robot was moved by hand during servo-off (including emergency stop).

⚠️ **CAUTION**
The DI0 to DI11 status must be specified before INC-PT is executed. (See "3-6-6 When executing a point movement command").

■ Automatic operation start command (AUTO-R)
This command executes the robot program continuously, starting from the current step. All tasks are executed if the robot program is a multi-task program.

■ Step operation start command (STEP-R)
This command executes the robot program one step at a time, starting from the current step. Only the selected task is executed even if the robot program is a multi-task program.
■ Return-to-origin command (ORG-S)
This command returns the robot to its origin position when the search method is selected as the origin detection method. When the mark method is selected, this command checks the return-to-origin status.

NOTE
Once return-to-origin is performed after the robot cable and absolute battery are connected, there is no need to repeat it even when the controller is turned off. (As an exception, return-to-origin becomes incomplete if the absolute backup function is disabled or a parameter relating to the origin is changed. Return-to-origin must be reperformed in that case.)

CAUTION
When performing return-to-origin by the stroke-end detection method, do not interrupt return-to-origin operation while the origin position is being detected (robot is making contact with its mechanical limit). Otherwise, the operation will stop due to a controller overload alarm and the power will need to be turned off and back on again.

CAUTION
If return-to-origin must be repeated by the stroke-end detection method, wait at least 5 seconds before repeating it.

■ Servo recovery command (SERVO)
After emergency stop, releasing the emergency stop button and turning this input on (closing the contact) turns the servo power on, so the robot is ready for restart.
(As with other dedicated command inputs, the servo recovery command should be a pulse input, so it must be turned off (contact open) when the BUSY signal turns on.)

■ Reset command (RESET)
This command returns the program step to the first step of the lead program and turns off DO0 to DO12 and the memory I/O. It also clears the point variable "P" to 0.

* When PRM33 ("Operation at return-to-origin complete" parameter) is set to 1 or 3, DO4 does not turn off even if the reset command is executed. Likewise, when PRM46 ("Servo status output" parameter) is set to 1, DO7 does not turn off even if the reset command is executed.

NOTE
The lead program is the program that has been selected as the execution program by the TPB or POPCOM. (See "9-4 Switching the Execution Program").
The lead program can also be selected by executing a communication command "@SWI". It may also be selected when the program data is loaded into the ERCX controller from the memory card.
3-2-2 General-purpose input (DI0 to DI15)

These general-purpose inputs are available to users for handling data input in a program. These inputs are usually connected to sensors or switches. These inputs can also be directly connected to a PLC output circuit.

As a special function during execution of an ABS-PT or INC-PT point movement command, DI0 to DI9 can be used to specify the point numbers and DI10 and DI11 to specify the movement speed. As the table below shows, the point numbers should be input with DI0 to DI9 in binary code, to specify P0 to P999. The movement speed is specified as 100% when DI10 and DI11 are off. In other cases, it is set to the speed specified by the parameter. (See "5-2 Parameter Description").

Example of point number setting

<table>
<thead>
<tr>
<th>DI No.</th>
<th>DI9 (2²)</th>
<th>DI8 (2¹)</th>
<th>DI7 (2¹)</th>
<th>DI6 (2¹)</th>
<th>DI5 (2¹)</th>
<th>DI4 (2¹)</th>
<th>DI3 (2¹)</th>
<th>DI2 (2¹)</th>
<th>DI1 (2¹)</th>
<th>DI0 (2¹)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>P1</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>P3</td>
<td>OFF</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>ON</td>
</tr>
<tr>
<td>P7</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>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P15</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>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P31</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>
<td>ON</td>
</tr>
<tr>
<td>P63</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>
<td>ON</td>
</tr>
<tr>
<td>P127</td>
<td>OFF</td>
<td>OFF</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>P254</td>
<td>OFF</td>
<td>OFF</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>P511</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>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>P999</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

Example of point movement speed setting

<table>
<thead>
<tr>
<th>DI11</th>
<th>DI10</th>
<th>Movement speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>100%</td>
</tr>
<tr>
<td>OFF</td>
<td>ON</td>
<td>PRM41</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>PRM42</td>
</tr>
<tr>
<td>ON</td>
<td>ON</td>
<td>PRM43</td>
</tr>
</tbody>
</table>

* DI15 functions as the SERVICE mode input when the SERVICE mode function is enabled. In this case, DI0 to DI14 can be used as the general-purpose inputs.
3-2-3 SERVICE mode input (SVCE)

When the SERVICE mode function is enabled, DI15 functions as the SERVICE mode input (SVCE). The SERVICE mode input is used to notify the ERCX controller whether the current state is a "SERVICE mode state". This input should be turned off (contact open) in "SERVICE mode state". Refer to "10-4 SERVICE mode function" for details on the SERVICE mode function.

**NOTE**
Operation stops immediately if the SERVICE mode input status is changed during robot operation while the SERVICE mode function is enabled.

**NOTE**
Even with the SERVICE mode function enabled, the SERVICE mode input status can be checked in the program as DI15.

3-2-4 Interlock (LOCK)

This input is used to temporarily stop robot movement. The robot immediately stops when this input is turned off (contact open) during execution of a dedicated I/O command or during program operation or return-to-origin operation from the TPB (or PC). (This also interrupts the robot program operation.) As long as this input is off (contact open), no dedicated I/O commands can be executed, and also no programs and return-to-origin operation can be performed from the TPB (or PC). The only exceptions to this are the reset command and servo recovery command that can be executed regardless of whether the LOCK signal is on or off. Leave this LOCK signal turned on (contact closed) during normal operation. Once this LOCK signal is turned off (contact open), the robot remains stopped even after this input is turned back on (contact closed), until another command (AUTO-R, ORG-S, etc.) is input.

3-2-5 Emergency stop inputs 1, 2 (EMG1, EMG2)

Use these inputs to trigger robot emergency stop from an external safety device (for example, safety enclosure, manual safety switch, etc.). Servo power turns off at the same time when the contact between EMG1 and EMG2 is open (turned off). Use a relay contact with a current capacity of at least 50mA. To resume operation, close (turn on) the contact between EMG1 and EMG2, check that the READY signal is turned on, and then input the servo recovery command (SERVO). The servo will turn on to enable robot operation. The TPB or PC can also be used to reset emergency stop when the ERCX controller is connected to the TPB or PC.
3-3 Output Signal Description

The output signals consist of 3 dedicated outputs (READY, BUSY and END) and 13 general-purpose outputs. In this section, terms "ON" and "OFF" mean the output transistors are "on and off".

3-3-1 Dedicated output

The dedicated outputs are used for exchanging signals between the ERCX controller and an external device such as a PLC.

■ Ready-to-operate output (READY)

This output is on as long as the ERCX controller system is in normal operation. If an emergency stop or alarm occurs, then this output turns off to let the motor idle.

- When emergency stop was triggered:
  The READY signal turns on again when emergency stop is cancelled.
  Operation can be restarted by input of the servo recovery command (SERVO) after canceling emergency stop.
- When an alarm was issued:
  If the READY signal is off while the robot is not in emergency stop, this means that an alarm was issued. If this happens, correct the problem while referring to Chapter 13, "Troubleshooting".

  In this case, the ERCX controller should be turned off before attempting to restart operation.

■ Command-in-progress output (BUSY)

The BUSY signal is on during execution of a dedicated command input or a command from the TPB or PC. The BUSY signal turns on when the ERCX controller accepts a dedicated command input. The dedicated command input should be turned off (contact open) when the BUSY signal turns on. The BUSY signal turns off when command execution is complete. (At this point, all the dedicated command inputs must be turned off (contact open).)

⚠️ CAUTION

The dedicated command input must be a pulse input so that it is off when the BUSY signal turns on. If the command input is left on, the BUSY signal cannot turn off even after the command execution is complete. As long as the BUSY signal is on, the ERCX controller will not accept other dedicated command inputs or commands from the TPB or PC. Avoid operating the TPB while the ERCX controller is being operated through the I/O interface. (Doing so might cause malfunctions during data exchange with a PLC or cause communication errors on the TPB side.)

■ End-of-run output (END)

The END signal turns off when a dedicated command input is received and turns on when command execution is complete. The END signal remains off if an error occurs or an interlock or emergency stop is triggered during command execution.

⚠️ CAUTION

When a reset command or a movement command specifying a very small amount of movement is used, the command execution time will be very short. In other words, the period that the END signal is off will be very short (1ms or less in some cases).

The END signal does not change by operation from the TPB or PC.

💡 NOTE

The PRM34 (system mode selection parameter) setting can be changed so that the execution result END signal output for the completed dedicated command occurs only after the dedicated command input turns off. (See section 5-2 "Parameter Description".)
3-3-2 General-purpose output (DO0 to DO12)

These general-purpose outputs are available to users for freely controlling on/off operation in a program. These outputs are used in combination with an external 24V power supply, to drive loads such as solenoid valves and LED lamps. These outputs of course, can be directly connected to a PLC input circuit.

All general-purpose outputs are reset (turned off) when the ERCX controller is turned on or the program is reset.

* When PRM33 ("Operation at return-to-origin complete" parameter) is set to 1 or 3, DO4 does not turn off even if the program is reset. Similarly, when PRM46 ("Servo status output" parameter) is set to 1, DO7 does not turn off even if the program is reset.

3-4 I/O Circuits

This section provides the ERCX controller I/O circuit specifications and examples of how the I/O circuits should be connected. Refer to these specifications and diagrams when connecting to external equipment such as a PLC.

3-4-1 I/O circuit specifications

- **Input power**
  - Input voltage: 24V ±10%

- **Input Circuit**
  - Excluding emergency stop input circuit
    - Insulation method: Photocoupler insulation
    - Input terminal: Relay contact or NPN open collector transistor connected between input terminal and 0V terminal.
    - Input response: 30ms max.
    - Input current: 5mA/DC24V
    - Input sensitivity: Current on: 3mA min.
                              Current off: 1mA max.

  - Emergency stop input circuit
    - Input terminal: Relay contact connected between emergency stop inputs 1 and 2 (between EMG1 and EMG2).
    - Input response: 5ms max.
    - Input current: 33.3mA/DC24V

- **Output Circuit**
  - Insulation method: Photocoupler insulation between internal circuit and output transistor
  - Output terminal: NPN open collector output of all collective output common terminals (0V side)
  - Output response: 1ms max.
  - Max. output current: 0.1A/DC24V per output
  - Residual ON voltage: 1.5V max.
3-4-2 I/O circuit and connection example

When using a separate 24V power supply for I/O control

![Diagram of I/O circuits](image_url)

**CAUTION**

Do not short the output terminal to the DC24V terminal. This may cause equipment breakdown.
When using an inductive load (such as a solenoid valve) as the output load, connect a high-speed diode as a surge killer in parallel and near to the load to reduce noise.
When using a 2-wire type proximity sensor as an input signal, the residual voltage during on/off might exceed the input range for the ER CX controller depending on the sensor type. Using such a sensor will cause erroneous operation. Always check that the sensor meets the input signal specifications.
Keep the wires separated from the power lines of other machines, or shield the wires to prevent noise.

External 24V power supply for I/O control must have a capacity for:
Emergency stop circuit (0.05A) + I/O drive (depends on user application)
Additional power capacity (0.3A) for brake control is needed for robots with brake.
When shared with 24V power supply for controller

**External 24V power supply**

- **Push-button**
- **NPN transistor**
- **Incandescent lamp**
- **Solenoid valve**
- **Photocoupler**

**Input signal**

**Output signal**

**DI**

**DO**

- **Controller side**
- **+IN COM**
- **0V**
- **24V**
- **N**

**CAUTION**

*Select a 24V power supply with a sufficient capacity. If the power supply capacity is insufficient, the robot may not operate normally resulting in an unexpected error or alarm.*

External 24V power supply must have a capacity for:

- Controller (3A) + Emergency stop circuit (0.05A) + I/O drive (depends on user application)
- Additional power capacity (0.3A) for brake control is needed for robots with brake.
3-5 I/O Connection Diagram

3-5-1 General connections

General connections

![I/O Connection Diagram](image-url)
3-5-2 Connection to PLC output unit

Connection to the Mitsubishi® PLC AY51 output unit

AY51 type output unit

ERCX series controller

External DC 24V power supply
3-5-3 Connection to PLC input unit

Connection to the Mitsubishi® PLC AX41 input unit

ERCX series controller

AX41 type input unit

Photocoupler

Internal circuit

External DC 24V power supply

Ready

Busy

End

DO 0

DO 1

DO 2

DO 3

DO 4

DO 5

DO 6

DO 7

DO 8

DO 9

DO 10

DO 11

DO 12

IN COM

0V

TB 1

X00

X01

X02

X03

X04

X05

X06

X07

X08

X09

X10A

X10B

X10C

X10D

X10E

X10F

DC24V

DC24V

+IN COM

0V

A13,B13

A15,B15

A19

A19

B18

B18

A16

A16

B16

B16

A17

A17

B17

B17

A18

A18

B18

B18

A20

A20

B20

B20

A21

A21

B21

B21

A22

A22

B22

B22

A23

A23

B23

B23

R

R

R

R

R

R

R
3-6 I/O Control Timing Charts

The following shows typical timing charts for I/O control. Refer to these diagrams when creating a sequence program.

3-6-1 When turning the power on

When emergency stop is triggered:

- Power supply
  - READY
  - END

- READY
  - END

When emergency stop is canceled:

- Power supply
  - READY
  - END

When an alarm is issued:

- Power supply
  - READY
  - END

- The ERCX initial state depends on whether emergency stop is triggered when the power is turned on.
  - When the power is turned on while emergency stop is cancelled, the ERCX controller starts with the READY signal and also the servo turned on. (Robot is ready to operate in this state.)
  - In contrast, when the power is on while emergency stop is triggered, the ERCX controller starts with the READY signal turned off under emergency stop conditions. (Robot operation is prohibited in this state.)
  - To enable robot operation, cancel the emergency stop to turn on the READY signal, and then input a servo recovery command (SERVO).

- After turning the power on, make sure that the END signal is on before inputting a dedicated command.

- If the READY and END signals are still off for more than the specified time after turning the power on, this means that an alarm has occurred. If that happens, correct the problem while referring to "13-2 Alarm and Countermeasures".
3-6-2 When executing a dedicated input command

- The BUSY signal turns on when a dedicated command is received. Whether the received command has ended normally can be checked with the END signal status at the point that the BUSY signal turns off. When the END signal is on, this means that the command has ended normally. If it is off, the command has not ended normally.

- The dedicated command input must be a pulse input. If the dedicated command input stays on, the BUSY signal does not turn off even after the command has been executed.

(1) When a command with a long execution time runs and ends normally:
   (Command execution is still in progress and the END signal is off when turning off (contact open) the dedicated command input.)

   ![Diagram of BUSY and END signals]

   1. At the rising edge of the dedicated command input, the END signal turns off and the BUSY signal turns on.
   2. Turn off (contact open) the dedicated command input after checking that the BUSY signal turns on.
   3. Wait until the BUSY signal turns off.
   4. The END signal should be on when the BUSY signal turns off, indicating that the command has ended normally.

   **CAUTION**

   In the case of the automatic operation start command (AUTO-R), the END signal turns on and the BUSY signal turns off when the program ends or a STOP statement is executed. If an endless program (one that automatically returns to the top of the program from the last step) is executed, the BUSY signal will not turn off until an interlock or emergency stop is triggered.
When a command with a short execution time runs and ends normally:
(Command execution has already ended and the END signal is on before turning off (contact open) the dedicated command input, as in the examples listed below.)

- A movement command (ABS-PT, INC-PT) for a very short distance was executed.
- A reset command (RESET) was executed.
- A step run was executed using a command with a very short execution time such as the L and DO statements.

(1) At the rising edge of the dedicated command input, the END signal turns off and the BUSY signal turns on.
(2) Turn off (contact open) the dedicated command input after checking that the BUSY signal turns on.
(3) Wait until the BUSY signal turns off. (The BUSY signal immediately turns off since the command execution is already complete.)
(4) The END signal should be on when the BUSY signal turns off, indicating that the command has ended normally.

However, the PRM34 (system mode selection parameter) "bit 7 END output sequence setting at command execution completion" setting can be changed so that the END signal turns on when the dedicated command input turns off.

**NOTE**

The PRM34 (system mode selection parameter) "bit 7 END output sequence setting at command execution completion" setting is supported only in Ver. 13.74 and later versions.

---

Even after dedicated command execution completion, the END signal does not turn on until the dedicated command input turns off.
(3) When a command cannot be executed from the beginning:
(Command execution is impossible from the beginning and the END signal does not turn on, as in the examples listed below.)

- A movement command (ABS-PT, INC-PT) was executed without return-to-origin being completed.
- An operation start command (AUTO-R, STEP-R) was executed while return-to-origin is incomplete (except for cases where PRM48 (Pre-operation action selection parameter) is set to 1 or 3).
- A movement command (ABS-PT, INC-PT) was executed by specifying a point number whose point data is unregistered.
- A dedicated command was executed during interlock or emergency stop (except for the reset (RESET) and servo recovery (SERVO) commands).
- When a dedicated command input (ABS-PT, INC-PT, AUTO-R, STEP-R, ORG-S, SERVO, RESET) was executed in "SERVICE mode state".

![Diagram](#)

(1) At the rising edge of the dedicated command input, the END signal turns off and the BUSY signal turns on.
(2) Turn off (contact open) the dedicated command input after checking that the BUSY signal turns on.
(3) Wait until the BUSY signal turns off. (The BUSY signal immediately turns off since the command cannot be executed from the beginning.)
(4) The END signal remains off when the BUSY signal turns off, indicating that the command could not end normally.
(4) When command execution cannot be completed:
(Command execution stops before completion and the END signal does not turn on, as in the examples listed below.)

- An interlock or emergency stop was triggered during execution of a dedicated command.
- The SERVICE mode input was changed during execution of a dedicated command.
- An error was caused due to a jump to an unregistered program or point during automatic operation.

(1) At the rising edge of the dedicated command input, the END signal turns off and the BUSY signal turns on.
(2) Turn off (contact open) the dedicated command input after checking that the BUSY signal turns on.
(3) Wait until the BUSY signal turns off.
(4) The BUSY signal turns off since the command execution stops before completion.
(5) The END signal remains off when the BUSY signal turns off, indicating that the command could not end normally.
3-6-3 When interlock signal is input

- When an interlock signal is input while a dedicated command is being executed, the BUSY signal turns off. The READY and END signals remain unchanged.
3-6-4 When emergency stop is input

- The READY signal turns off. The BUSY signal also turns off while a dedicated command is being executed. The END signal remains unchanged.
- To enable robot operation, cancel emergency stop to turn on the READY signal, then input the servo recovery command (SERVO).

3-6-5 When alarm is issued

- The READY, BUSY and END signals all turn off.
- Correct the problem while referring to "13-2 Alarm and Countermeasures".
3-6-6 When executing a point movement command

- When executing a point movement command (ABS-PT, INC-PT), the point data and speed data must first be input before inputting the command. The point data and speed data can be specified with DI0 to DI11. Refer to "3-2-2 General-purpose input (DI0 to DI15)".

![Diagram](image)

(1) Specify the point data and speed data, using the general-purpose input DI0 to DI11. These input conditions should be kept unchanged until the BUSY signal turns on. (If these conditions are changed before the BUSY signal turns on, then the data might be misrecognized.)

(2) When a minimum of 30ms has elapsed, input the point movement command (ABS-PT, INC-PT).

(3) At the rising edge of the dedicated command input, the END signal turns off and the BUSY signal turns on.

(4) Turn off (contact open) the dedicated command input after checking that the BUSY signal turns on. Now, you may change the point data and speed data (DI0 to DI11) for the next movement.

(5) Wait until the BUSY signal turns off.

(6) The END signal should be on when the BUSY signal turns off, indicating that the command has ended normally.
3-7  I/O Assignment Change Function

3-7-1  Changing the I/O assignment

The function assigned to each I/O signal can be changed with PRM59 (I/O assignment selection parameter) setting. Refer to the table on next page when you want to change the I/O assignment. After changing the I/O assignment, the ERCX controller must be restarted to enable the changes.

NOTE

The I/O assignment change function is available on controllers whose version is 13.57 or later.
### I/O assignment list

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>ABS-PT</td>
<td>ABS-PT</td>
</tr>
<tr>
<td>B1</td>
<td>INC-PT</td>
<td>INC-PT</td>
</tr>
<tr>
<td>A2</td>
<td>AUTO-R</td>
<td>-</td>
</tr>
<tr>
<td>B2</td>
<td>STEP-R</td>
<td>CHG</td>
</tr>
<tr>
<td>A3</td>
<td>ORG-S</td>
<td>ORG-S</td>
</tr>
<tr>
<td>B3</td>
<td>RESET</td>
<td>RESET</td>
</tr>
<tr>
<td>A4</td>
<td>SERVO</td>
<td>SERVO</td>
</tr>
<tr>
<td>B4</td>
<td>LOCK</td>
<td>LOCK</td>
</tr>
<tr>
<td>A5</td>
<td>D10</td>
<td>P10</td>
</tr>
<tr>
<td>B5</td>
<td>D11</td>
<td>P11</td>
</tr>
<tr>
<td>A6</td>
<td>D12</td>
<td>P12</td>
</tr>
<tr>
<td>B6</td>
<td>D13</td>
<td>P13</td>
</tr>
<tr>
<td>A7</td>
<td>D14</td>
<td>P14</td>
</tr>
<tr>
<td>B7</td>
<td>D15</td>
<td>P15</td>
</tr>
<tr>
<td>A8</td>
<td>D16</td>
<td>SPD1</td>
</tr>
<tr>
<td>B8</td>
<td>D17</td>
<td>SPD2</td>
</tr>
<tr>
<td>A9</td>
<td>D18</td>
<td>-</td>
</tr>
<tr>
<td>B9</td>
<td>D19</td>
<td>-</td>
</tr>
<tr>
<td>A10</td>
<td>D10</td>
<td>-</td>
</tr>
<tr>
<td>B10</td>
<td>D11</td>
<td>-</td>
</tr>
<tr>
<td>A11</td>
<td>D12</td>
<td>-</td>
</tr>
<tr>
<td>B11</td>
<td>D13</td>
<td>-</td>
</tr>
<tr>
<td>A12</td>
<td>D14</td>
<td>-</td>
</tr>
<tr>
<td>B12</td>
<td>D15/SVCE</td>
<td>(SVCE)</td>
</tr>
<tr>
<td>A16</td>
<td>D00</td>
<td>PO0</td>
</tr>
<tr>
<td>B16</td>
<td>D01</td>
<td>PO1</td>
</tr>
<tr>
<td>A17</td>
<td>D02</td>
<td>PO2</td>
</tr>
<tr>
<td>B17</td>
<td>D03</td>
<td>PO3</td>
</tr>
<tr>
<td>A18</td>
<td>D04</td>
<td>PO4</td>
</tr>
<tr>
<td>A20</td>
<td>D05</td>
<td>PO5</td>
</tr>
<tr>
<td>B20</td>
<td>D06</td>
<td>-</td>
</tr>
<tr>
<td>A21</td>
<td>D07</td>
<td>-</td>
</tr>
<tr>
<td>A22</td>
<td>D09</td>
<td>-</td>
</tr>
<tr>
<td>B22</td>
<td>D10</td>
<td>-</td>
</tr>
<tr>
<td>A23</td>
<td>D11</td>
<td>-</td>
</tr>
<tr>
<td>B23</td>
<td>D12</td>
<td>-</td>
</tr>
<tr>
<td>B18</td>
<td>END</td>
<td>END</td>
</tr>
<tr>
<td>A19</td>
<td>BUSY</td>
<td>BUSY</td>
</tr>
<tr>
<td>B19</td>
<td>READY</td>
<td>READY</td>
</tr>
</tbody>
</table>

**Type**
- **Type 0 (Conventional type)**
- **Type 1 (Point number output type)**
- **Type 2 (Point teaching type)**

**Setting**
- **PRM59 (Standard) 0**
- **Setting Type Function**
- **No. of points**
- **No. of speed switching points**
- **Program operation by I/O**

**Pin No.**

**Type 3 (Point teaching type)**
- **Point trace mode**
- **Teaching mode**

---

*1 The PO output format varies depending on whether the PRM59 setting is specified in "hundreds" or "thousands" units. (See section 5-2 "Parameter Description")

*2 Specifies the permissible number of movement points for a point movement command (ABS-PT, INC-PT).

*3 Specifies the permissible number of speed switching points for a point movement command (ABS-PT, INC-PT).
3-7-2 I/O signal description

The meaning of each signal is explained below.

■ Point number designation inputs 0 to 5 (PI0 to PI5)
  These inputs designate the point number of the target position where the robot moves with a point movement command (ABS-PT, INC-PT). (For details on the ABS-PT and INC-PT commands, see 3.2.1, “Dedicated command input” in this chapter.)
  These inputs are also used to designate the point number of the target position where point data is written with a point data write command (PSET).
  The point number of the target position must be specified before running a point movement command or point write command. The point number is specified by a binary code. See the table below to specify each point number.

  **Point number designation example**

<table>
<thead>
<tr>
<th>PI No.</th>
<th>PI5 (2^5)</th>
<th>PI4 (2^4)</th>
<th>PI3 (2^3)</th>
<th>PI2 (2^2)</th>
<th>PI1 (2^1)</th>
<th>PI0 (2^0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>P1</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>P7</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P15</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P31</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P63</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

■ Movement speed setting (SPD1, SPD2)
  Designates the speed at which the robot moves with a point movement command (ABS-PT, INC-PT) or jog movement command (JOG+, JOG-). (For details on the ABS-PT and INC-PT commands, see 3.2.1, “Dedicated command input” in this chapter.)
  The movement speed must be specified before running a point movement command or jog movement command. See the table below to specify the movement speed.

  **Movement speed setting example**

<table>
<thead>
<tr>
<th>SPD2</th>
<th>SPD1</th>
<th>Movement speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>OFF</td>
<td>100%</td>
</tr>
<tr>
<td>OFF</td>
<td>ON</td>
<td>PRM41</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
<td>PRM42</td>
</tr>
<tr>
<td>ON</td>
<td>ON</td>
<td>PRM43</td>
</tr>
</tbody>
</table>

■ Jog movement (+ direction) command (JOG+)
  Moves the robot in jog mode along the + (plus) direction.
  The robot moves in jog mode along the + (plus) direction as long as this signal is on. The movement speed is 100mm/sec.
  This speed can be changed by using SPD1 and SPD2. In this case, the movement speed is given by the following equation.
  \[
  \text{Movement speed [mm/sec]} = 100 \times \frac{\text{Movement speed [%] specified with SPD1 and SPD2}}{100}
  \]

  **CAUTION**
  If the CHG (mode switch input) signal is switched during jog movement, the robot comes to an error stop.

■ Jog movement (- direction) command (JOG-)
  Moves the robot in jog mode along the - (minus) direction.
  The robot moves in jog mode along the - (minus) direction as long as this signal is on. The movement speed is 100mm/sec.
  This speed can be changed by using SPD1 and SPD2. In this case, the movement speed is given by the following equation.
  \[
  \text{Movement speed [mm/sec]} = 100 \times \frac{\text{Movement speed [%] specified with SPD1 and SPD2}}{100}
  \]

  **CAUTION**
  If the CHG (mode switch input) signal is switched during jog movement, the robot comes to an error stop.
**I/O INTERFACE**

3-7 I/O Assignment Change Function

- **Point data write command (PSET)**
  Writes the current position data in the specified point number.
  To use this command, the point number for writing the current position data must first be specified using a PI (point number designation input) input.
  The PSET is enabled only when return-to-origin has been completed.

- **Mode switch input (CHG)**
  Switches the Type 3 (Point teaching type) mode. Selectable modes are as follows.
  (1) Point trace mode
  (2) Teaching mode
  The Type 3 (Point teaching type) mode is switched to "Point trace mode" when the CHG signal is off, and is switched to "teaching mode" when the CHG signal is on.

  **CAUTION**
  If the CHG signal is switched during execution of a point movement command (ABS-PT, INC-PT) or jog movement command (JOG+, JOG-), the robot comes to an error stop.

- **Target position's point number outputs 0 to 5 (PO0 to PO5)**
  These are the output signals for the point movement command (ABS-PT, INC-PT) target position point numbers, and for the point numbers corresponding to the point zone output and movement point zone output functions. (For details on ABS-PT and INC-PT commands, see 3.2.1, "Dedicated command input" in this chapter.)
  The "point zone output function" outputs the corresponding point number to the PO when the robot enters the point zone output range (corresponding point ± position judgment parameter range). The corresponding point of this point zone output range is the point data registered at the controller.
  Moreover, the point zone output range's corresponding point can be further narrowed to correspond to point movement commands (ABS-PT, INC-PT), with the point number being output to the PO. This is referred to as the movement point zone output function.

  **Point zone output function**

  ![Diagram of point zone output function]

  Target position point numbers for point movement commands (ABS-PT, INC-PT) are output as binary values. The same applies to point numbers which correspond to the point zone output function and the movement point zone output function.
  The PO output format is determined by the PRM59 (I/O assignment selection parameter) setting's "hundreds" digit value.
  0: PO output occurs at normal movement completion.
  1: PO output occurs when movement command is received.
  2: Point zone output
    (PO output occurs when the current position enters the point data (registered at the controller) ± position judgment parameter range.)
  3: Movement point zone output (supported only by Ver. 13.64 and later versions)
    (PO output occurs when the current position enters the point data registered at the controller, and the point movement command's (ABS-PT, INC-PT) movement point data ± position judgment parameter range.)
### Output example

<table>
<thead>
<tr>
<th>PO No.</th>
<th>PO5</th>
<th>PO4</th>
<th>PO3</th>
<th>PO2</th>
<th>PO1</th>
<th>PO0</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>P1</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>F7</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P31</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>P63</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

**CAUTION**

- When using PO as an output signal that indicates the target position's point number for point movement commands (ABS-PT, INC-PT):
  - If moving the robot to point 0 with the first point movement command which is executed after turning the controller on, all the PO0 to PO5 signals still remain off (because P0 = 000000 (binary)) even after the robot has moved to point 0. This means that the PO0 to PO5 signal statuses do not change even after the robot has moved to P0, so no information is available to indicate whether the robot motion to P0 is complete (or whether the movement command was received). This should be kept in mind when moving the robot to point 0.
- When using PO as an output signal that indicates the corresponding point number at the point zone output function or the movement point zone output function:
  - If outputting point 0 (P0) as the corresponding point for the point zone output function or the movement point zone output function, all the PO0 to PO5 signals remain off (because P0 = 000000 (binary)). This means that the PO0 to PO5 signal statuses do not change even after the robot has entered the zone specified by P0. This should be kept in mind when monitoring P0.

**NOTE**

- When using PO as an output signal that indicates the target position's point number for point movement commands (ABS-PT, INC-PT):
  - When a point movement is received through a parallel I/O, the target position's point number is output to the corresponding parallel I/O (PO0 to PO5). When received through a serial I/O such as a CC-Link, the target position's point number is output to the corresponding serial I/O (PO200 to PO205).
  - All PO outputs are reset (OFF) when a program reset is performed.
- When using PO as an output signal that indicates the corresponding point number at the point zone output function:
  - The corresponding point number for the point zone output function is output to both the corresponding parallel I/O (PO0 to PO5) and the serial I/O (PO200 to PO205).
  - All PO outputs are reset (OFF) when a program reset is performed.
- When using PO as an output signal that indicates the corresponding point number at the movement point zone output function:
  - The corresponding point number for the movement point zone output function is output to both the corresponding parallel I/O (PO0 to PO5) and the serial I/O (PO200 to PO205).
  - Movement points are reset immediately after a controller power on, and all PO outputs are therefore turned off at that time. Movement points are also reset if a program reset is performed, and the movement point zone PO outputs are reset (OFF) at that time as well.

**Return-to-origin complete output / Zone output 0 (ORG-O / ZONE0)**

This output notifies that return-to-origin operation is complete.

- The ORG-O output turns on when return-to-origin is complete. It remains off as long as return-to-origin is incomplete.
- When Zone 0 output is enabled with PRM53 (Zone output selection parameter), the ORG-O output is used as the output port of Zone 0. For details on the zone output signal, refer to “5.2 Parameter Description”.

**Servo status output / Zone output 1 (SRV-O / ZONE 1)**

The SRV-O output turns on when the servo is on and turns off when the servo is off.

- When Zone 1 output is enabled with PRM53 (Zone output selection parameter), the SRV-O output is used as the output port of Zone 1. For details on the zone output signal, refer to “5.2 Parameter Description”.

---

3-7 I/O Assignment Change Function

I/O INTERFACE
3-7-3 Timing chart

This section shows timing charts for the operations that are added by changing the I/O assignment.

■ Jog movement (JOG+, JOG-)

Mode switch input (CHG)

Jog movement command (JOG+ / JOG-)

END

BUSY

READY

Robot movement

30ms or less 1ms or less 30ms or less 1ms or less

(1) Turn on the CHG signal.
(2) Turn on the JOG+ (or JOG-) input signal while the CHG signal is on.
(3) The END signal turns off and the BUSY signal turns on, indicating that the ERCX received the jog movement command.
(4) The robot moves in jog mode as long as the JOG+ (or JOG-) input signal is on.
(5) Turn off the JOG+ (or JOG-) input signal.
(6) Wait until the BUSY signal turns off.
(7) The BUSY signal turns off. The END signal should be on at this point, indicating that the jog movement is normally complete.

⚠️ CAUTION

If the CHG signal is switched during execution of a jog movement command (JOG+, JOG-), the robot comes to an error stop and the END signal remains off.
Precondition: The CHG signal is on before and during point data writing (until the following procedure is complete).

(1) Designate the point number input (PI0 to PI5) to write the point data.
   - The point numbers that can be used depend on the I/O assignment type. Refer to the I/O assignment list in "3-7-1 Changing the I/O assignment".
   - The input status for designating the point number must be kept unchanged until step (3) is complete. If this input status is changed, the ERCX might misrecognize the data.

(2) After 30ms or more has elapsed, turn on the PSET.

(3) The END signal turns off and the BUSY signal turns on, indicating that the ERCX received the point data write command.

(4) Turn off the PSET.

(5) Wait until the BUSY signal turns off.

(6) The BUSY signal immediately turns off since point data writing is already finished. The END signal should be on at this point, indicating that the point data writing was completed normally.
■ Target position’s point number output (PO)

(1) Outputting the point number at the timing that movement is normally completed

<table>
<thead>
<tr>
<th>Command ①</th>
<th>Command ②</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point number output ①</td>
<td>Point number output ②</td>
</tr>
</tbody>
</table>

*: The number of point number outputs that can be used depends on the I/O assignment type.

Precondition: 1) The following steps are explained assuming that PRM59=30.

When the PRM59 setting = 30

<table>
<thead>
<tr>
<th>I/O assignment type</th>
<th>Type 3 (point teaching type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permissible number of movement points</td>
<td>64 points</td>
</tr>
<tr>
<td>Point output selection</td>
<td>Point No. output to PO when movement ends normally</td>
</tr>
</tbody>
</table>

2) The point numbers of the target positions are designated before running a point movement command (ABS-PT, INC-PT).

[Point movement command execution ①]

(1) Turn on the ABS-PT (or INC-PT).

(2) The END signal turns off and the BUSY signal turns on, indicating that the ERCX received the point movement command.

(3) Turn off the ABS-PT (or INC-PT).

(4) Wait until the BUSY signal turns off.

(5) The BUSY signal turns off. The END signal should be on at this point, indicating that the point movement is normally finished.

(6) When the END signal is on in step (5), the target position’s point number is output from the specified point number (PO0 to PO5).

  • The output status of the target position’s point number is retained until execution of the next point movement command is complete.

↓

[Point movement command execution ②]

(7) Execute the next point movement command.

(8) Point movement ends.

(9) The END signal turns on. The previous target position’s point number being output from the specified point number (PO0 to PO5) is cleared and the current target position’s point number is then output.

⚠️ **CAUTION**

If moving the robot to point 0 with a point movement command that is first executed after turning on the controller, all of PO0 to PO5 still remain off (because P0 = 000000 (binary)) even after the robot has moved to point 0. This means that the PO0 to PO5 status does not change even after the robot has moved to P0, so no information is available to indicate whether the robot motion to P0 is complete (or whether the movement command was received). This should be kept in mind when moving the robot to point 0.
(2) Outputting the point number at the timing that a movement command is received

Point movement command
(ABS-PT, INC-PT)

Target position’s point number
outputs 0 to 5*
(PO0 to PO5)

End

Busy

Robot movement

30ms or less 1ms or less 1ms or less 30ms or less 1ms or less 1ms or less

*: The number of point number outputs that can be used depends on the I/O assignment type.

Precondition:

1) The following steps are explained assuming that PRM59=130.

<table>
<thead>
<tr>
<th>When the PRM59 setting = 130</th>
<th>I/O assignment type</th>
<th>Type 3 (point teaching type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permissible number of movement points</td>
<td>64 points</td>
<td></td>
</tr>
<tr>
<td>Point output selection</td>
<td>Point No. output to PO when movement command is received</td>
<td></td>
</tr>
</tbody>
</table>

2) The point numbers of the target positions are designated before running a point movement command (ABS-PT, INC-PT).

[Point movement command execution ①]

(1) Turn on the ABS-PT (or INC-PT).

(2) The END signal turns off and the BUSY signal turns on, indicating that the ERCX received the point movement command.

(3) When the BUSY signal turns on in step (2), the target position’s point number is output from the specified point number (PO0 to PO5).

\[\bullet\text{The output status of the target position’s point number is retained until the next point movement command is received.}\]

(4) Turn off the ABS-PT (or INC-PT).

(5) Wait until the BUSY signal turns off.

(6) The BUSY signal turns off. The END signal should be on at this point, indicating that the point movement finished normally.

↓

[Point movement command execution ②]

(7) Execute the next point movement command.

(8) When the ERCX received the point movement command and the BUSY signal turned on, the previous target position’s point number being output from the specified point number (PO0 to PO5) is cleared and the current target position’s point number is then output.

⚠️ **CAUTION**

*If moving the robot to point 0 by specifying it with a point movement command that is first executed after turning on the controller, all of PO0 to PO5 still remain off (because P0 = 000000 (binary)) even after the robot has moved to point 0. This means that the PO0 to PO5 status does not change even after specifying P0 as the target position, so no information is available to indicate whether the movement command to P0 was received. This should be kept in mind when moving the robot to point 0.*
### 3-7 I/O Assignment Change Function

#### (3) Outputting the corresponding point number by the point zone output function

Zone outputs (ZONE 0, ZONE 1) are also explained here.

- **Target position’s point number outputs 0 to 3** (PO0 to PO3)
  - **Point output (point m)**
  - **Point output (point n)**

#### Zone output 0 (ZONE 0)
- **Positive logic**

#### Zone output 1 (ZONE 1)
- **Positive logic**

#### Current robot position

- **Point zone output range**
- **Zone output range**

---

* The number of target point number outputs that can be used depends on I/O assignment type.

### Precondition:

1. The following steps are explained assuming that PRM59=221.

<table>
<thead>
<tr>
<th>I/O assignment type</th>
<th>Type 2 (Point No. output type)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permissible number of movement points</td>
<td>16 points</td>
</tr>
<tr>
<td>Point output selection</td>
<td>Point zone output</td>
</tr>
<tr>
<td>Point zone judgment method (position judgment parameter)</td>
<td>OUT valid position</td>
</tr>
</tbody>
</table>

2. The Zone 0 and Zone 1 output signals are enabled and set to positive output by the Zone output selection parameter (PRM53).

1. Target position’s point number outputs PO0 to PO3 are off since the current robot position is not within the point zone output range. ZONE 0 and ZONE 1 output signals are also off since the robot does not yet enter the zone output range.

2. Outputs the corresponding point number through PO0 to PO3 since the current robot position is within the point zone output range (Pm ± OUT valid position range). ZONE 0 and ZONE 1 output signals are still off since the robot does not yet enter the zone output range.

3. As with (1), all the target position’s point number outputs PO0 to PO3, ZONE 0 signal and ZONE 1 output signal are off.

4. ZONE 0 output signal turns on since the current robot position is within the zone output range (P900 to P901). (ZONE 1 signal remains off since the robot is not within the zone output range of P902 to P903). At this point, the target position’s point number outputs PO0 to PO3 are still off since the robot is not within the point zone output range.

5. Outputs the corresponding point number through PO0 to PO3 since the current robot position is within the zone output range (P902 to P903) and also within the point output range (Pn ± OUT valid position range). At this point, ZONE 1 output signal turns on. (ZONE 0 output signal turns off since the robot is not within the zone output range of P900 to P901).
NOTE

- When using an optional unit such as a CC-Link, the corresponding point number for the point zone output function is output to both the corresponding parallel I/O (PO0 to PO5) and the serial I/O (PO200 to PO205).
- At controller Ver. 13.64 and later versions, the point zone judgment method can be selected (by the position judgment parameter) as either the "OUT valid position" or "positioning-completed pulse" (this setting is specified by the PRM59 setting's "thousands" digit value).
  In versions prior to Ver. 13.64, only the "OUT valid position" can be used as the point zone judgment method (specified by the position judgment parameter).
- The "OUT valid position" can be changed by parameter setting (PRM20).
- The "positioning-completed pulse" can be changed parameter setting (PRM6).

CAUTION

- When the current robot position is within two or more point zone output ranges, the smaller or smallest point number is output.
  Example: If the current robot position is within two point output ranges specified by P2 and P5, then P2 is output.
- If the current robot position is not within any point output range, all of PO0 to PO5 turn off.
- A 10ms sampling time is needed for position monitoring, so the point zone output might not be detected when moving the robot at high speeds.
- If outputting point 0 (P0) as the corresponding point for the point zone output function, all of PO0 to PO5 remain off (because P0 = 000000 (binary)). This means that the PO0 to PO5 status does not change even after the robot has entered the zone specified by P0. This should be kept in mind when monitoring P0.
(4) Outputting the corresponding point number by the movement point zone output function

Zone outputs (ZONE 0) are also explained here.

Target position’s point number outputs 0 to 3* (PO0 to PO3)

<table>
<thead>
<tr>
<th>Point output</th>
<th>PO0</th>
<th>PO1</th>
<th>PO2</th>
<th>PO3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zone output 0 (ZONE 0)</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
</tbody>
</table>

Current robot position

- **Precondition:**
  1) The following assumes a PRM59=321 setting.

<table>
<thead>
<tr>
<th>When PRM59 setting = 321</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O assignment type</td>
</tr>
<tr>
<td>Permissible number of movement points</td>
</tr>
<tr>
<td>Point output selection</td>
</tr>
<tr>
<td>Point zone judgment method (position judgment parameter)</td>
</tr>
</tbody>
</table>

2) The Zone 0 output signal is enabled and set to positive output by the Zone output selection parameter (PRM53).

3) Set the movement point as P6.

- **(1)** Although the robot is within the P1 ± OUT valid position range (point zone output range), all the PO0 to PO3 target position point number outputs are off because P1 is not the movement point. Moreover, the ZONE 0 output is also off because the robot is not within the specified zone output range.

- **(2)** All the PO0 to PO3 target position point number outputs are off because the robot is not within the point zone output range. Moreover, the ZONE 0 output is also off because the robot is not within the specified zone output range.

- **(3)** The corresponding point number P6 is output to PO0 through PO3 (P1, P2 are on; P0, P3 are off) because the robot is within the P6 ± OUT valid position range (point zone output range), and because P6 is the movement point. ZONE 0 remains off at this time because the robot is not within the specified zone output range.

- **(4)** The ZONE 0 output turns on because the robot is within the specified zone output range (P900 to P901). All the PO0 to PO3 target position point number outputs are off at this time because the robot is not within any point zone output range.

* The number of target point number outputs that can be used depends on I/O assignment type.
NOTE

- The movement point zone output function is supported only in Ver. 13.64 and later versions.
- When using an optional unit such as CC-Link, the corresponding point number for the movement point zone output function is output to the corresponding parallel I/O (PO0 to PO5) and the serial I/O (PO200 to PO205).
- The movement point number specified just prior to movement START by point movement command (ABS-PT, INC-PT) is registered as the movement point.
- Because movement points are reset immediately after a controller power on, all PO outputs turn off. Movement points are also reset when the RESET command is executed, and movement point zone outputs by PO are cleared.
- The point zone judgment method can be selected as either the "OUT valid position" or "positioning-completed pulse" (this setting is specified by the PRM59 setting's "thousands" digit value).
- The "OUT valid position" can be changed by parameter setting (PRM20).
- The "positioning-completed pulse" can be changed parameter setting (PRM6).

CAUTION

- All the PO0 to PO5 outputs are off when the robot is not within the point zone output range.
- A 10ms sampling time is needed for position monitoring, so the point zone output may not be detected during high-speed robot motion.
- When outputting point 0 (P0) as the corresponding point for the movement point zone output function, all the PO0 to PO5 outputs remain off (because P0 = 000000 (binary)). Therefore, the PO0 to PO5 statuses do not change even after the robot has entered the zone specified by P0. This should be kept in mind when monitoring P0.
The TPB is a hand-held, pendant-type programming box that connects to the ERCX controller to edit or run programs for robot operation. The TPB allows interactive user operation on the display screen so that even first-time users can easily operate the robot with the TPB. This chapter describes the basic operation of the TPB.
4-1 Connecting and Disconnecting the TPB

4-1-1 Connecting the TPB to the ERCX controller

⚠️ CAUTION

Do not modify the TPB cable or use any type of relay unit for connecting the TPB to the ERCX controller. Doing so might cause communication errors or malfunctions.

■ When the power supply to the controller is turned off

Connect the TPB connector to the connector labelled "TPB" on the front panel of the controller and supply power to the controller. A beep sounds for approximately 1 second and then the screen shown at the right appears. This screen is referred to as the "Initial screen" from this point onwards.

■ When the power supply to the controller is turned on

The TPB can also be connected to the ERCX controller if the power supply to the controller is on. When the TPB is connected, a beep sounds for about 1 second and then the initial screen appears. At this point, the robot servo may turns off from the turn-on state. (See "4-1-3 Different points from SRCX and DRCX controllers".)

If the TPB is connected while the controller is executing a program or an I/O dedicated command, the execution will be interrupted and the robot operation will halt.

⚠️ CAUTION

Any of the messages "08: PNT DATA DESTROY", "09: PRM DATA DESTROY" or "10: PGM DATA DESTROY" may appear on the TPB when the power to the controller is turned on. (See "13-2 Alarm and Countermeasures".) If one of these messages appears, turn off the power to the controller and then turn it back on again while the emergency stop button of the TPB is still depressed. In this state, the robot servo remains off, but the initial screen appears on the TPB to allow key operation, so initialize and restore the data.

If the message "05: BATT. LOW-VOLTAGE" appears on the TPB when the power is turned on, turn off the power to the controller and then turn it on again while the emergency stop button of the TPB is still depressed. In this state, the robot servo remains off, but the initial screen appears on the TPB to allow key operation, so make a backup of the data, and then replace the lithium battery in the controller (the lithium battery normally lasts five years). (See "14-2 Replacing the System Backup Battery").

If the message "SIO error" is displayed on the TPB, check whether the I/O dedicated command input is on. If the dedicated command input is on, the TPB cannot be used, so the dedicated input must always be a pulse input (the dedicated command input must be off when the BUSY signal turns on.) (Refer to "3-2-1 Dedicated command input").
4-1-2 Disconnecting the TPB from the ERCX controller

The TPB can be disconnected from the controller regardless of whether the power is on or off. There is no problem even when the robot is operating. When the TPB will be left disconnected from the controller for a long period of time, we recommend attaching the RS-232C connector dust cover (supplied) to the TPB connector on the controller.

4-1-3 Different points from SRCX and DRCX controllers

The SRCX and DRCX controllers have an ESC switch (used to connect or disconnect the TPB from the controller) on the front panel of the controller, but the ERCX does not have it. Because of this, the robot servo may turn off when the TPB is connected or disconnected from the controller. (The status LED that is lit in green changes to green/red blinking.) If this happens, perform the servo recovery processing with the TPB (according to the menu that automatically appears in the AUTO mode operation) or execute the servo recovery command (SERVO) through the I/O port. This allows the robot to restart the normal operation. If a problem occurs in the system when the servo is turned off, try connecting and disconnecting the TPB as illustrated below. This will prevent the robot servo from being turned off. Use caution not to deform the connector pins when connecting and disconnecting the TPB.

Connecting the TPB

First plug in the lower part of the connector as shown, and then slowly insert the entire connector in place.

Disconnecting the TPB

Disconnect while pulling from the upper part of the TPB.
4-2 Basic Key Operation

1) Selectable menu items are displayed on the 4th line (bottom line) of the TPB screen.
Example A is the initial screen that allows you to select the following modes.

![Diagram of menu options]

The number to the left of each mode corresponds to the function keys from F1 to F4.

2) On the initial screen shown in A, pressing a function key moves to a lower level in the menu hierarchy. (A→B→C→D)

To return to the previous screen or menu level, press the ESC key. (See "4-4 Hierarchical Menu Structure" in this chapter.)

3) If an error occurs during operation, a buzzer sounds for approximately 1 second and an error message like that shown in Example E appears on the 3rd line of the screen. If this happens, check the contents of the error message and then press the ESC key. The error message will be cleared to allow continuing operation. To correct the error, refer to the message tables in Chapter 12.

4) If an alarm occurs during operation, its alarm message appears on the 3rd line of the screen and a buzzer keeps sounding. The TPB cannot be used in this state. Turn off the power to the controller and then correct the problem by referring to "13-2 Alarm and Countermeasures".
4-3  Reading the Screen

The following explains the basic screen displays and what they mean.

4-3-1  Program execution screen

The display method slightly differs depending on the version of TPB.

Ver. 12.50 or earlier

Ver. 12.51 or later

1. Current mode
2. Execution speed
3. No. of task being executed
4. No. of program being executed
   * On TPB version 12.51 or later, when switched from the lead program to another program, this area shows the program numbers as the "currently executed program / lead program".
5. No. of step being executed
6. Current position

4-3-2  Program edit screen

1. Current mode
2. No. of program being edited
3. No. of step being edited
4-3-3  Point edit screen (teaching playback)

1. Current mode
2. Speed selection number
3. Speed parameter (%)
4. Edit point number
5. Current position

4-3-4  DIO monitor screen

1. General-purpose input
   From left
   DI15 to DI8

2. General-purpose input
   From left
   DI7 to DI0

3. Dedicated input
   From left
   Interlock (LOCK)
   0: Locked state (robot movement not possible)
   1: Unlocked state (robot movement possible)
   Return-to-origin command (ORG-S)
   Reset command (RESET)
   Automatic operation start command (AUTO-R)
   Step operation start command (STEP-R)
   Absolute point movement command (ABS-PT)
   Relative point movement command (INC-PT)
   Servo recovery command (SERVO)

4. General-purpose output
   From left
   DO12 to DO5

5. Dedicated and general-purpose outputs
   From left
   READY, BUSY, END, DO4 to DO0

6. Origin sensor status and servo status
   From left
   XO: Origin sensor status
   0: Off (Closed)
   1: On (Open)
   XS: Servo status
   0: Servo off
   1: Servo on
The menu hierarchy might slightly differ depending on the versions of the controller and TPB.
4-5 Restricting Key Operation by Access Level

The TPB key operations can be limited by setting the access levels (operation levels). A person not trained in robot operation might accidentally damage the robot system or endanger others by using the TPB incorrectly. Set the access levels to restrict TPB key operations and prevent such accidents.

NOTE
The access level settings are protected by a password so that changes cannot be instantly made.

4-5-1 Explanation of access level

The access levels can be set individually for editing, operation, system and memory card. The details of the key operations limited at each level are explained below.

Editing

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All operations are permitted.</td>
</tr>
<tr>
<td>1</td>
<td>Program editing is prohibited. (Program data can be checked.)</td>
</tr>
<tr>
<td>2</td>
<td>In addition to Level 1, point data editing, manual release of brake and point trace (movement to registered data point) are prohibited. (The [X+] and [X-] keys can be used to move the robot and general-purpose outputs can be controlled.)</td>
</tr>
<tr>
<td>3</td>
<td>Any operation in EDIT mode is prohibited. (Cannot enter EDIT mode.)</td>
</tr>
</tbody>
</table>

Operation

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All operations are permitted.</td>
</tr>
<tr>
<td>1</td>
<td>Changing the execution speed and program is prohibited.</td>
</tr>
<tr>
<td>2</td>
<td>In addition to Level 1, automatic operation, step operation and program reset are prohibited. (Return-to-origin can be performed and variables can be monitored.)</td>
</tr>
<tr>
<td>3</td>
<td>Any operation in OPRT mode is prohibited. (Cannot enter OPRT mode.)</td>
</tr>
</tbody>
</table>

System-related data

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All operations are permitted.</td>
</tr>
<tr>
<td>1</td>
<td>Initialization is prohibited.</td>
</tr>
<tr>
<td>2</td>
<td>In addition to Level 1, changing the parameters and setting the option units are prohibited. (Parameter data and option unit settings can be checked.)</td>
</tr>
<tr>
<td>3</td>
<td>Parameter editing, initialization and option setting are prohibited. (Cannot enter SYS-PRM, SYS-INIT and SYS-OPT modes.)</td>
</tr>
</tbody>
</table>
4-5 Restricting Key Operation by Access Level

4-5-2 Changing an access level

1) Press [F3](SYS) on the initial screen.

2) Press [F4](next) to switch the menu display and then press [F1](SAFE).

3) When the password entry screen appears, enter the password and press ▶.

4) When the password is accepted, the screen shown on the right appears.

Press [F1](ACLV) here.
5) Select the item you want to change.
   To change the access level for editing, press \textbf{F1}(EDIT).
   To change the access level for operation, press \textbf{F2}(OPRT).
   To change the access level for system-related data, press \textbf{F3}(SYS).
   To change the access level for memory card, press \textbf{F4}(CARD).

6) The currently set access level appears.
   To change this setting, use the number key to enter the access level and then press \textbf{F5}.

7) When the access level has been changed, the memory write screen appears.
   To save the change permanently (retain the change even after the controller power is turned off), press \textbf{F1}(SAVE).
   To save the change temporarily (retain the change until the power is turned off), press \textbf{F2}(CHG).
   To cancel changing of the setting, press \textbf{F3}(CANCEL).

8) When writing is complete, the screen returns to step 6.

\underline{NOTE}
The password is identical to the ERCX controller’s version number. For example, if the controller version is 13.13, enter 13.13 as the password. Once the password is accepted, it will not be requested unless the TPB is disconnected from the controller or the controller power is turned off.

\underline{NOTE}
To avoid access level conflict between operation and others, the access levels may be automatically adjusted. For example, if the access levels related to editing, system and memory card are “0”, they are automatically changed to “1” when the operation-related access level is “1” or “2” or “3”. The access levels remain unchanged if they are “1” or “2” or “3”.
Chapter 5  PARAMETERS

The ERCX controller uses a software servo system, so no adjustment of hardware components such as potentiometers or DIP switches are required. Instead, the ERCX controller uses parameters that can be easily set or changed by the TPB or PC (personal computer).

This chapter contains a detailed description of each of the parameters, and explains how to use the TPB to change and specify parameter settings.

SAFETY

Errors such as motor overload are detected by the software, so the controller parameters must be set correctly to match the connected robot model. The parameters are initialized to match the robot model when the robot is shipped, so confirm them before starting use. If there is any trouble, please contact our sales office or sales representative.
5-1 Setting the Parameters

1) On the initial screen, press F3 (SYS).

![Menu]
select menu

EDIT OPRT SYS MON

2) Next, press F1 (PRM).

![Sys]
select menu

PRM B.UP INIT

3) The current PRM0 (robot type number) setting appears on the screen. Use the STEP UP and STEP DOWN keys to scroll the parameters until you find the parameter you want to set.

![Sys-prm]
PRM0 = 90
robot type
read only

4) When the desired parameter is displayed, enter the new value with the number keys and then press .

![Sys-prm]
PRM1 = 450 [mm]
(+ )soft limit
range -9999→9999

5) When the setting is complete, the cursor moves back to the beginning of the parameter data.

![Sys-prm]
PRM1 = 450 [mm]
(+ )soft limit
range -9999→9999
5-2 Parameter Description

The parameters are described in order below.

⚠️ CAUTION
Parameters not displayed on the TPB screen are automatically set or optimized to match the robot type when the robot parameters are initialized. You usually do not have to change these parameter settings. If for some special reason you need to change or check these hidden parameters, use any of the following methods.

- Turn on the power to the controller while holding down the [ESC] key on the TPB.
- Connect the TPB to the controller while holding down the [ESC] key on the TPB.
- Use the system utility mode that allows you to display hidden parameters. (See "10-5-1 Viewing hidden parameters").

Take extra caution when handling hidden parameters.

PRM0: Robot type number

This parameter shows the robot number currently used. (See "15-1-2 Robot number list"). This is a read-only parameter. When changing the robot number or if the memory contents are corrupted, perform parameter initialization. (See "10-1 Initialization").

PRM1: (+) soft limit

The + side robot movement range is set. Set a suitable value for safety purposes.

Input range: -9999 to 9999 (mm) or -360 to 360 (°)
Default value: Depends on robot type.

⚠️ CAUTION
The soft limit will not work unless return-to-origin has been completed.

PRM2: (-) soft limit

The - side robot movement range is set. Set a suitable value for safety purposes.

Input range: -9999 to 9999 (mm) or -360 to 360 (°)
Default value: Depends on robot type.

⚠️ CAUTION
The soft limit will not work unless return-to-origin has been completed.

PRM3: Payload

This specifies the total weight of the workpiece and tool attached to the robot. In cases where this weight varies, enter the maximum payload. Based on this parameter, the controller determines the optimum acceleration speed for the robot, so ensure that the correct payload is set. If set too small, abnormal vibration or overheat may occur resulting in troubles with the robot or controller. Conversely, if this parameter is larger than the actual payload, a loss of the cycle time occurs which lowers productivity.

Input range: Depends on robot type. Units are in kilograms (kg).
Default value: 0

* This parameter is set to maximum payload when the controller is shipped from factory.
**PRM4: Acceleration**

This parameter sets the acceleration. The controller will automatically set optimum acceleration according to the robot type and payload. Change this parameter when the acceleration is to be decreased beyond this state.

- **Input range:** 1 to 100 (%)
- **Default value:** 100

**PRM5: Return-to-origin direction**

This parameter sets the return-to-origin direction. Return-to-origin is usually performed toward the motor side when this parameter is set to 0, and toward the non-motor side when set to 1. However, this direction may be reversed depending on the robot variations (such as bent model and vertical type model).

- **Input range:** 0 or 1
- **Default value:** Depends on robot type.

* In terms of motor rotation, when this parameter is 0, the return-to-origin direction is CCW (counterclockwise) as seen from the load.

⚠️ **CAUTION**

The return-to-origin direction cannot be changed in some robot types. Before attempting to change this parameter for the robot you are using, be sure to read the robot mechanical manual or catalog specs to check whether the return-to-origin direction can be changed.

**PRM6: Positioning-completed pulse**

This specifies the range in which the controller determines that positioning is complete. When a movement command is executed, the robot moves toward the target position. The controller then determines that the positioning has been completed when the remaining distance to the target position is within this parameter setting. However, the robot continues moving until it reaches the target position even after the robot enters the "positioning-completed pulse" range.

Since executing the next movement command is not allowed until the positioning is complete, setting a large value for this parameter can reduce cycle time in cases where critical positioning accuracy is not required.

- **Input range:** 1 to 4000 (pulses)
- **Default value:** 80

* If the range specified by this parameter is larger than the range of the OUT valid position, the controller does not decide that the "positioning-completed pulse" range is entered until the axis reaches the OUT valid position.

**PRM7: I/O point movement command speed**

This parameter is used when using the SRCA compatible mode. It is not used in normal operation mode.

- **Input range:** 0 to 100 (%)
- **Default value:** 30
PRM8: **No. of conditional input points**

This parameter specifies the number of effective points for the third data conditional input for executing the JMPF statement of the robot language.

For example, when the default setting is selected for this parameter, the four points from DI0 to DI3 are used as the conditional inputs for the JMPF statement.

- **Input range:** 1 to 8 (points)
- **Default value:** 4

<table>
<thead>
<tr>
<th>No. of conditional input points</th>
<th>General-purpose input</th>
<th>Setting range</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DI0</td>
<td>0 to 1</td>
</tr>
<tr>
<td>2</td>
<td>DI0 to DI1</td>
<td>0 to 3</td>
</tr>
<tr>
<td>3</td>
<td>DI0 to DI2</td>
<td>0 to 7</td>
</tr>
<tr>
<td>4</td>
<td>DI0 to DI3</td>
<td>0 to 15</td>
</tr>
<tr>
<td>5</td>
<td>DI0 to DI4</td>
<td>0 to 31</td>
</tr>
<tr>
<td>6</td>
<td>DI0 to DI5</td>
<td>0 to 63</td>
</tr>
<tr>
<td>7</td>
<td>DI0 to DI6</td>
<td>0 to 127</td>
</tr>
<tr>
<td>8</td>
<td>DI0 to DI7</td>
<td>0 to 255</td>
</tr>
</tbody>
</table>

PRM9: **MOVF speed**

This sets the speed at which the robot moves when the program language MOVF statement is executed.

- **Input range:** 1 to 10000 (mm/sec)
- **Default value:** 10

PRM10: **Return-to-origin speed**

This specifies the movement speed during return-to-origin.

- **Input range:** 1 to 100 (mm/sec)
- **Default value:** 20

**CAUTION**

When the return-to-origin speed is increased, an alarm might be issued during return-to-origin depending on the robot type. We recommend using the default value as much as possible.

PRM11: **No. of encoder pulses (4X mode)**

This parameter sets the number of signal pulses (resolver resolution) per one turn of the motor.

- **Default value:** 16384 (pulse/rev.)

PRM12: **Lead length**

This parameter sets the robot lead length (distance the robot moves while the motor makes one turn). For rotational type robots such as the FROP, this parameter is set to an angle through which the robot rotates while the motor makes one turn.

- **Default value:** Depends on robot type. (Unit: 0.01mm or 0.01deg.)
**PRM13: Origin detection method**

This parameter is used to select the origin (reference point) detection method. There are two methods for detecting the origin: search method and mark method. The search method is further divided into the origin sensor method and stroke-end detection method. In the mark method, you can move the robot to a desired position (mark position) and set it as the particular coordinate position to determine a reference point.

Set this parameter to "0" when detecting the origin position with an origin sensor (sensor method), or set to "1" when detecting the origin by the stroke-end detection method, or set to "2" when using the mark method.

- **Input range:** 0 to 2
- **Meaning:**
  - 0: Sensor method (Cannot be used with ERCX)
  - 1: Stroke-end detection method
  - 2: Mark method
- **Default value:** Depends on robot type.

⚠️ **CAUTION**
The ERCX does not support the sensor method for return-to-origin.

**PRM14: Overload current**

This sets the reference current value used to detect an overload.

**Default value:** Equal to the motor rated current.

**PRM15: Overload time**

This specifies conditions such as time required to detect an overload.

The default value is set so that an overload alarm is issued when a current three times higher than the overload current (PRM14) flows for a period of 3 seconds or an equivalent condition is detected.

**Default value:** 240

**PRM16: Current limit**

This sets the maximum motor input current.

**Default value:** Depends on robot type.

**PRM17: Speed proportional gain**

This sets the speed control gain. Typically, PRM17 and PRM18 should be input at a ratio of 3 : 2.

Generally, the larger the gain, the higher the acceleration will be. However, if the gain is set too high, abnormal oscillation or noise might be generated, causing serious problems in the robot and controller. Use caution when selecting this parameter to avoid such problems.

**Default value:** Depends on robot type.
**PRM18: Speed integration gain**

This sets the speed control gain. Typically, PRM17 and PRM18 should be input at a ratio of 3:2. Generally, the larger the gain, the higher the acceleration will be. However, if the gain is set too high, abnormal oscillation or noise might be generated, causing serious problems in the robot and controller. Use caution when selecting this parameter to avoid such problems.

Default value: Depends on robot type.

**PRM19: Position proportional gain**

This sets the position control gain. If this parameter is changed carelessly, serious problems may occur in the robot and controller.

Default value: Depends on robot type.

**PRM20: OUT valid position**

This specifies the range in which the controller determines that movement command is complete. When a movement command is executed, the robot moves toward the target position. The controller then determines that the movement command has ended when the remaining distance to the target position is within this parameter setting. The controller then initiates the subsequent step processing when the robot reaches this OUT valid position, so setting this parameter to a larger value can reduce cycle time. However, if the subsequent command is a movement command, it is not executed until the ongoing positioning is complete.

Input range: 0 to 9999 (mm)  
0 to 360 (°)

Default value: 1

**PRM21: Position data unit**

This parameter sets the units in which point data is to be displayed. It also specifies whether to enable the limitless movement function.

Input range: 0 to 3  
Meaning: 0: mm (millimeters); limitless movement function disabled (off)  
1: ° (deg.); limitless movement function disabled (off)  
2: mm (millimeters), limitless movement function enabled (on)  
3: ° (deg.); limitless movement function enabled (on)

Default value: Depends on robot type.

For more details, see "8-3-2 Limitless movement function".

**PRM22: English/Japanese selection**

This parameter sets the language for the response messages displayed on the TPB or handled by RS-232C communications.

Input range: 0 or 1  
Meaning: 0: English  
1: Japanese

Default value: 0
PRM23: *Payload-dependent acceleration coefficient*

The value calculated from PRM0, PRM12 and PRM3 is set automatically for this parameter.

Default value: Depends on robot type.

PRM24: *Teaching count data (TPB entry)*

This is entered in the TPB and cannot be used.

Default value: 0

PRM25: *Not used*

Default value: 0

PRM26: *Teaching movement data*

This parameter is used during movement with a communication command @X+ or @XINC. This is also used for point teaching playback.

Input range: 1 to 100 (%)
Default value: 100

PRM27: *Teaching movement data 1 (for TPB)*

This is entered in the TPB and cannot be used.

Input range: 1 to 100 (%)
Default value: 100

* The TPB writes the contents of PRM27 into PRM26 when connected to the controller.

PRM28: *Teaching movement data 2 (for TPB)*

This is entered in the TPB and cannot be used.

Input range: 1 to 100 (%)
Default value: 50

PRM29: *Teaching movement data 3 (for TPB)*

This is entered in the TPB and cannot be used.

Input range: 1 to 100 (%)
Default value: 10

PRM30: *Maximum program speed*

The speed data defined by the MOVA, MOVI and MOVM statements in a program is multiplied by this parameter value to determine the maximum speed at which the robot actually moves. This is used to lower the speed of the overall program. When the TPB is used, any speed changes in the AUTO and STEP modes will also change this parameter value.

Max. speed (%) = PRM30 × speed operand (%) of movement command / 100

Input range: 1 to 100 (%)
Default value: 100
PRM31: **Open-circuit fault detection level**

This parameter sets the sensitivity for detecting an open-circuit fault. The upper limit of this parameter is 254. The sensitivity lowers as the parameter value increases. Leave this parameter set to 255 if you want to disable this detection function.

- Input range: 1 to 255
- Default value: 255 (This function is disabled.)

PRM32: **Alarm number output**

When an alarm is issued, this parameter selects whether the alarm number is to be output as a general-purpose output. When this parameter is set to 1, the alarm number is output as a 5-bit binary signal through DO0 to DO4.

- Input range: 0 or 1
- Meaning: 0: No output  1: Output
- Default value: 0

**Example of alarm Number - DO output**

<table>
<thead>
<tr>
<th>Alarm No.</th>
<th>Alarm Message</th>
<th>DO4</th>
<th>DO3</th>
<th>DO2</th>
<th>DO1</th>
<th>DO0</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>OVER LOAD</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>02</td>
<td>OVER CURRENT</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>03</td>
<td>OVER HEAT</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>04</td>
<td>POWER DOWN</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>16</td>
<td>ABNORMAL VOLTAGE</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>17</td>
<td>SYSTEM FAULT 2</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>18</td>
<td>FEEDBACK ERROR 3</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>19</td>
<td>SYSTEM FAULT 3</td>
<td>ON</td>
<td>OFF</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

* For more details on the alarm No. and contents, refer to "13-2-2 Alarm message list".

PRM33: **Operation at return-to-origin complete**

Selects the operation to be executed simultaneously with completion of return-to-origin. A signal can be output as a general-purpose output indicating that return-to-origin has been completed or to reset the program.

- Input range: 0 to 3
- Meaning: 0: Nothing is executed  1: DO4 is turned on  2: Program reset is executed  3: DO4 turns on after program reset
- Default value: 2

* When this parameter is set to 1 or 3, DO4 is not affected by program reset (in other words, DO4 does not turn off even when the program is reset). If you want to turn off DO4 after return-to-origin is complete, use the program command to execute DO 4,0 or manually operate the general-purpose output by using the TPB. (See “7-4 Manual Control of General-Purpose Output”.)
PRM34: System mode selection

This parameter specifies the system operation mode. When you want to use the ERCX controller in operating specifications that differ from normal mode, for example, to make it compatible with the conventional controllers, change this parameter as explained below. This parameter functions are allocated in bit units.

Input range: 0 to 255
Default value: 16

### Function allocation in bit units

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
<th>Selected operating mode</th>
<th>Setting</th>
<th>Addition value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>General-purpose input definition for using an I/O point movement command</td>
<td>Normal mode (DI0 to DI11)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Conventional compatible mode (DI0 to DI8)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>READY output sequence setting</td>
<td>DRCA compatible output mode</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SRCA compatible output mode</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>END output sequence setting when the controller has started normally</td>
<td>Normal mode (no to be output)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Conventional compatible mode (not to be output)</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Voltage check setting for system backup battery</td>
<td>Check</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>No check</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Absolute backup function setting</td>
<td>Disable</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enable</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>5 to 6</td>
<td>Reserved for system use</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>END output sequence setting at command execution completion</td>
<td>ON at normal command completion</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON at command signal OFF at normal command completion</td>
<td>1</td>
<td>128</td>
</tr>
<tr>
<td>8 to 15</td>
<td>Reserved for system use</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Example: For conventional compatibility with I/O point movement command general-purpose inputs, and the END sequence at normal controller startups, PRM34 should be set to “21” because of 0000000000010101(binary) = 21(decimal)

Bit 0: General-purpose input definition for using an I/O point movement command

This selects a general-purpose input used for an I/O point movement command (ABS-PT, INC-PT).

In normal mode, use DI0 to DI9 to specify the point number and DI10 to DI11 to select the speed. All points (P0 to P999) can be specified with a movement command.

In conventional compatible mode, use DI0 to DI7 to specify the point number when PRM7 is set to other than 0, and DI8 to select the speed. Points P0 to P254 can be specified with a movement command but points P255 to P999 cannot be selected. If PRM7 is set to 0, use DI0 to DI6 to specify the point number and DI7 to DI8 to select the speed. Points P0 to P127 can be specified with a movement command but points P128 to P999 cannot be selected.

Bit 1: READY signal sequence setting

This selects whether to set the READY signal sequence compatible with the DRCA or SRCA controller.

In DRCA compatible mode, the READY signal turns on at the instant that emergency stop is released. In the SRCA compatible mode the READY signal turns on when the servo is turned on. (The READY signal will not turn on just by releasing emergency stop.)

Bit 2: END signal sequence setting when the controller has started normally

This selects whether to turn on the END signal when the controller has started normally.

In normal mode, the END signal turns on when the controller has started normally. In conventional compatible mode, the END signal remains off even when the controller has started normally.
5-2 Parameter Description

Bit 3: Voltage check setting for system backup battery
This selects whether to check the system backup battery voltage when the controller servo is turned on.
In such cases where you want to operate the robot immediately even when the battery needs to be replaced, you can temporarily disable this voltage check.
(System backup batteries are located inside the controller and used to back up the parameters and point data.)

Bit 4: Absolute backup function setting
This selects whether to enable or disable the absolute backup function.
 Normally, this is set to "enable" and a battery for absolute backup is required. If set to "disable", the controller can be operated without using an absolute backup battery.
 When set to "enable", the robot position is maintained even after the power is turned off. When set to "disable", however, the origin position will be incomplete each time the power is turned off.

Bit 7: END output sequence setting at command execution completion (supported by Ver. 13.74 and later versions):
This selects the END output sequence at dedicated command completion.
With the standard setting ("0"), the command's execution result is output to the END output when the command is completed. When set to "1", the command's execution result is output to the END output when the command is completed, but only after the command signal turns off.

PRM35: Origin shift
This parameter specifies a shift to the origin position after return-to-origin is complete.
When return-to-origin is complete, the origin position is usually "0" (specified value when the mark method is used). If for some reason the origin position needs to be shifted by a particular amount, then change this parameter. For example, if an unwanted position shift occurred, then reteaching of all point data needs to be performed. However, the time and effort needed for this reteaching can be eliminated by setting the shift amount for this parameter to quickly correct the point data.
Input range: \(-9999\) to \(9999\) (0.01mm)
Input range: \(-9999\) to \(9999\) (0.01°)
Default value: 0

* The parameter change is enabled after reperforming return-to-origin.

PRM36: Origin search data
This specifies the performance data for detecting the origin position during return-to-origin by the origin search method.
Default value: Depends on robot type.

PRM37: QP band width
This parameter specifies the control switching point (pulse width) that compensates for the frictional resistance during deceleration.
Input range: 1 to 1000 (pulses)
Default value: Depends on robot type.

PRM38: Speed delay compensation gain
Default value: Depends on robot type.
**5-2 Parameter Description**

**PRM39: No. of motor poles**

**Default value:** Depends on robot type.

**PRM40: RESET execution condition selection**

Selects the operation to be executed with the I/O reset command.

- **Input range:** 0 to 2
- **Meaning:**
  - 0: Turns on the servo and resets the program.
  - 1: Switches the operation depending on the LOCK signal status.
    - When OFF (interlocked), only the servo is turned on.
    - When ON, the servo is turned on and the program is reset.
  - 2: Resets only the program.
- **Default value:** 2

**PRM41: I/O point movement command speed 1**

This parameter specifies the movement speed (%) at which the robot moves when a point movement command (ABS-PT, INC-PT) is executed.

When "type 3" (point teaching type) is selected by the I/O assignment setting, this parameter specifies the jog speed at which the robot moves at a jog movement command (JOG+, JOG-).

This movement speed specified here is the speed used with DI10 turned ON and DI11 turned OFF.

When "type 2" (point number output type) or "type 3" (point teaching type) is selected by the I/O assignment setting and the speed is changeable by SPD1 and SPD2, this parameter specifies the speed when SPD1 is set to ON and SPD2 is set to OFF.

- **Input range:** 1 to 100 (%)
- **Default value:** 10

* The actual speed at which the robot moves with a point movement command (ABS-PT, INC-PT) is the speed obtained by multiplying the execution speed displayed in AUTO or STEP mode by this parameter. (Refer to "4-3-1 Program execution screen".)
  For example, if the execution speed displayed in AUTO or STEP mode is 50 and this parameter is set to 10, then the actual speed will be:
  \[3000 \text{ rpm} \times \frac{50}{100} \times \frac{10}{100} = 150 \text{ rpm} \text{ (when PRM44=3000).}\]
* If this parameter is set to 10, then the jog speed will be:
  \[100 \times \frac{10}{100} = 10 \text{ mm/sec.}\]
**PRM42: I/O point movement command speed 2**

This parameter specifies the movement speed (%) at which the robot moves when a point movement command (ABS-PT, INC-PT) is executed.

When "type 3" (point teaching type) is selected by the I/O assignment setting, this parameter specifies the jog speed at which the robot moves at a jog movement command (JOG+, JOG-).

This movement speed specified here is the speed used with DI10 turned OFF and DI11 turned ON.

When "type 2" (point number output type) or "type 3" (point teaching type) is selected by the I/O assignment setting and the speed is changeable by SPD1 and SPD2, this parameter specifies the speed when SPD1 is set to OFF and SPD2 is set to ON.

Input range: 1 to 100 (%)
Default value: 30

* The actual speed at which the robot moves with a point movement command (ABS-PT, INC-PT) is the speed obtained by multiplying the execution speed displayed in AUTO or STEP mode by this parameter. (Refer to "4-3-1 Program execution screen".)

For example, if the execution speed displayed in AUTO or STEP mode is 50 and this parameter is set to 30, then the actual speed will be:

\[3000 \text{ rpm} \times (50/100) \times (30/100) = 450 \text{ rpm} \text{ (when PRM44=3000).}\]

* If this parameter is set to 30, then the jog speed will be:

\[100 \times 30/100 = 30 \text{ mm/sec} \]

---

**PRM43: I/O point movement command speed 3**

This parameter specifies the movement speed (%) at which the robot moves when a point movement command (ABS-PT, INC-PT) is executed.

When "type 3" (point teaching type) is selected by the I/O assignment setting, this parameter specifies the jog speed at which the robot moves at a jog movement command (JOG+, JOG-).

This movement speed specified here is the speed used with DI10 and DI11 turned ON.

When "type 2" (point number output type) or "type 3" (point teaching type) is selected by the I/O assignment setting and the speed is changeable by SPD1 and SPD2, this parameter specifies the speed when both SPD1 and SPD2 are set to ON.

Input range: 1 to 100 (%)
Default value: 70

* The actual speed at which the robot moves with a point movement command (ABS-PT, INC-PT) is the speed obtained by multiplying the execution speed displayed in AUTO or STEP mode by this parameter. (Refer to "4-3-1 Program execution screen".)

For example, if the execution speed displayed in AUTO or STEP mode is 50 and this parameter is set to 70, then the actual speed will be:

\[3000 \text{ rpm} \times (50/100) \times (70/100) = 1050 \text{ rpm} \text{ (when PRM44=3000).}\]

* If this parameter is set to 70, then the jog speed will be:

\[100 \times 70/100 = 70 \text{ mm/sec} \]
PRM44: Maximum speed setting
This parameter sets the maximum motor revolution speed.

Input range: 1 to 4500 (rpm)
Default value: Depends on robot type.

⚠️ CAUTION
Changing this parameter carelessly might shorten the robot service life or cause other problems.

PRM45: Feed forward gain
Default value: Depends on robot type.

PRM46: Servo status output
This parameter selects whether to output the axis servo status as a general-purpose output.
When this parameter is set to 1, DO7 turns on and off along with servo on/off.

Input range: 0 or 1
Meaning: 0: Does not output the servo status.
1: Outputs the servo status.
Default value: 0

* When this parameter is set to 1, DO7 is not affected by program reset (in other words, DO7 does not turn off even when the program is reset).

PRM47: Communication parameter setting
This sets communication parameters used for data transmission through RS-232C. For more details, see "11-1 Communication Parameter Specifications".

Default value: 0

PRM48: Pre-operation action selection
This parameter checks whether return-to-origin has been performed or resets the program before running automatic operation or step operation.
When set to 0 or 2, an error (return-to-origin incomplete) is issued if return-to-origin has not been performed and automatic operation and step operation are not accepted.
When set to 1 or 3, the program runs even when return-to-origin has not been performed. However, an error (return-to-origin incomplete) is issued when a movement command (MOVA, etc.) is executed if return-to-origin is still incomplete. To avoid this, perform return-to-origin in advance or insert the ORGN command into the program.

Input range: 0 to 3
Meaning: 0: Checks whether return-to-origin has been performed.
1: Nothing is executed.
2: Resets the program after checking return-to-origin.
3: Resets the program.
Default value: 1

* When set to 2 or 3, the program is reset only during automatic operation. (The program is not reset during step operation.)
PRM49: Controller version 1

This parameter reads out the version information (1) on the control software in the controller.
This is a read-only parameter.

PRM50: Deceleration (Available with Ver. 13.33 or later)

Use this parameter to reduce only the deceleration.
When this parameter is left set to the default value (100), the deceleration is the same as the acceleration. If vibration occurs during positioning, then set this parameter to a smaller value to reduce only the deceleration.
This parameter value can be changed in 1% steps, with 100% equal to the value determined by PRM4.

Input range: 1 to 100 (%)
Default value: 100

PRM51: Lead program number (Available with Ver. 13.50 or later)

This parameter sets the lead program number.

Default value: 0

NOTE
The lead program is the program that has been selected as the execution program by the TPB or POPCOM. (See "9-4 Switching the Execution Program").
The lead program can also be selected by executing a communication command "@SWI". It may also be switched when the program data is loaded into the controller from the memory card.

PRM52: Hold gain (Available with Ver. 13.50 or later)

Default value: Depends on the robot.
PRM53: Zone output selection (Available with Ver. 13.50 or later)

This parameter is used to select the output destination and output logic when the zone output function is enabled. The zone output is used to control the signal output when the robot’s current position is within the specified range.

A maximum of 4 zone outputs are available by setting for PRM53. The output logic can also be changed.

This parameter functions are allocated in bit units.

Input range: 0 to 255
Default value: 0

Function allocation in bit units

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
<th>Selected value</th>
<th>Addition value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Zone 0 output enable setting</td>
<td>0: Disabled</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Enabled</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>Zone 1 output enable setting</td>
<td>0: Disabled</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Enabled</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Zone 2 output enable setting</td>
<td>0: Disabled</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Enabled</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Zone 3 output enable setting</td>
<td>0: Disabled</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Enabled</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Zone 0 output logic setting</td>
<td>0: Positive logic</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Negative logic</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>Zone 1 output logic setting</td>
<td>0: Positive logic</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Negative logic</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>Zone 2 output logic setting</td>
<td>0: Positive logic</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Negative logic</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>Zone 3 output logic setting</td>
<td>0: Positive logic</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Negative logic</td>
<td>128</td>
</tr>
<tr>
<td>8 to 15</td>
<td>Reserved for system use</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

Example: To set zone 1 output to positive logic and zone 2 output to negative logic while enabling zone 1 output and zone 2 output, make the following settings.
PRM53 should be set to “70” because of 0000000001000110 (binary)=70 (decimal).

Zone output function

To use the zone output function, the desired zone must be specified with point data. (See Chapter 7, “EDITING POINT DATA”.) When the robot enters the specified zone, its result is output to the specified port. Point numbers and output port that can be used for each zone output are listed below.

Zone setting range and output port

<table>
<thead>
<tr>
<th>Zone No.</th>
<th>Specified range</th>
<th>Output port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zone 0</td>
<td>P900-P901</td>
<td>DO0</td>
</tr>
<tr>
<td>Zone 1</td>
<td>P902-P903</td>
<td>DO1</td>
</tr>
<tr>
<td>Zone 2</td>
<td>P904-P905</td>
<td>DO2</td>
</tr>
<tr>
<td>Zone 3</td>
<td>P906-P907</td>
<td>DO3</td>
</tr>
</tbody>
</table>

⚠️ CAUTION

The zone output function does not work if one item of the point data is unspecified or return-to-origin is incomplete.
PRM54: Not used (Available with Ver. 13.50 or later)

Default value: 0

PRM55: Not used (Available with Ver. 13.50 or later)

Default value: 0

PRM56: Controller version 2 (Available with Ver. 13.50 or later)

This parameter reads out the version information (2) on the control software in the controller.
This is a read-only parameter.

PRM57: Not used

Default value: 0

CAUTION
Do not change the setting.

PRM58: Not used

Default value: 0
PRM59: I/O assignment selection (Available with Ver. 13.57 or later)

This parameter selects the function to be assigned to each I/O signal. This parameter setting allows changing the function assigned to each I/O signal. This makes it possible to output the destination point number and perform jog movement. After changing the I/O assignment, the ERCX controller must be restarted to enable the changes. For more details, see "3-7 I/O assignment change function".

Input range: 0 or another number (See the I/O assignment list)

Meaning: \[
\text{PRM59} = \begin{pmatrix} \text{X} \\ \text{X} \\ \text{X} \\ \text{X} \\ \text{\uparrow\uparrow} \\ \text{\uparrow} \end{pmatrix}
\]

1. I/O assignment type selection
   - 00 : Type 0 (Conventional type/standard)
   - 20, 21 : Type 2 (Point number output type)
   - 30, 31 : Type 3 (Point teaching type)
   * Type 1 cannot be used with the ERCX controller.

2. Point output selection
   Make setting only for type 2 (point number output type) and type 3 (point teaching type).
   - 0: Outputs when movement ends normally.
   - 1: Outputs when movement command is received.
   - 2: Point zone output
     PO output occurs when the robot enters the point data (registered at controller) ± position judgment parameter range.
   - 3: Movement point zone output (supported in Ver. 13.64 and later versions):
     PO output occurs when the robot enters the ± position judgment parameter range for the point data registered at the controller and which serves as the movement point data for the point movement command (ABS-PT, INC-PT).

3. Point zone judgment method selection (supported in Ver. 13.64 and later versions):
   The position judgment parameter is selected when the point output selection is "2" (point zone output) or "3" (movement point zone output).
   - 0: OUT valid position
   - 1: Positioning-completed pulse

Default value: 0

⚠️ CAUTION

Any value other than the above is handled as a "0" (type 0).
Example: If set to 2331, this is handled as a "0" (type 0).
   If set to 10, this is handled as a "0" (type 0).
Moreover, if Type 2 (point signal output type) or Type 3 (point teaching type) is selected in ERCX versions prior to Ver. 13.64, with the point output selection specified as "3", this is processed as a "0" (Type 0) setting.
Example: At ERCX version prior to Ver. 13.64:
   If set to 331, this is handled as a "0" (type 0).

💡 NOTE

In controller versions prior to Ver. 13.64, the "OUT valid position" is the only point zone judgment method.
### I/O assignment list

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type 0</td>
<td>ABS-PT</td>
<td>INC-PT</td>
<td>AUTO-R</td>
<td>STEP-R</td>
<td>ORG-S</td>
<td>ORG-S</td>
<td>SERVO</td>
<td>LOCK</td>
<td>D10</td>
<td>D11</td>
<td>D12</td>
<td>D13</td>
<td>D14</td>
</tr>
<tr>
<td>Type 1</td>
<td>ABS-PT</td>
<td>INC-PT</td>
<td>-</td>
<td>-</td>
<td>ORG-S</td>
<td>ORG-S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Type 2</td>
<td>ABS-PT</td>
<td>INC-PT</td>
<td>PSET</td>
<td>CHG</td>
<td>ORG-S</td>
<td>ORG-S</td>
<td>SERVO</td>
<td>LOCK</td>
<td>P00</td>
<td>P01</td>
<td>P02</td>
<td>P03</td>
<td>P04</td>
</tr>
<tr>
<td>Type 3</td>
<td>JOG+</td>
<td>INC-PT</td>
<td>-</td>
<td>CHG</td>
<td>ORG-S</td>
<td>ORG-S</td>
<td>SERVO</td>
<td>LOCK</td>
<td>P00</td>
<td>P01</td>
<td>P02</td>
<td>P03</td>
<td>ORG-O-ZONE0</td>
</tr>
</tbody>
</table>

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Setting</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRM59</td>
<td>(Standard)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>No. of points</td>
<td>1000</td>
<td>64</td>
<td>16</td>
</tr>
<tr>
<td>No. of speed switching points</td>
<td>4</td>
<td>None</td>
<td>4</td>
</tr>
<tr>
<td>Program operation by I/O</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

### Notes

1. The P0 output format varies depending on whether the PRM59 setting is specified in "hundreds" or "thousands" units.
2. Specifies the permissible number of movement points for a point movement command (ABS-PT, INC-PT).
3. Specifies the permissible number of speed switching points for a point movement command (ABS-PT, INC-PT).
In this chapter we will try programming some operations. First, you will learn how to enter a program using the TPB programming box.
6-1  Basic Contents

6-1-1  Robot language and point data

The ERCX controller uses the YAMAHA robot language that is very similar to BASIC. It allows you to easily create programs for robot operation.

In programs created with the YAMAHA robot language, the robot position data (absolute position, amount of movement) are not expressed in terms of direct numeric values. Instead, point numbers are used to express the position data indirectly. Point numbers and their corresponding position information are stored as point data separately from programs. This means that when you want to change only the position information while using the same program, all that you have to do is edit the point data.

Example Program Point Data:

<table>
<thead>
<tr>
<th>Program</th>
<th>Point Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>005: MOV A 0, 100</td>
<td>P0 = 50.00</td>
</tr>
<tr>
<td>006: MOVI 1, 50</td>
<td>P1 = 100.00</td>
</tr>
</tbody>
</table>

In the above example, the robot first moves to a position (P0) 50mm from the origin point, and then moves to another point (P1) 100mm away from that position.

To change the above operation so that the robot first moves to a position (P0) 50.5mm from the origin point and then moves to another point (P1) 100mm away from that position, just change the P0 point data to P0=50.50.

6-1-2  Using the TPB to enter the robot language

Robot language commands frequently used to create programs are printed on the lower part of each number key on the TPB. When creating or editing a program, you can enter robot language commands simply by pressing these keys. To select other robot language commands not printed on these keys, use the function key matching that command.

During program editing, you can enter numbers (numerical values) with the number keys except when the edit cursor for robot language command input appears on the TPB screen.

6-1-3  Program specifications

The ERCX controller has the following memory capacity:

- Total number of programs: 100 programs (NO0 to NO99)
- Max. number of steps per program: 255 steps
- Max. number of steps in all programs together: 3000 steps
- Max. number of points: 1000 points (P0 to P999)
6-2 Editing Programs

"Program editing" refers to operations such as creating a program right after initialization, creating a new program, changing an existing program, and deleting or copying a program. In this section, you will learn the basic procedures for program editing using the TPB.

"Creating a program right after initialization" means creating a program for the first time after purchasing the controller or creating a program right after initialization while there are still no programs stored in the controller (see "10-1 Initialization").

"Creating a new program" means creating or editing a new program while at least one program has already been created and stored.

"Changing an existing program" means correcting, adding, deleting, or inserting steps in a program to change only part of it.

This section explains all the above program editing procedures, and also describes how to view program information such as the number of steps left in a program.

- Creating a program right after initialization
  6-2-1 Creating a program (right after initialization) ............ 6-4

- Creating a new program
  6-2-2 Creating a new program ............................................. 6-6

- Changing an existing program
  6-2-3 Adding a step ............................................................. 6-7
  6-2-4 Correcting a step ....................................................... 6-9
  6-2-5 Inserting a step ......................................................... 6-10
  6-2-6 Deleting a step ......................................................... 6-11

- Copying a program
  6-3-1 Copying a program ................................................... 6-12

- Deleting a program
  6-3-2 Deleting a program ................................................... 6-13

- Viewing the program information
  6-3-3 Viewing the program information ............................... 6-14
6-2-1 Creating programs after initialization

1) On the initial screen, press [F1] (EDIT).

2) Next, press [F1] (PGM).

3) Since no program is registered after initialization, an error message appears on the screen, indicating that no program exists.

4) Press the [ESC] key to reset the error. A confirmation message then appears asking whether to create a new program as program No. 0. To select and edit program No. 0, press [F1] (yes). To select and edit a program other than No. 0, press [F2] (no).

5) When you selected [F2] (no) in step 4, enter the number of the program to be edited with the number keys and press [ENT]. The screen returns to step 4. Make sure the program number is correct and press [F1] (yes).

6) Select [F1] to [F3] or a robot language command shown on the lower part of each number key. To change the robot language menu display, press [F4] (next). To go back to the previous menu display, press the [BS] key.
7) After selecting the robot language command, enter the operand data.
When you press $\text{XZ}$, the cursor moves to operand 1, so enter the data with the number keys.
(Do not press $\rightarrow$ at this point.)
While pressing $\text{XZ}$ or $\text{XZ}$ to move the cursor, enter all necessary operand data as needed.

8) After entering the operand data, press $\rightarrow$.

9) When entry is completed correctly, the cursor moves to the operation code part.
To edit the next step, press $\text{STEP UP}$ to scroll the step and repeat the procedure from step 6.
6-2-2 Creating a new program

1) On the initial screen, press [F1] (EDIT).

2) Next, press [F1] (PGM).

3) The execution program number and step are displayed on the screen. Press [F4] (CHG) here.

4) Enter the new program number with the number keys and press →.

5) A confirmation message appears. Make sure the program number is correct and press [F1] (yes).

6) Proceed with program editing by following step 6 onward in “6-2-1 Creating programs after initialization.”
6-2-3 Adding a step

1) On the initial screen, press [F1] (EDIT).

2) Next, press [F1] (PGM).

3) The execution program number and step are displayed on the screen. Press [F4] (CHG) here.

4) Enter the program number you want to edit with the number keys and press →.

5) Enter the last step number with the number keys and press →.

6) When the last step is displayed, press [STEP UP].
7) Select [F1] to [F3] or a robot language command shown on the lower part of each number key.

To change the robot language menu display, press [F4] (next). To go back to the previous menu display, press the [BS] key.

8) After selecting the robot language command, enter the operand data.

When you press [X Z +], the cursor moves to operand 1, so enter the data with the number keys.

(Do not press [X Z –] at this point.)

While pressing [X Z +] or [X Z –] to move the cursor, enter all necessary operand data as needed.

9) After entering the operand data, press [EXEC].

10) When the program has been edited correctly, the screen returns to step 6.

When you want to add another step, press [STEP UP] to scroll to the next step and then repeat from step 7.
6-2-4 Correcting a step

1) Use the same procedure up to step 4 in "6-2-3 Adding a step".

2) Enter the number of the step you want to correct with the number keys and press \[EDIT-PGM\].

3) Press \[F1\] (MOD).

4) Select \[F1\] to \[F3\] or a robot language command shown on the lower part of each number key.
   To change the robot language menu display, press \[F4\] (next). To go back to the previous menu display, press the \[BS\] key.

5) After selecting the robot language command, enter the operand data.
   When you press \[X\] or \[Z\], the cursor moves to operand 1, so enter the data with the number keys.
   (Do not press \[\rightarrow\] at this point.)
   While pressing \[X\] or \[Z\] to move the cursor, enter all necessary operand data as needed.

6) After entering the operand data, press \[\rightarrow\].

7) When entry is completed correctly, the cursor moves to the operation code part.
   If you want to change another step, press \[STEP\] to scroll the step and repeat the procedure from step 4.
6-2-5 Inserting a step

1) Use the same procedure up to step 4 in "6-2-3 Adding a step".

2) Enter the number of the step where you want to insert a step with the number keys and press \[ \text{INS} \].

3) Press \( F_2 \) (INS).

4) Select \( F_1 \) to \( F_3 \) or a robot language command shown on the lower part of each number key.
   To change the robot language menu display, press \( F_4 \) (next). To go back to the previous menu display, press the \[ \text{BS} \] key.

5) After selecting the robot language command, enter the operand data.
   When you press \[ \text{INS} \], the cursor moves to operand 1, so enter the data with the number keys. (Do not press \[ \text{INS} \] at this point.)
   While pressing \[ \text{INS} \] or \[ \text{BS} \] to move the cursor, enter all necessary operand data as needed.

6) After entering the operand data, press \[ \text{INS} \].

7) When entry is completed correctly, the screen returns to step 3.
6-2-6 Deleting a step

1) Use the same procedure up to step 4 in "6-2-3 Adding a step".

2) Enter the number of the step you want to delete with the number keys and press [EDIT-PGM].


5) When the step has been deleted, the screen returns to step 3.
6-3 Program Utility

6-3-1 Copying a program

1) On the initial screen, press [F1] (EDIT).

   [MENU]
   select menu
   EDITOPRTSYS MON

   [EDIT]
   select menu
   PGM PNT UTL

2) Next, press [F3] (UTL).

3) Press [F1] (COPY).

4) Enter the program number you want to copy from with the number keys, and then press →.

   [EDIT-UTL-COPY]
   Copy from No = _
   (Program No) 0 → 99

5) Enter the program number you want to copy to with the number keys, and then press →.

   [EDIT-UTL-COPY]
   Copy from No = 0
   Copy to No = 99_
   (Program No) 0 → 99
6) If program data is already registered with the selected program number, a confirmation message appears.
   To overwrite the program, press [F1] (yes).
   To cancel, press [F2] (no).

7) When the program has been copied, the screen returns to step 3.

6-3-2 Deleting a program

1) Use the same procedure up to step 2 in "6-3-1 Copying a program".

2) Press [F2] (DEL).

3) Enter the number of the program you want to delete with the number keys and press .

4) A confirmation message appears asking whether to delete the selected program.
   To delete the program, press [F1] (yes).
   To cancel the deletion, press [F2] (no).

5) If the program has been deleted, the screen returns to step 2.
6-3-3 Viewing the program information

1) Use the same procedure up to 2 in "6-3-1 Copying a program".

2) Press $\text{F3}$ (LIST).

3) The program numbers are displayed on the screen, along with the number of registered steps and the number of available remaining steps.

   To view other program information, press the $\text{STEP UP}$ and $\text{STEP DOWN}$ keys to scroll the screen.

4) Press the $\text{ESC}$ key to return to the screen of step 2.

   * In addition to the number of existing steps, the steps equivalent to the number of programs are used internally as the program control steps. For example, if two programs are registered and their respective 50 and 100 steps are registered, then the number of available remaining steps will be as follows:

   \[
   3000 - 2 - 50 - 100 = 2848 \text{ steps}
   \]
There are three methods to enter point data: manual data input (MDI), teaching playback, and direct teaching. Manual data input allows you to directly enter point data with the TPB number keys. Teaching playback moves the robot in manual operation to a desired position and then obtains that position as point data. Direct teaching is basically the same as teaching playback, except that you move the robot by hand.
7-1 Manual Data Input

1) On the initial screen, press [F1] (EDIT).

2) Next, press [F2] (PNT).

3) Press [F1] (MDI).

4) The currently selected point data in the execution program appears on the screen. If you want to edit another point data, press the STEP UP and STEP DOWN keys to scroll the point data. To directly select the point data, press [F1] (CHG).

5) Enter the point number you want to edit with the number keys, and press .

6) Enter the point data with the number keys and press .

7) The input data is then registered as point data.
7-2  Teaching Playback

1) On the initial screen, press [F1] (EDIT).

2) Next, press [F2] (PNT).


4) The currently selected point data in the execution program appears on the screen. If you want to edit another point data, press the STEP UP and STEP DOWN keys to scroll the point data. To directly select the point data, press [F1] (CHG).

5) Enter the point number you want to edit with the number keys, and press ➔.
6) Move the robot to the teaching position with the \( \text{X} \text{Z}^- \) or \( \text{X} \text{Z}^+ \) keys. Each time the \( \text{X} \text{Z}^- \) or \( \text{X} \text{Z}^+ \) key is pressed, the robot moves a certain amount in the direction indicated by the key and then stops.

Holding down the \( \text{X} \text{Z}^- \) or \( \text{X} \text{Z}^+ \) key moves the robot continuously at a constant speed until the key is released.

The amount of robot movement and the speed are proportional to the number (teaching movement data) displayed on the upper right of the screen.

In the example at the right, the teaching movement data is 50 (\%), so the robot moves 0.5mm each time the \( \text{X} \text{Z}^- \) or \( \text{X} \text{Z}^+ \) key is pressed, as calculated below:

\[
1\text{mm (constant)} \times \left( \frac{50}{100} \right) = 0.5\text{mm}
\]

If the \( \text{X} \text{Z}^- \) or \( \text{X} \text{Z}^+ \) key is kept pressed, the robot continuously moves at a speed of 50mm/s, as calculated below:

\[
100\text{mm/s (constant)} \times \left( \frac{50}{100} \right) = 50\text{mm/s}
\]

7) Three different speed settings, SPEED (1), SPEED (2), and SPEED (3), are selectable as the teaching movement data. Each time \( \text{SPD} \) is pressed, the setting changes in the order of 1→2→3→1.

To change the teaching movement data setting, press \( \text{S_SET} \), enter the desired speed with the number keys, and press \( \text{SPD} \). The screen then returns to step 6 when the data has been changed correctly.

8) Move the robot to the teaching position in this way and press \( \text{SPD} \). The current position is input as point data.

**CAUTION**

When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function").

- Robot movement speed is limited to 10mm/s or less (10 deg/s for rotary robot) in "SERVICE mode state" when the robot movement speed limit is enabled.
7-3 Direct Teaching

1) On the initial screen, press [F1] (EDIT).

2) Next, press [F2] (PNT).


4) Following the message, press the emergency stop button on the TPB.

5) The currently selected point data in the execution program appears on the screen.
   If you want to edit another point data, press the [STEP] and [STEP] keys to scroll the point data.
   To directly select the point data, press [F1] (CHG).

6) Enter the point number you want to edit with the number keys, and press →.

[EDIT-PNT-DTCH]
press EMG.button

[EDIT-PNT-DTCH]
P0 = 0.00 [mm]
[ 0.00]

[EDIT-PNT-DTCH]
Pn : n = _
(point No) 0→999
7) Move the robot to the teaching position by hand.

8) Press \( \Rightarrow \) to input the current position as point data. Use the same procedure to input all other necessary point data, and then press the \( \text{ESC} \) key.

9) Following the message, release the emergency stop button on the TPB.

10) A confirmation message appears asking whether to turn the servo on. To turn the servo on, press \( \text{F1} \) (yes). To leave the servo off, press \( \text{F2} \) (no).

11) The screen returns to step 3.
7-4 Manual Control of General-Purpose Output

When performing teaching playback or direct teaching with systems that use a general-purpose output through the I/O interface to operate a gripper or other tools, you may want to check the position of workpiece by actually moving it. For this reason, the ERCX controller is designed to allow manual control of general-purpose outputs from the TPB.

1) Move the robot with the same procedure up to step 6 in "7-2 Teaching Playback" or up to step 7 in "7-3 Direct Teaching".
   The following steps are explained using the teaching playback screen.

2) When the robot reaches the position where you want to operate general-purpose output, stop the robot. Then press [F4](next) to change the menu display and then press [F1](DO).

3) The current status of the general-purpose output appears on the screen.
   Press the function key that matches the DO number to switch the output on and off (on=1, off=0).

   If selecting DO3 to DO12, press [F4](next) a few times to change the menu display.

4) Press [ESC] to return to step 2.
7-5 Manual Release of Holding Brake

The holding brake on the vertical type robot can be released. Since the movable part will drop when the brake is released, attaching a stopper to protect the tool tip from being damaged is recommended.

1) Use the same procedure up to step 4 in "7-3 Direct Teaching".

2) Press \[F3\] (BRK).

3) A confirmation message appears asking whether to release the brake.
   To release the brake, press \[F1\] (yes).
   To cancel releasing the brake, press \[F2\] (no).

4) The screen returns to step 2.
   The brake stays released until \[F3\] (BRK) is pressed again or the robot servo is turned on.
7-6 Deleting Point Data

1) Use the same procedure up to step 2 in "7-1 Manual Data Input".


3) Enter the point number at the start to delete point data with the number keys and press 🔄.

4) Enter the point number at the end to delete point data with the number keys and press 🔄.

5) A confirmation message appears asking whether to delete the data.
   To delete the data, press [F1] (yes).
   To cancel the deletion, press [F2] (no).

6) When the point data has been deleted, the screen returns to step 2.
7-7 Tracing Points (Moving to a registered data point)

The robot can be moved to the position specified by a registered data point. You can check the input point data by actually moving the robot.

1) Use the same procedure up to step 5 in "7-2 Teaching Playback".

2) Press [F4] (next) to change the menu display and then press [F2] (TRC).

3) The coordinate data of the movement destination and the movement speed are displayed. To move the robot, press [F1] (yes). To cancel moving the robot, press [F2] (no).

The movement speed will be 10% of the number (speed parameter) displayed at the upper right of the screen.

4) When the movement is completed, the screen returns to step 2.

⚠️ CAUTION
When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function").
- Robot movement speed is limited to 3% or less of maximum speed in "SERVICE mode state" when the robot movement speed limit is enabled.
- If the hold-to-run function is enabled, robot movement stops upon releasing [F1] (yes) in "SERVICE mode state". (You must hold down [F1] (yes) in step 3 until the robot reaches the target point.)
This chapter explains the robot language. It describes what kind of commands are available and what they mean. The ERCX series uses the YAMAHA robot language. This is an easy-to-learn BASIC-like programming language. Even a first-time user can easily create programs to control complex robot and peripheral device movements. This robot language is an upper version of the SRC, SRCA, ERC and SRCH series robot language but fully compatible with it, so that the previous programs used with these conventional controllers can be easily upgraded. At the beginning of this chapter, you will find a convenient table of robot language commands. At the end of this chapter, sample programs are listed for just your reference.
# 8-1 Robot Language Table

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description and Format</th>
<th>Applicable version</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVA</td>
<td>Moves to point data position. MOVA &lt;point number&gt;, &lt;maximum speed&gt;</td>
<td></td>
</tr>
<tr>
<td>MOVI</td>
<td>Moves from current position by amount of point data. MOVI &lt;point number&gt;, &lt;maximum speed&gt;</td>
<td></td>
</tr>
<tr>
<td>MOVF</td>
<td>Moves until specified DI input is received. MOVF &lt;point number&gt;, &lt;DI number&gt;, &lt;DI status&gt;</td>
<td></td>
</tr>
<tr>
<td>JMP</td>
<td>Jumps to a specified label in a specified program. JMP &lt;label number&gt;, &lt;program number&gt;</td>
<td></td>
</tr>
<tr>
<td>JMPF</td>
<td>Jumps to a specified label in a specified program according to the input condition. JMPF &lt;label number&gt;, &lt;program number&gt;, &lt;input condition&gt;</td>
<td></td>
</tr>
<tr>
<td>JMPB</td>
<td>Jumps to a specified label when general-purpose input or memory input is in the specified state. JMPB &lt;label number&gt;, &lt;DI or MI number&gt;, &lt;input status&gt;</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>Defines the jump destination for a JMP or JMPF statement, etc. L &lt;label number&gt;</td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>Runs another program. CALL &lt;program number&gt;, &lt;number of times&gt;</td>
<td></td>
</tr>
<tr>
<td>DO</td>
<td>Turns general-purpose output or memory output on and off. DO &lt;DO or MO number&gt;, &lt;output status&gt;</td>
<td></td>
</tr>
<tr>
<td>WAIT</td>
<td>Waits until a general-purpose input or memory input is set in the specified state. WAIT &lt;DI or MI number&gt;, &lt;input status&gt;</td>
<td></td>
</tr>
<tr>
<td>TIMR</td>
<td>Waits the specified amount of time before advancing to the next step. TIMR &lt;time&gt;</td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>Defines a point variable. P &lt;point number&gt;</td>
<td></td>
</tr>
<tr>
<td>P+</td>
<td>Adds 1 to a point variable. P+</td>
<td></td>
</tr>
<tr>
<td>P-</td>
<td>Subtracts 1 from a point variable. P-</td>
<td></td>
</tr>
<tr>
<td>SRVO</td>
<td>Turns a servo on and off. SRVO &lt;servo status&gt;</td>
<td></td>
</tr>
<tr>
<td>STOP</td>
<td>Temporarily stops program execution. STOP</td>
<td></td>
</tr>
<tr>
<td>ORGN</td>
<td>Performs return-to-origin. ORGN</td>
<td></td>
</tr>
<tr>
<td>TON</td>
<td>Runs a specified task. TON &lt;task number&gt;, &lt;program number&gt;, &lt;start type&gt;</td>
<td></td>
</tr>
<tr>
<td>TOFF</td>
<td>Stops a specified task. TOFF &lt;task number&gt;</td>
<td></td>
</tr>
<tr>
<td>JMPP</td>
<td>Jumps to a specified label when the axis position relation meets the specified conditions. JMPP &lt;label number&gt;, &lt;axis position condition&gt;</td>
<td></td>
</tr>
<tr>
<td>MAT</td>
<td>Defines a matrix. MAT &lt;number of rows&gt;, &lt;number of columns&gt;, &lt;pallet number&gt;</td>
<td></td>
</tr>
<tr>
<td>MSEL</td>
<td>Specifies a matrix to move. MSEL &lt;pallet number&gt;</td>
<td></td>
</tr>
<tr>
<td>MOVM</td>
<td>Moves to a specified pallet work position on matrix. MOVM &lt;pallet work position&gt;, &lt;maximum speed&gt;</td>
<td></td>
</tr>
<tr>
<td>J MPC</td>
<td>Jumps to a specified label when counter array variable C equals the specified value. J MPC &lt;label number&gt;, &lt;counter value&gt;</td>
<td></td>
</tr>
<tr>
<td>J MPD</td>
<td>Jumps to a specified label when counter variable D equals the specified value. J MPD &lt;label number&gt;, &lt;counter value&gt;</td>
<td></td>
</tr>
<tr>
<td>C S EL</td>
<td>Specifies the array element of counter array variable C. C S EL &lt;array element number&gt;</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>Defines counter array variable C. C counter value&gt;</td>
<td></td>
</tr>
<tr>
<td>C+</td>
<td>Adds a specified value to counter array variable C. C+ [&lt;addition value&gt;]</td>
<td></td>
</tr>
<tr>
<td>C-</td>
<td>Subtracts a specified value from counter array variable C. C- [&lt;subtraction value&gt;]</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Defines counter variable D. D counter value&gt;</td>
<td></td>
</tr>
<tr>
<td>D+</td>
<td>Adds a specified value to counter variable D. D+ [&lt;addition value&gt;]</td>
<td></td>
</tr>
<tr>
<td>D-</td>
<td>Subtracts a specified value from counter variable D. D- [&lt;subtraction value&gt;]</td>
<td></td>
</tr>
<tr>
<td>SHFT</td>
<td>Shifts the coordinate position by a distance equal to specified point data. SHFT &lt;point number&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Values in brackets [ ] can be omitted. Available with Ver. 13.23 or later
8-2 Robot Language Syntax Rules

8-2-1 Command statement format

The robot language command statement format for the ERCX controller is as follows. When creating a program using the TPB, each command statement can be automatically entered in this format, so you do not have to be aware of this format while creating the program.

<operation code>   [<operand 1>],[<operand 2>],[<operand 3>]   [:<comment>]

- A command statement is basically composed of an operation code and an operand. Depending on the command statement, either no operand is used, or up to three operands are used. A comment can be written following the operand. (But, no comment can be written with the TPB.) A line consisting of only a comment cannot be created. Items in [ ] (brackets) can be omitted.

- A command statement must be entered with one-byte characters (alphanumeric characters, special characters) except for comment. Input characters can be upper case or lower case. The controller automatically converts the input characters to upper case.

- One command statement must be described within one line. It cannot be written over multiple lines. Multiple command statements cannot be described on one line. Up to 80 one-byte characters (including carriage line return) can be described on one line.

- One or more spaces must be inserted between the operation code and the operand.

- Operands enclosed in < > marks must be specified by the user. Check the description of each robot language and enter the appropriate data. (Refer to "8-4 Robot Language Description").

- When two or more operands are entered, insert a comma (,) between them.

- Any entry after a semicolon (;) is recognized as a comment. When creating a program using a PC (personal computer), a comment is helpful to easily identify the program. Note, however, that the comment is not stored in the controller. A comment can be any number of characters as long as it is within one line. Characters that can be used as a comment are one-byte characters (alphanumeric characters, special characters) and two-byte characters (full space characters).
8-2-2 Variables

Variables are used in a program to hold data. The following variables can be used with the ERCX controller.

■ Point variable P
A point variable can contain a point number. It is used in movement commands such as MOVA and MOVI statements instead of specifying the point number directly. Sometimes the number of program steps can be reduced by using point variables.

■ Counter array variable C, Counter variable D
A counter variable can contain counter values and is used to specify the pallet work position number in a palletizing program and to count the number of runs. A counter array variable is an array of a total of 32 counter variables that can be selected by the CSEL statement of robot language.

■ Flag variable: memory input/output 100 to 147
A flag variable can only have a data value of 1 (ON) or 0 (OFF). It is used in a multi-task program to synchronize between tasks or in a condition judgement program.

Memory I/O from 100 to 131 can be freely turned on or off by the user or their values can be referenced. However, outputs 132 to 147 are controlled by the system so the user can only refer to their values.

Memory I/O description

<table>
<thead>
<tr>
<th>Type</th>
<th>Memory I/O No.</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>General-purpose</td>
<td>100 to 131</td>
<td>Memory I/O available to the user</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The user can freely set this with a DO statement.</td>
</tr>
<tr>
<td>Dedicated</td>
<td>132</td>
<td>Task 0 (main task) status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Always set to 1.</td>
</tr>
<tr>
<td></td>
<td>133</td>
<td>Task 1 status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Task has started. 0: Task has ended or has not yet started.</td>
</tr>
<tr>
<td></td>
<td>134</td>
<td>Task 2 status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Task has started. 0: Task has ended or has not yet started.</td>
</tr>
<tr>
<td></td>
<td>135</td>
<td>Task 3 status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Task has started. 0: Task has ended or has not yet started.</td>
</tr>
<tr>
<td></td>
<td>136 to 139</td>
<td>Reserved for system use (Always set to 0.)</td>
</tr>
<tr>
<td></td>
<td>140</td>
<td>X-axis hold status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Hold 0: Non hold</td>
</tr>
<tr>
<td></td>
<td>141 to 143</td>
<td>Reserved for system use (Always set to 0.)</td>
</tr>
<tr>
<td></td>
<td>144</td>
<td>X-axis constant movement status</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: Constantly moving 0: Accelerating, decelerating or in stop</td>
</tr>
<tr>
<td></td>
<td>145 to 147</td>
<td>Reserved for system use (Always set to 0.)</td>
</tr>
</tbody>
</table>
8-3 Program Function

8-3-1 Multi-task function

A multi-task function allows simultaneous executing two or more programs (tasks). The ERCX controller can execute a maximum of 4 programs at the same time. Since the multi-task function simultaneously executes two or more programs, the following processing can be performed.

- Other processing can be performed during robot movement.
  For example, a general-purpose output can be turned on or off while a robot movement command such as MOV A or MOVI statement is being executed. This reduces the cycle time.

A multi-task program can be written by the same method as normal programs. A TON statement used as the task start command is written into the main program and the subtask program is registered as another program number. When the TON command is processed during the program execution, the subtask starts to perform multiple tasks. The subtask will end when its last step has been executed or the TOFF command is issued.

Each task and data have the following relation.
Point variables P in a task are independent of those in other tasks.
Point data, general-purpose I/O and memory I/O are shared with each task.

⚠️ CAUTION
In addition to the tasks (up to 4 tasks) specified by the user, the system task starts inside the controller, so a maximum of 5 tasks are executed.
In general, the multi-task is defined as a function that simultaneously executes two or more programs (tasks). Strictly speaking, if the CPU is one unit, it executes two or more programs (tasks) while switching between them in an extremely short time almost as if they were being simultaneously executed. The ERCX controller uses this multi-task function to perform multiple tasks while switching the programs within a very short time (5ms maximum). Because of this, if 4 tasks are executed with this function, there is a maximum time of 20ms during which no processing is performed on one task. So the user must take this time into account when designing a system having multi-task functions.
8-3-2 Limitless movement function

The limitless movement function allows multiple turns in the same direction along the robot axis. The ERCX controller incorporates the soft limit function that prohibits any robot motion which exceeds the soft limits specified by the parameters. This soft limit function is very useful for linear movement type robots such as FLIP-X series. However, this function is sometimes undesirable for rotary type robots such as FROP because it limits multi-turn movements in the same direction. In such cases, the limitless movement function will prove useful.

To enable the limitless movement function, set the position data unit parameter (PRM21) to 2 or 3. The limitless movement function does not work when PRM21 is set to 0 or 1. Setting PRM21 to 2 is suitable for applications using servo conveyors, and setting to 3 is suitable for applications using FROP or index tables.

When the position data unit parameter is set to 2:

When this parameter is set to 2, the current position is expressed in millimeters from 0 to the "plus soft limit - 0.01mm" as a basic cycle. Therefore, even if the robot moves to the plus soft limit point, that position sets to 0mm so that the robot can move continuously in the same direction. In limitless movement, the movement direction can also be selected with a movement command *1) such as MOV A which specifies a target position.

To select a movement direction opposite the return-to-origin direction, add 5000mm to the target point for point setting. To select the same movement direction as when performing return-to-origin, add 5000mm to the target point and give a minus sign to this value. When the movement direction is not specified, the robot moves in the direction of shorter distance. For example, when just 30mm is specified for the point setting, the movement direction differs depending on the current position. However, when 5030mm is specified, the robot always moves in the direction opposite the return-to-origin direction. In contrast, when -5030mm is specified, the robot always moves in the same direction as when performing return-to-origin.

In the case of a movement command such as MOVI which specifies the amount of movement, the movement direction is determined by the plus/minus sign of the point data, just as with normal movement.

CAUTION

- When the parameter is 2, set the plus soft limit to always be an integer multiple of the lead equivalent value.*2)
  If it is not a value multiplied by an integer, positioning at the desired point may sometimes be impossible. The + soft limit setting range is 1 to 4999.
- The maximum distance of one movement is a distance equal to one cycle (+ soft limit). To move a distance longer than one cycle, divide the movement distance into two or more portions.
- In limitless movement, @XINC (@XDEC) allows moving a distance equal to one cycle. The movement speed setting and stop method are just the same as for normal movement.
- If the target point is the same as the current position on the program when executing a movement command such as MOV A which specifies a position, the robot motion differs depending on whether the movement direction is selected by point setting, as follows:
  When the movement direction is selected: Moves a distance equal to one cycle in the selected direction and stops.
  When no movement direction is selected: Does not move.
- Use caution when operating the robot since the soft limits are disabled during limitless movement.

*1) These movement commands include MOVA, MOVF and MOVD statements. The MOVD statement is provided only for communication commands and can directly specify the target point.

*2) The lead equivalent value can be checked with the lead length parameter (PRM12). The lead length parameter indicates a distance the robot axis (or workpiece on the axis) moves while the motor makes one turn, in units of 1/100mm.
When the position data unit parameter is set to 3:

When this parameter is set to 3, the current position is expressed in degrees (°) from 0 to 359.99 as a basic cycle. Therefore, even if the robot moves to the 360° point, that position sets to 0° (≈360°) so that the robot can rotate continuously in the same direction.

In limitless movement, the rotation direction can also be selected with a movement command *1) such as MOVA which specifies a target position.

To select the rotation direction opposite the return-to-origin direction, add 5000° to the target point for point setting. To select the same rotation direction as when performing return-to-origin, add 5000° to the target point and give a minus sign to this value. When the rotation direction is not specified, the robot moves in the direction of shorter distance. For example, when just 30° is specified for the point setting, the rotation direction differs depending on the current position. However, when 5030° is specified, the robot always rotates in the direction opposite the return-to-origin direction. In contrast, when -5030° is specified, the robot always rotates in the same direction as when performing return-to-origin.

In the case of a movement command such as MOVI which specifies the amount of movement, the movement direction is determined by the plus/minus sign of point data, just as with normal movement.

**CAUTION**

- The maximum distance of one movement is a distance equal to one cycle (360°). To move a distance longer than one cycle, divide the movement distance into two or more portions.
- In limitless movement, @XINC (@XDEC) allows moving a distance equal to one cycle. The movement speed setting and stop method are just the same as for normal movement.
- If the target point is the same as the current position on the program when executing a movement command such as MOVA which specifies a position, the robot movement differs depending on whether the rotation direction is selected by point setting, as follows:
  - When the rotation direction is selected: Rotates through 360° in the selected direction and stops.
  - When no rotation direction is selected: Does not move.
- Use caution when operating the robot since the soft limits are disabled during limitless movement.

*1) These movement commands include MOVA, MOVF and MOVD statements. The MOVD statement is provided only for communication commands and can directly specify the target point.
8-4 Robot Language Description

8-4-1 MOVA
Function: Moves to a point specified by a point number (Moves to an absolute position relative to the origin point).
Format: MOVA <point number>, <maximum speed>
Example: MOVA 51, 80
Moves to P51 at speed 80.
Explanation: This command moves the robot to a position on the absolute coordinates whose origin position is defined as 0.
The robot starts moving when all axes enter the positioning-completed pulse range, and stops when all axes reach the OUT valid position.
(1) Point number
The point number is a number designated to each point from 0 to 999, a total of 1000 points, and used to create point date in PNT (point) mode. When a character "P" is entered here for special use, a point variable defined by the "P" statement is set as the point number. (See "8-4-12 P").
(2) Maximum speed
The maximum speed can be set to any level between 1 and 100. If the execution speed in OPRT mode is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).

8-4-2 MOVI
Function: Moves a distance specified by a point number from the current position.
Format: MOVI <point number>, <maximum speed>
Example: MOVI 10, 80
Moves an amount equal to point data P10 from the current position at speed 80.
Explanation: This command moves the robot on the relative coordinates with the current position viewed as 0.
The robot starts moving when all axes enter the positioning-completed pulse range, and stops when all axes reach the OUT valid position.
(1) Point number
The point number is a number designated to each point from 0 to 999, a total of 1000 points, and used to create point date in PNT (point) mode. When a character "P" is entered here for special use, a point variable defined by the "P" statement is set as the point number. (See "8-4-12 P").
(2) Maximum speed
The maximum speed can be set to any level between 1 and 100. If the execution speed in OPRT mode is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).
8-4-3 MOVF

**Function:** Moves until a specified DI number input is received.

**Format:** MOVF <point number> <DI number> <DI status>

**Example:** MOVF 1, 2, 1

The robot moves toward P1 and stops when DI2 turns on. Program execution then proceeds to the next step.

**Explanation:** This is used when searching for a target position using sensors or other devices. The robot starts moving when all axes enter the positioning-completed pulse range, and stops when the DI conditions are met. Even if the DI conditions are not met, this command terminates when the robot reaches the specified point and proceeds to the next step.

(1) Point number

The point number is a number designated to each point from 0 to 999, a total of 1000 points, and used to create point date in PNT (point) mode. When a character "P" is entered here for special use, a point variable defined by the "P" statement is set as the point number. (See "8-4-12 P").

(2) DI number

Specify one of the 16 (0 to 15) general-purpose inputs.

(3) DI status

"1" means "on" and "0" means "off".

**Other:**

- The robot speed during execution of the MOVF movement can be specified by PRM9. (Refer to "PRM9: MOVF speed") Note that this will not be affected by the OPRT mode execution speed.

8-4-4 JMP

**Function:** Jumps to a specified step in a specified program.

**Format:** JMP <label number>, <program number>

**Example:** JMP 10, 8

Jumps to label 10 in program 8.

**Explanation:** This command is used to control the flow of program execution.

(1) Label number

The label number is a number defined by the "L" statement, and indicates the destination to jump. Any number from 0 to 255 can be specified. (See "8-4-7 L").

(2) Program number

The program number is a number used to identify the 100 individual programs from 0 to 99.

**Other:**

- Even when the program number is changed by the JMP statement, resetting it will return to the original program number when program execution begins.
8-4-5 JMPF

Function: If the conditional jump input matches the setting value, program execution jumps to a specified label in a specified program.

Format: JMPF  <label number>, <program number>, <input condition value>

Example: JMPF     12, 3, 5
If the conditional jump input is 5, program execution jumps to label 12 in program 3. If the jump input is not 5, program execution advances to the next step.

Explanation: This command is used to control the flow of program execution according to the conditional jump input.
(1) Label number
The label number is a number defined by the "L" statement, and indicates the destination to jump. Any number from 0 to 255 can be specified. (See "8-4-7 L").
(2) Program number
The program number is a number used to identify the 100 individual programs from 0 to 99.
(3) Input condition value
This is the condition used to make a jump. A general-purpose input is viewed as binary code input, and if it meets this input condition value, a jump is executed. The number of points that can be branched under the input condition depends on the number of conditional input points which is set with PRM8. (See "PRM8: No. of conditional input points").

⚠️ CAUTION
Select a number of conditional input points that is large enough to accommodate the actual number of input conditions to be used. If an error is made in setting the number of conditional input points, there will be a discrepancy between the input condition value required by the program and that recognized by the controller. This could keep the program from operating properly.

General-purpose input status and input condition value when the number of conditional input points is 4 (input range 0 to 15)

<table>
<thead>
<tr>
<th>General use DI Input</th>
<th>Input Condition Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>D13 (2^3)</td>
<td>D12 (2^2)</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>OFF</td>
<td>ON</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>
8-4-6 JMPB

Function: Jumps to a specified label when a specified general-purpose input or memory input is ON or OFF.

Format: JMPB <label number>, <DI or MI number>, <input status>

Example: JMPB 12, 8, 1
Jumps to label 12 when DI8 input is ON.
If DI8 is OFF, the program execution proceeds to the next step.

Explanation: This command controls the program flow according to the general-purpose input or memory input status.

(1) Label number
The label number is a number defined by the "L" statement, and indicates the destination to jump. Any number from 0 to 255 can be specified. (See "8-4-7 L.")

(2) DI or MI number
Specify one of the general-purpose input numbers from 0 to 15 (16 points) or memory input numbers from 100 to 147 (48 points).

(3) Input status
"1" means "on" and "0" means "off".

8-4-7 L

Function: Defines the jump destination for JMP, JMPF or JMPB statements, etc.

Format: L <label number>

Example: L 100
Defines label 100.

Explanation: This command is used to define the destination to which program execution jumps with a jump command. The label number may be any number between 0 and 255. The same label numbers may be used if they are in different programs.
8-4-8 CALL

Function: Calls and executes another program.

Format: CALL <program number>, <number of times>

Example: CALL 5, 2

Calls program 5 and executes it twice. Program execution then proceeds to the next step.

Explanation: When repeating the same operation a number of times, the CALL statement is used as needed to call and execute the subroutine defined as a separate program.

(1) Program number
   The program number is a number used to identify the 100 individual programs from 0 to 99.

(2) Number of times
   This is the number of times that the program is to be repeated. This can be specified from 1 to 255.

Other:
• The nesting level is 6.
• When the end of the program initiated by the CALL statement is detected, program execution advances to the step following the CALL statement in the main program.
• An error occurs and program execution stops if the program being executed is called by the CALL statement.
• Even when the program number is changed by the CALL statement, resetting it will return to the original program number when program execution begins.
• An error "stack overflow" might occur if a jump is made to another program by the JMP or JMPF statement in a program called as a subroutine by the CALL statement.

8-4-9 DO

Function: Controls ON/OFF of general-purpose output or memory output.

Format: DO <DO or MO number>, <output status>

Example: DO 3, 1

Turns on DO3.

Explanation: This command turns the general-purpose output or memory output on and off.

(1) DO or MO number
   Specify one of the general-purpose output numbers from 0 to 12 (13 points) or memory output numbers from 100 to 131 (32 points).

(2) Output status
   "1" means "on" and "0" means "off".
8-4-10 WAIT

Function: Waits until a specified general-purpose input or memory input changes to a specified state.
Format: WAIT <DI or MI number>, <input status>
Example: WAIT 5, 1
Waits until DI5 turns on.
Explanation: This command adjusts the timing according to the general-purpose input or memory input state.
(1) DI or MI number
Specify one of the general-purpose input numbers from 0 to 15 (16 points) or memory input numbers from 100 to 147 (48 points).
(2) Input status
"1" means "on" and "0" means "off".

8-4-11 TIMR

Function: Waits for a specified amount of time before advancing to the next step.
Format: TIMR <time>
Example: TIMR 100
Moves to the next step after waiting one second.
Explanation: This command is used when adjusting the time within the program. Time may be specified in lengths from 1 to 65535, in units of 10ms. In other words, time may be specified from 0.01 seconds up to 655.35 seconds.
8-4 Robot Language Description

8-4-12 P

Function: Sets a point variable P.
Format: P <point number>
Example: P 200
Sets a point variable P to 200.
Explanation: The point variable can contain a point number as a variable, which can be from 0 to 999. By using a movement command such as MOVA with a P+ or P- statement, the number of steps required to create a repeating program can be reduced.
Other: • The contents of point variable P are retained even when the controller power is turned off, but when the program is reset or when the program reset is applied for example by switching the execution program, the point variable P will be initialized to 0.
• Point variables P in a task are independent of those in other tasks. For example, the definition and edited contents of a point variable used in task 1 do not affect the point variable used in task 0.

8-4-13 P+

Function: Adds 1 to a point variable P.
Format: P+
Example: P+
Adds 1 to a point variable P. (P ← P+1)
Explanation: Adds 1 to a point variable P.

8-4-14 P-

Function: Subtracts 1 from a point variable P.
Format: P-
Example: P-
Subtracts 1 from a point variable P. (P ← P-1)
Explanation: Subtracts 1 from a point variable P.
8-4-15 SRVO

Function: Turns the servo on and off.
Format: SRVO <servo status>
Example: SRVO 1
This turns the servo on.
SRVO 0
This turns the servo off.

Explanation: This command is used to prevent an overload on the motor that may occur if the robot is locked mechanically after positioning is completed. This command is executed after the specified axis enters the positioning-completed pulse range.
(1) Servo status
"1" means "on" and "0" means "off."

8-4-16 STOP

Function: Temporarily interrupts program execution.
Format: STOP
Example: STOP
Temporarily interrupts program execution.

Explanation: This command temporarily interrupts execution of a program. If two or more tasks are being executed, then all those tasks are interrupted. This command can be used at any point in a program. The next execution will restart from the subsequent step.

Others:
• Normally, the program terminates when the last step is detected. At the same time, the program is reset and the execution step number will return to 1 (top line of the program).
• To interrupt only a subtask temporarily without stopping the main task, use the TOFF statement. (Refer to "8-4-19 TOFF").
8-4-17 ORGN

Function: Performs return-to-origin when the search method is selected as the origin detection method, or checks whether return-to-origin has been performed when the mark method is selected.

Format: ORGN

Example:

Performs return-to-origin when the search method is selected as the origin detection method, or checks whether return-to-origin has been performed when the mark method is selected.

Explanation: Return-to-origin is performed based on the return-to-origin parameter data when the search method is selected as the origin detection method. When the mark method is selected, this command checks whether return-to-origin has been performed and proceeds to the next step when it has been performed, but halts the operation as an error if not performed.

Others:

- Once return-to-origin is performed after the robot cable and absolute battery are connected, there is no need to repeat it even when the controller is turned off. (As an exception, return-to-origin becomes incomplete if the absolute backup function is disabled or a parameter relating to the origin is changed. Return-to-origin must be reperformed in this case.)
- When performing return-to-origin by the stroke-end detection method, do not interrupt the return-to-origin operation while detecting the origin (while contacting the mechanical limit). Otherwise, the operation will stop due to a controller overload alarm and the power will have to be turned on again.
- If return-to-origin must be repeated by the stroke-end detection method, wait at least 5 seconds before repeating it.
8-4-18 TON

Function: Executes a specified task.
Format: TON <task number>, <program number>, <start type>
Example: TON 1, 2, 0
Newly executes program 2 as task 1.
Explanation: This command starts multiple tasks and can be used to control the I/O signals in parallel with the axis movement and perform different processing for each axis.
(1) Task number
The task number is a number used to identify the four individual tasks from 0 to 3. Since task 0 is the main task, tasks numbers from 1 to 3 can be specified.
(2) Program number
The program number is a number used to identify the 100 individual programs from 0 to 99.
(3) Start type
This specifies whether to start a new task or suspended task. Set to 0 when executing a new task, and set to 1 when restarting a suspended task.
Others: • A task number which is being executed cannot be specified. (A task number which has been suspended can be specified.)
• The task terminates when the last step is detected. When a subtask terminates, it does not affect operation of other tasks. But, if task 0 (main task) terminates, all other tasks in operation also terminate.

8-4-19 TOFF

Function: Suspends a specified task.
Format: TOFF <task number>
Example: TOFF 1
Suspends the program being executed as task 1.
Explanation: This command is used to suspend the execution of a particular task.
(1) Task number
The task number is a number used to identify the four individual tasks from 0 to 3. Since task 0 is the main task, tasks numbers from 1 to 3 can be specified.
Others: • This command cannot suspend its own task.
8-4-20 JMPP

Function: Jumps to a specified label when the axis position relation meets the specified conditions.

Format: JMPP <label number>, <axis position condition>

Example: JMPP 3, 1
Jumps to label 3 if the X-axis position is smaller than the point specified with the point variable P.

Explanation: This command controls the program flow according to the specified position of the axis, by comparing it with the point specified with the point variable P.

(1) Label number
The label number is a number defined by the "L" statement, and indicates the destination to jump. Any number from 0 to 255 can be specified. (See "8-4-7 L ".)

(2) Axis position condition
When set to 1, this establishes the condition that the robot must be closer to the origin than the specified position. When set to 2, this establishes the condition that the robot must be farther away from the origin than the specified position.

Others:
- When the axis is at the specified coordinate position, this views that the condition is met.

<table>
<thead>
<tr>
<th>Axis position condition</th>
<th>Robot position that meets the condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Robot is in area A.</td>
</tr>
<tr>
<td>2</td>
<td>Robot is in area B.</td>
</tr>
</tbody>
</table>

Point specified with point variable P
**8-4-21 MAT**

**Function:** Defines the number of rows and columns of the matrix.

**Format:** MAT <number of rows>, <number of columns>, <pallet number>

**Example:** MAT 3, 6, 0

Defines a matrix of $3 \times 6$ on pallet number 0.

**Explanation:** This command defines a matrix for palletizing movement. A palletizing program can be easily created by using this command with MSEL or MOVM, etc.

1. **Number of rows**
   - Set any value from 1 to 255.
2. **Number of columns**
   - Set any value from 1 to 255.
3. **Pallet number**
   - This number is used for matrix identification and can be set from 0 to 31. A total of 32 matrix data can be handled.

**Others:**
- A common method for matrix coordinate definition specifies only the positions of the 4 corners of the matrix by 4-point teaching. The remaining points are then found by calculation. When teaching the positions of the 4 corners in PNT (point) mode to create point data, the point numbers are generally specified as follows: If pallet number is "n" for instance, enter the position of the reference point (row 1, column 1) in p(251-4n), the position at the end of row 1 in p(252-4n), the position at the end of column 1 in p(253-4n), and the position of the remaining corner in p(254-4n). To define a one-dimension matrix such as "row 1, column m", enter the position of the reference point (row 1, column 1) in p251, and the position of last point (row 1, column m) in p252. You do not have to enter any data in p253 and p254 (when pallet number is 0).
- The matrix definition contents are shared with each task.
- Because only a single-axis robot is controlled with the ERCX series, the actual movement is linear even if a 2-dimensional matrix is defined.

---

**Matching point numbers for inputting pallet numbers and coordinate values A to D**

<table>
<thead>
<tr>
<th>Pallet No.</th>
<th>0</th>
<th>1</th>
<th>n</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>p251</td>
<td>p247</td>
<td>p(251-4n)</td>
<td>p127</td>
</tr>
<tr>
<td>B</td>
<td>p252</td>
<td>p248</td>
<td>p(252-4n)</td>
<td>p128</td>
</tr>
<tr>
<td>C</td>
<td>p253</td>
<td>p249</td>
<td>p(253-4n)</td>
<td>p129</td>
</tr>
<tr>
<td>D</td>
<td>p254</td>
<td>p250</td>
<td>p(254-4n)</td>
<td>p130</td>
</tr>
</tbody>
</table>
8-4-22 MSEL

Function: Specifies a matrix where the robot moves with a MOVM statement.
Format: MSEL <pallet number>
Example: MSEL 0
Example: Points where the robot moves with a MOVM statement are calculated based on matrix data of pallet number 0.
Explanation: This command selects a matrix and is always used with a MOVM statement as a pair.
   (1) Pallet number
   This number is used for matrix identification and can be set from 0 to 31.
Others: • The pallet number assigned with the MSEL statement is independent of each task. For example, when different pallet numbers are assigned to task 0 and task 1, then task 0 and task 1 execute the MOVM statement based on different pallet data.
8-4-23 MOVM

Function: Moves to a point on the specified matrix.
Format: MOVM <pallet work position>, <maximum speed>
Example: MOVM 23, 100

Moves to the point at row 3, column 7 at speed 100 when a matrix of $5 \times 8$ is defined by the MAT statement.

Explanation: This command moves the robot to each point on a matrix specified by the MSEL statement.

This command allows the robot to start moving when all axes are within the "positioning-completed pulse" range. This command ends when all axes enter the OUT valid position.

(1) Pallet work position

The pallet work position is a number used to identify each point on a matrix, and can be from 1 to 65025 ($=255 \times 255$). For example, on a "row M, column N" matrix, the pallet work position at "row A, column B" is found by $(A-1) \times N+B$. When a character "C" or "D" is entered here for special use, a counter variable is set in each pallet work position.

(2) Maximum speed

The maximum speed can be set to any level between 1 and 100. If the execution speed in OPRT mode is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).

Others:

- The MOVM statement performs calculation on the assumption that the robot operates on the Cartesian coordinate system.
- Because only a single-axis robot is controlled with the ERCX series, the actual movement is linear even if a 2-dimensional matrix is defined.

Example of pallet work position in $5 \times 8$ matrix

<table>
<thead>
<tr>
<th>Work Position</th>
<th>Position No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: Reference point</td>
<td>1</td>
</tr>
<tr>
<td>B: End of row 1</td>
<td>8</td>
</tr>
<tr>
<td>C: End of column 1</td>
<td>33</td>
</tr>
<tr>
<td>D: Last point</td>
<td>40</td>
</tr>
</tbody>
</table>

Example of pallet work position in $5 \times 8$ matrix
8-4-24 JMPC

Function: Jumps to a specified label when the counter array variable C matches a specified value.

Format: JMPC <label number>, <counter value>

Example: JMPC 5, 100

Jumps to label 5 when the counter array variable C is 100. Program execution proceeds to the next step except when the counter array variable C is 100.

Explanation: This command controls the program flow according to the counter array variable C. The counter array variable C to be compared is the element number specified with the CSEL statement.

(1) Label number
   The label number is a number defined by the "L" statement, and indicates the destination to jump. Any number from 0 to 255 can be specified. (See "8-4-7 L")

(2) Counter value
   Set any value from 0 to 65535.

8-4-25 JMPD

Function: Jumps to a specified label when the counter variable D matches a specified value.

Format: JMPD <label number>, <counter value>

Example: JMPD 5, 100

Jumps to label 5 when the counter variable D is 100. Program execution proceeds to the next step except when the counter variable D is 100.

Explanation: This command controls the program flow according to the counter variable D.

(1) Label number
   The label number is a number defined by the "L" statement, and indicates the destination to jump. Any number from 0 to 255 can be specified. (See "8-4-7 L")

(2) Counter value
   Set any value from 0 to 65535.
8-4-26 CSEL

Function: Specifies an array element of the counter array variable C to be used.

Format: CSEL <array element number>

Example: CSEL 1

The counter array variable of element number 1 is used in the subsequent steps.

Explanation: This command designates an array element number of the counter array variable C.

The array element data designated with the CSEL statement is used in the C statement, C+ statement, C- statement, JMPC statement and MOVM statement.

(1) Array element number

This is a number used to designate the array element number of a counter array variable and can be any value from 0 to 31.

When a character “D” is entered here, the counter variable D is used to designate the element of the counter array variable.

Others:

• The array element designation is held even when the controller power is turned off, but when the program is reset or when the program reset is applied by switching the execution program, the element designation number will be initialized to 0.

• The element number designated with the CSEL statement is independent of each task. For example, when different array elements are designated for task 0 and task 1, the definition or change in the counter array variable C of task 1 does not affect task 0.

8-4-27 C

Function: Sets the counter array variable C.

Format: C <counter value>

Example: C 200

Sets the counter array variable C to 200.

Explanation: This command sets a counter value for the counter array variable specified with the CSEL statement. The counter array variable is an array variable containing 32 elements, and can be set to any value from 0 to 65535. This command can be used with a C+ or C– statement and a JMPC statement for a repeating program and also with a MOVM statement for a palletizing program.

Others:

• Counter array variable C is not initialized even if the program is reset or the controller power is turned off. To initialize, rewrite the program.

• The counter array variable C is a variable shared with all tasks. For example, task 0 and task 1 use a counter array variable with the same element number, the edited contents of task 1 affect task 0.
8-4-28 C+

Function: Adds a specified value to the counter array variable C.
Format: C+ [<addition value>]
Example: C+ 100

Adds 100 to the counter array variable C. (C ← C+100)
C+

Adds 1 to the counter array variable C. (C ← C+1)

Explanation: This command adds a specified value to the counter array variable C specified with the CSEL statement. The addition value can be set to any value from 1 to 65535. If the addition value is omitted, then 1 is added to the counter array variable C.

8-4-29 C-

Function: Subtracts a specified value from the counter array variable C.
Format: C- [<subtraction value>]
Example: C- 100

Subtracts 100 from the counter array variable C. (C ← C-100)
C-

Subtracts 1 from the counter array variable C. (C ← C-1)

Explanation: This command subtracts a specified value from the counter array variable C specified with the CSEL statement. The subtraction value can be set to any value from 1 to 65535. If the subtraction value is omitted, then 1 is subtracted from the counter array variable C.

8-4-30 D

Function: Sets the counter variable D.
Format: D <counter value>
Example: D 200

Sets the counter variable D to 200.

Explanation: The counter variable D can be set to any value by the user from 0 to 65535. This command can be used with a D+ or D- statement and a JMPD statement for a repeating program, and also with a MOVM statement for a palletizing program.

Others: • The counter variable D is not initialized even if the program is reset or the controller power is turned off. To initialize, rewrite the program.
• The counter variable D is a variable shared with all tasks.

For example, task 0 and task 1 use the counter variable D, the edited contents of task 1 affect task 0.
8-4-31 D+

**Function:** Adds a specified value to the counter variable D.

**Format:** D+ \[<addition value>\]

**Example:**
- D+ 100
- Adds 100 to the counter variable D. (D ← D+100)
- D+
- Adds 1 to the counter variable D. (D ← D+1)

**Explanation:** This command adds a specified value to the counter variable D. The addition value can be set to any value from 1 to 65535. If the addition value is omitted, then 1 is added to the counter variable D.

8-4-32 D-

**Function:** Subtracts a specified value from the counter variable D.

**Format:** D- \[<subtraction value>\]

**Example:**
- D- 100
- Subtracts 100 from the counter variable D. (D ← D-100)
- D-
- Subtracts 1 from the counter variable D. (D ← D-1)

**Explanation:** This command subtracts a specified value from the counter variable D. The subtraction value can be set to any value from 1 to 65535. If the subtraction value is omitted, then 1 is subtracted from the counter variable D.
8-4-33 SHFT

**Function:** Shifts the position data.

**Format:** SHFT <point number>

**Example:**

```
004 :    :
005 : SHFT 1
006 : MOVA 0,100
007 :    :
```

Shifts the coordinates on which the subsequent movement commands are executed, by a data amount defined by point 10.

**Explanation:** This command shifts position data in the subsequent movement commands to be executed, by coordinates equal to the specified point data. The shift data is valid until the SHFT statement is executed again or until the program reset is executed.

1. **Point number**
   
   The point number is a number designated to each point from 0 to 999, a total of 1000 points, and used to create point data in PNT (point) mode. When a character "P" is entered here for special use, a point variable defined by the "P" statement is set as the point number. (Refer to "8-4-12 P".)

2. **Others:**
   
   - When the program is reset or when the program reset is applied by switching the execution program, the shift data will be initialized to (0.00).
   - The SHFT statement affects MOVA, MOVF and MOVM, but does not affect MOVI.
   - The coordinate shift amount specified with the SHFT statement is independent of each task. For example, when task 0 and task 1 are being executed, the coordinate shift of task 1 has no effect on the movement command for task 0.

For example, with a program shown on the left, the target position of MOVA statement in the 6th step will be the position of P0 + P1.
### 8-5 Sample Programs

#### 8-5-1 Moving between two points

![Diagram of two points, P1 and P2, connected by a line.]

<table>
<thead>
<tr>
<th>Program</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>[NO0]</td>
<td></td>
</tr>
<tr>
<td>001: L</td>
<td>0</td>
</tr>
<tr>
<td>002: MOVA</td>
<td>1, 100</td>
</tr>
<tr>
<td>003: MOVA</td>
<td>2, 100</td>
</tr>
<tr>
<td>004: TIMR</td>
<td>100</td>
</tr>
<tr>
<td>005: JMP</td>
<td>0, 0</td>
</tr>
</tbody>
</table>

#### 8-5-2 Moving at an equal pitch

![Diagram showing a series of points moving 50mm four times.]

<table>
<thead>
<tr>
<th>Program</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>[NO0]</td>
<td></td>
</tr>
<tr>
<td>001: L</td>
<td>0</td>
</tr>
<tr>
<td>002: MOVA</td>
<td>0, 100</td>
</tr>
<tr>
<td>003: MOVI</td>
<td>1, 100</td>
</tr>
<tr>
<td>004: MOVI</td>
<td>1, 100</td>
</tr>
<tr>
<td>005: MOVI</td>
<td>1, 100</td>
</tr>
<tr>
<td>006: MOVI</td>
<td>1, 100</td>
</tr>
<tr>
<td>007: MOVI</td>
<td>1, 100</td>
</tr>
<tr>
<td>008: JMP</td>
<td>0, 0</td>
</tr>
</tbody>
</table>
8-5-3 Positioning 2 points and sending job commands to a PLC at each position

Point
- P1: Position at which job 1 is complete
- P2: Position at which job 2 is complete

General-purpose input
- DI1: Job 1 completion 1: Complete 0: Not complete
- DI2: Job 2 completion 1: Complete 0: Not complete

General-purpose output
- DO1: Job 1 command 1: Output 0: Canceled
- DO2: Job 2 command 1: Output 0: Canceled

Program

<table>
<thead>
<tr>
<th>Program</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>[NO0]</td>
<td></td>
</tr>
<tr>
<td>001: DO 1, 0</td>
<td>Cancels job 1 command</td>
</tr>
<tr>
<td>002: DO 2, 0</td>
<td>Cancels job 2 command</td>
</tr>
<tr>
<td>003: L 1</td>
<td>Label definition</td>
</tr>
<tr>
<td>004: MOVA 1, 100</td>
<td>Moves to P1</td>
</tr>
<tr>
<td>005: DO 1, 1</td>
<td>Outputs job 1 command</td>
</tr>
<tr>
<td>006: WAIT 1, 1</td>
<td>Waits until job 1 is complete</td>
</tr>
<tr>
<td>007: DO 1, 0</td>
<td>Cancels job 1 command</td>
</tr>
<tr>
<td>008: MOVA 2, 100</td>
<td>Moves to P2</td>
</tr>
<tr>
<td>009: DO 2, 1</td>
<td>Outputs job 2 command</td>
</tr>
<tr>
<td>010: WAIT 2, 1</td>
<td>Waits until job 2 is complete</td>
</tr>
<tr>
<td>011: DO 2, 0</td>
<td>Cancels job 2 command</td>
</tr>
<tr>
<td>012: JMP 1, 0</td>
<td>Returns to L1</td>
</tr>
</tbody>
</table>
8-5 Sample Programs

8-5-4 Robot stands by at P0, and moves to P1 and then to P2 to pick and place a workpiece

**Operation**
1. Moves to the workpiece feed position from the standby position, and picks up a workpiece.
2. Moves to the workpiece mount position and places the workpiece.
3. Returns to the standby position.

**Actuator**

<table>
<thead>
<tr>
<th>Direction</th>
<th>Actuator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal</td>
<td>AC servo motor</td>
</tr>
<tr>
<td>Vertical</td>
<td>Air cylinder</td>
</tr>
<tr>
<td>Hold</td>
<td>Air chuck</td>
</tr>
</tbody>
</table>

**General-purpose input**

<table>
<thead>
<tr>
<th>Switch Code</th>
<th>Description</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>DI0</td>
<td>Upper end limit switch</td>
<td>ON: 0, OFF: 1</td>
</tr>
<tr>
<td>DI1</td>
<td>Lower end limit switch</td>
<td>ON: 0, OFF: 1</td>
</tr>
<tr>
<td>DI2</td>
<td>Workpiece detection sensor</td>
<td>Detected: 1, No: 0</td>
</tr>
</tbody>
</table>

**General-purpose output**

<table>
<thead>
<tr>
<th>Output Code</th>
<th>Description</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO0</td>
<td>Air cylinder</td>
<td>Down: 1, Up: 0</td>
</tr>
<tr>
<td>DO1</td>
<td>Air chuck</td>
<td>Close: 1, Open: 0</td>
</tr>
</tbody>
</table>

**Point**

<table>
<thead>
<tr>
<th>Point Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td>Robot standby position</td>
</tr>
<tr>
<td>P1</td>
<td>Workpiece feed position</td>
</tr>
<tr>
<td>P2</td>
<td>Workpiece mount position</td>
</tr>
</tbody>
</table>

**Program [NO1]**

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Parameter</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>L 1</td>
<td></td>
<td>Label definition</td>
</tr>
<tr>
<td>002</td>
<td>MOVA 0, 100</td>
<td></td>
<td>Moves to the standby position</td>
</tr>
<tr>
<td>003</td>
<td>WAIT 2, 1</td>
<td></td>
<td>Waits for workpiece feed</td>
</tr>
<tr>
<td>004</td>
<td>MOVA 1, 100</td>
<td></td>
<td>Moves to the workpiece feed position</td>
</tr>
<tr>
<td>005</td>
<td>DO 0, 1</td>
<td></td>
<td>Air cylinder moves down</td>
</tr>
<tr>
<td>006</td>
<td>WAIT 1, 1</td>
<td></td>
<td>Waits until the air cylinder moves down</td>
</tr>
<tr>
<td>007</td>
<td>DO 1, 1</td>
<td></td>
<td>Chuck closes</td>
</tr>
<tr>
<td>008</td>
<td>TIMR 100</td>
<td></td>
<td>Delays for one second</td>
</tr>
<tr>
<td>009</td>
<td>DO 0, 0</td>
<td></td>
<td>Air cylinder moves up</td>
</tr>
<tr>
<td>010</td>
<td>WAIT 0, 1</td>
<td></td>
<td>Waits until the air cylinder moves up</td>
</tr>
<tr>
<td>011</td>
<td>MOVA 2, 100</td>
<td></td>
<td>Moves to the workpiece mount position</td>
</tr>
<tr>
<td>012</td>
<td>DO 0, 1</td>
<td></td>
<td>Air cylinder moves down</td>
</tr>
<tr>
<td>013</td>
<td>WAIT 1, 1</td>
<td></td>
<td>Waits until the air cylinder moves down</td>
</tr>
<tr>
<td>014</td>
<td>DO 1, 0</td>
<td></td>
<td>Chuck opens</td>
</tr>
<tr>
<td>015</td>
<td>TIMR 100</td>
<td></td>
<td>Delays for one second</td>
</tr>
<tr>
<td>016</td>
<td>DO 0, 0</td>
<td></td>
<td>Air cylinder moves up</td>
</tr>
<tr>
<td>017</td>
<td>WAIT 0, 1</td>
<td></td>
<td>Waits until the air cylinder moves up</td>
</tr>
<tr>
<td>018</td>
<td>JMP 1, 1</td>
<td></td>
<td>Returns to L1</td>
</tr>
</tbody>
</table>
8-5-5 Picking up 3 kinds of workpieces flowing on the front conveyor and placing them on the next conveyors while sorting

Operation
1. Moves to the workpiece feed position and picks up a workpiece.
2. Moves to the workpiece mount position and places the workpiece.

Actuator
- Horizontal direction: AC servo motor
- Vertical direction: Air cylinder
- Hold: Air chuck

General-purpose input
- DI0: Upper end limit switch
- DI1: Lower end limit switch
- DI2: Workpiece A detection sensor
- DI3: Workpiece B detection sensor
- DI4: Workpiece C detection sensor

General-purpose output
- DO0: Air cylinder
- DO1: Air chuck

Point
- P0: Workpiece feed position on the front conveyor
- P1: Workpiece A mount position on next conveyor
- P2: Workpiece B mount position on next conveyor
- P3: Workpiece C mount position on next conveyor
### Program NO1

**<<Main routine>>**

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>L 1</td>
<td>Label definition</td>
</tr>
<tr>
<td>002</td>
<td>JMPB 2, 2, 1</td>
<td>Jumps to L2 when workpiece A is detected</td>
</tr>
<tr>
<td>003</td>
<td>JMPB 3, 3, 1</td>
<td>Jumps to L3 when workpiece B is detected</td>
</tr>
<tr>
<td>004</td>
<td>JMPB 4, 4, 1</td>
<td>Jumps to L4 when workpiece C is detected</td>
</tr>
<tr>
<td>005</td>
<td>JMP 1, 1</td>
<td>Returns to L1</td>
</tr>
<tr>
<td>006</td>
<td>L 2</td>
<td>Label definition</td>
</tr>
<tr>
<td>007</td>
<td>P 1</td>
<td>Sets the point variable to 1</td>
</tr>
<tr>
<td>008</td>
<td>JMP 5, 1</td>
<td>Jumps to L5</td>
</tr>
<tr>
<td>009</td>
<td>L 3</td>
<td>Label definition</td>
</tr>
<tr>
<td>010</td>
<td>P 2</td>
<td>Sets the point variable to 2</td>
</tr>
<tr>
<td>011</td>
<td>JMP 5, 1</td>
<td>Jumps to L5</td>
</tr>
<tr>
<td>012</td>
<td>L 4</td>
<td>Label definition</td>
</tr>
<tr>
<td>013</td>
<td>P 3</td>
<td>Sets the point variable to 3</td>
</tr>
<tr>
<td>014</td>
<td>L 5</td>
<td>Label definition</td>
</tr>
<tr>
<td>015</td>
<td>CALL 2, 1</td>
<td>Executes a [PICK] subroutine</td>
</tr>
<tr>
<td>016</td>
<td>CALL 3, 1</td>
<td>Executes a [PLACE] subroutine</td>
</tr>
<tr>
<td>017</td>
<td>JMP 1, 1</td>
<td>Returns to L1</td>
</tr>
</tbody>
</table>

### Program NO2

**<<Picking up a workpiece>>**

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>MOVA 0, 100</td>
<td>Moves to the workpiece feed position</td>
</tr>
<tr>
<td>002</td>
<td>DO 0, 1</td>
<td>Air cylinder moves down</td>
</tr>
<tr>
<td>003</td>
<td>WAIT 1, 1</td>
<td>Waits until the air cylinder moves down</td>
</tr>
<tr>
<td>004</td>
<td>DO 1, 1</td>
<td>Chuck closes</td>
</tr>
<tr>
<td>005</td>
<td>TIMR 100</td>
<td>Delays for one second</td>
</tr>
<tr>
<td>006</td>
<td>DO 0, 0</td>
<td>Air cylinder moves up</td>
</tr>
<tr>
<td>007</td>
<td>WAIT 0, 1</td>
<td>Waits until the air cylinder moves up</td>
</tr>
</tbody>
</table>

### Program NO3

**<<Placing a workpiece>>**

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>MOVA P, 100</td>
<td>Moves to the workpiece mount position</td>
</tr>
<tr>
<td>002</td>
<td>DO 0, 1</td>
<td>Air cylinder moves down</td>
</tr>
<tr>
<td>003</td>
<td>WAIT 1, 1</td>
<td>Waits until the air cylinder moves down</td>
</tr>
<tr>
<td>004</td>
<td>DO 1, 0</td>
<td>Chuck opens</td>
</tr>
<tr>
<td>005</td>
<td>TIMR 100</td>
<td>Delays for one second</td>
</tr>
<tr>
<td>006</td>
<td>DO 0, 0</td>
<td>Air cylinder moves up</td>
</tr>
<tr>
<td>007</td>
<td>WAIT 0, 1</td>
<td>Waits until the air cylinder moves up</td>
</tr>
</tbody>
</table>
8-5-6 Switching the program from I/O

The ERCX series controller does not accept dedicated command inputs for program switching. To switch the program through the I/O, use the program selection signal as a conditional jump input as explained below.

The following method is an example for switching among 16 kinds of programs.

Parameter
Since the number of programs to be selected is 16, set the PRM8 (No. of conditional input points) to 4.

⚠️ CAUTION
In actual programming, PRM8 must be set to match the number of programs you use. (See the table at the right.)

<table>
<thead>
<tr>
<th>General-purpose input</th>
<th>General-purpose output</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIO, DII, DII, DI3, DI4, DI5, DI6</td>
<td>DIO, DIO to DI1, DI2, DI3, DI4, DI5, DI6</td>
</tr>
</tbody>
</table>

<<Program description>>
When using the JMPF statement to select a program, select the general-purpose input/output points (DIO0 and DI0 in this case) one at a time and perform the handshake. This is for synchronizing the ERCX controller program with an external device such as a PLC. If this part is omitted, the wrong program might be selected during program selection with the JMPF statement.

In specific operations, an external device should turn on DI8 after confirming DI3 to DI0. The ERCX controller then turns on DIO0 just after detecting that DI8 is on, and informs the external device that the program is being selected. When the external device detects that DIO0 is on, DI8 should turn off. (DI3 to DI0 should be retained.) Then, when DIO0 turns off, this means that the program selection is complete, so it is okay to change DI3 to DI0. The program selection is now complete and actual program operations are executed.

When each selected program has been executed, the operation returns to the top of the program (L0 in the program NO0). The operation returns to the top of the program when there is no program matching the program selection input.

Time chart

Input

| DIO0 | Program selection start |
| DIO1 | Program selection 0 |
| DIO2 | Program selection 1 |
| DIO3 | Program selection 2 |
| DIO4 | Program selection 3 |
| DIO5 | Confirmation of selected program |
| DIO6 | Confirmation of selected program |

Output

| DIO0 | Program selection start |
| DIO1 | Program selection 0 |
| DIO2 | Program selection 1 |
| DIO3 | Program selection 2 |
| DIO4 | Program selection 3 |
| DIO5 | Confirmation of selected program |
| DIO6 | Confirmation of selected program |

Execution of selected program
Program Comment

[NO0]
001: L 0 ; Label definition
002: WAIT 8, 1 ; Waits for confirmation ON of the selected program
003: DO 0, 1 ; Program selection start turns on
004: WAIT 8, 0 ; Waits for confirmation OFF of the selected program
005: JMPF 1, 1, 0 ; Jumps to L1 of NO1 when input is 0
006: JMPF 1, 2, 1 ; Jumps to L1 of NO2 when input is 1
007: JMPF 1, 3, 2 ; Jumps to L1 of NO3 when input is 2
008: JMPF 1, 4, 3 ; Jumps to L1 of NO4 when input is 3
009: JMPF 1, 5, 4 ; Jumps to L1 of NO5 when input is 4
010: JMPF 1, 6, 5 ; Jumps to L1 of NO6 when input is 5
011: JMPF 1, 7, 6 ; Jumps to L1 of NO7 when input is 6
012: JMPF 1, 8, 7 ; Jumps to L1 of NO8 when input is 7
013: JMPF 1, 9, 8 ; Jumps to L1 of NO9 when input is 8
014: JMPF 1, 10, 9 ; Jumps to L1 of NO10 when input is 9
015: JMPF 1, 11, 10 ; Jumps to L1 of NO11 when input is 10
016: JMPF 1, 12, 11 ; Jumps to L1 of NO12 when input is 11
017: JMPF 1, 13, 12 ; Jumps to L1 of NO13 when input is 12
018: JMPF 1, 14, 13 ; Jumps to L1 of NO14 when input is 13
019: JMPF 1, 15, 14 ; Jumps to L1 of NO15 when input is 14
020: JMPF 1, 16, 15 ; Jumps to L1 of NO16 when input is 15
021: JMP 0, 0 ; Returns to L0 of program NO0

[NO1]
001: L 1 ; Label definition
002: DO 0, 0 ; Program selection is complete (selection start OFF)
              ; Actual program operation
              ; Returns to L0 of program NO0

[NO2]
001: L 1 ; Label definition
002: DO 0, 0 ; Program selection is complete (selection start OFF)
              ; Actual program operation
              ; Returns to L0 of program NO0

Programs NO3–NO15 should be created in the same way

[NO16]
001: L 1 ; Label definition
002: DO 0, 0 ; Program selection is complete (selection start OFF)
              ; Actual program operation
              ; Returns to L0 of program NO0
8-5-7  Axis movement and I/O multi-task

The robot moves between two points and performs multi-task I/O operation in asynchronous mode.

```
Program
[NO0]
001: TON 1, 1, 0; Starts program NO1 as task 1
002: L 0; Label definition
003: MOVA 0, 100; Moves to P0 at speed 100
004: TIMR 100; Delays for one second
005: MOVA 1, 100; Moves to P1 at speed 100
006: TIMR 100; Delays for one second
007: JMP 0, 0; Returns to L0

Program
[NO1]
001: L 0; Label definition
002: WAIT 0, 0; Waits until the job is finished
003: DO 0, 1; Issues the job start instruction
004: WAIT 0, 1; Confirms that the job has started
005: DO 0, 0; Turns off the job start signal
006: JMP 0, 1; Returns to L0
```
8-5-8 Turning ON general-purpose outputs during robot movement after a certain time has elapsed

**Program [NO0]**

- **001: L 0** ; Label definition
- **002: MOVA 0, 100** ; Moves to P0 at speed 100
- **003: DO 0, 0** ; Turns DO0 off
- **004: DO 1, 0** ; Turns DO1 off
- **005: DO 2, 0** ; Turns DO2 off
- **006: TON 1, 1, 0** ; Starts program NO1 as task 1
- **007: MOVA 1, 10** ; Moves to P1 at speed 10
- **008: JMP 0, 0** ; Returns to L0

**Program [NO1]**

- **001: TIMR 300** ; Delays for 3 seconds
- **002: DO 0, 1** ; Turns DO0 on
- **003: TIMR 300** ; Delays for 3 seconds
- **004: DO 1, 1** ; Turns DO1 on
- **005: TIMR 300** ; Delays for 3 seconds
- **006: DO 2, 1** ; Turns DO2 on
8-5-9 Turning ON a general-purpose output during robot movement when it has passed a specified position

![Robot movement diagram]

- When P1 is nearer to the plus side than P0:

Program [NO0]

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001:</td>
<td>L 0</td>
<td>; Label definition</td>
</tr>
<tr>
<td>002:</td>
<td>MOVA 0, 100</td>
<td>; Moves to P0 at speed 100</td>
</tr>
<tr>
<td>003:</td>
<td>TON 1, 1, 0</td>
<td>; Starts program NO1 as task 1</td>
</tr>
<tr>
<td>004:</td>
<td>MOVA 1, 10</td>
<td>; Moves to P1 at speed 10</td>
</tr>
<tr>
<td>005:</td>
<td>JMP 0, 0</td>
<td>; Returns to L0</td>
</tr>
</tbody>
</table>

Program [NO1]

<table>
<thead>
<tr>
<th>Line</th>
<th>Instruction</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001:</td>
<td>DO 0, 0</td>
<td>; Turns DO0 off</td>
</tr>
<tr>
<td>002:</td>
<td>P 10</td>
<td>; Sets the point variable to 10</td>
</tr>
<tr>
<td>003:</td>
<td>L 0</td>
<td>; Label definition</td>
</tr>
<tr>
<td>004:</td>
<td>JMPF 0, 1</td>
<td>; Jumps to L0 when the robot does not reach P10</td>
</tr>
<tr>
<td>005:</td>
<td>DO 0, 1</td>
<td>; Turns DO0 on</td>
</tr>
<tr>
<td>006:</td>
<td>P 11</td>
<td>; Sets the point variable to 11</td>
</tr>
<tr>
<td>007:</td>
<td>L 1</td>
<td>; Label definition</td>
</tr>
<tr>
<td>008:</td>
<td>JMPF 1, 1</td>
<td>; Jumps to L1 when the robot does not reach P11</td>
</tr>
<tr>
<td>009:</td>
<td>DO 0, 0</td>
<td>; Turns DO0 off</td>
</tr>
</tbody>
</table>
8-5-10 Limitless movement at same pitch

The robot moves continuously in the same direction at the same pitch (e.g. 150mm) for cycle conveyor applications.

- Make the following settings in advance to enable the limitless movement function.
  - Set the position data unit parameter to 2.
  - Set the plus soft limit to 200. This is a multiple of the lead equivalent value. (The lead equivalent value is assumed to be 20mm.)

- Set P0=0, P1=150 in advance in PNT(point) mode.

<table>
<thead>
<tr>
<th>Program</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV0 0, 100</td>
<td>Moves to P0 at speed 100</td>
</tr>
<tr>
<td>L 0</td>
<td>Label definition</td>
</tr>
<tr>
<td>MOV1 1, 100</td>
<td>Moves 150mm</td>
</tr>
<tr>
<td>JMP 0, 0</td>
<td>Jumps to L0</td>
</tr>
</tbody>
</table>
8-5-11 Limitless rotation

The robot moves continuously in the same direction for index table applications.

- Make the following setting in advance to enable the limitless movement function.
  - Set the position data unit parameter (PRM21) to 3.

- Teach each point of P0 to P3 in advance in PNT(point) mode.

---

### Program

<table>
<thead>
<tr>
<th>No.</th>
<th>Command</th>
<th>Parameter 1</th>
<th>Parameter 2</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>L</td>
<td>0</td>
<td></td>
<td>Label definition</td>
</tr>
<tr>
<td>002</td>
<td>MOVA</td>
<td>0, 100</td>
<td></td>
<td>Moves to P0 at speed 100</td>
</tr>
<tr>
<td>003</td>
<td>MOVA</td>
<td>1, 100</td>
<td></td>
<td>Moves to P1 at speed 100</td>
</tr>
<tr>
<td>004</td>
<td>MOVA</td>
<td>2, 100</td>
<td></td>
<td>Moves to P2 at speed 100</td>
</tr>
<tr>
<td>005</td>
<td>MOVA</td>
<td>3, 100</td>
<td></td>
<td>Moves to P3 at speed 100</td>
</tr>
<tr>
<td>006</td>
<td>JMP</td>
<td>0, 0</td>
<td></td>
<td>Jumps to L0</td>
</tr>
</tbody>
</table>
This chapter describes how to actually operate the robot. If the program has already been completed, you will be able to operate the robot by the time you finish reading this chapter.

There are two types of robot operation: step and automatic. In step operation, the program is executed one step at a time, with a step being carried out each time the RUN key on the TPB is pressed. This is used when you want to check the program as it is being carried out. In automatic operation, the entire program is executed without stopping, from beginning to end.

This chapter also covers how to initiate and recover from an emergency stop.
9-1 Performing Return-to-Origin

There are two methods for detecting the origin position (reference point): search method and mark method. The search method is further divided into the origin sensor method and stroke-end detection method. In the mark method, you can move the robot to a desired position (mark position) and set it as the particular coordinate position to determine a reference point.

The following sections explain how to perform return-to-origin by using the search method and mark method.

Once return-to-origin is performed after the robot cable and absolute battery are connected, there is no need to repeat it even when the controller is turned off. (As an exception, return-to-origin becomes incomplete if the absolute backup function is disabled or a parameter relating to the origin is changed. Return-to-origin must be reperformed in that case.)

9-1-1 Return-to-origin by the search method

When the search method is selected as the origin detection method (PRM13=0 or 1), perform return-to-origin with the procedure below.

1) On the initial screen, press [F2] (OPRT).

2) Next, press [F1] (ORG).


4) This screen is displayed during return-to-origin. Pressing [STOP] during the operation brings the robot to a halt and displays a message. Then, pressing the [ESC] key returns to the screen of step 2.

5) When return-to-origin is completed normally, the machine reference appears on the lower right of the screen. Pressing the [ESC] key returns to the screen of step 2.
9-1 Performing Return-to-Origin

⚠️ CAUTION
When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function").
- Return-to-origin movement speed is limited to 10mm/s or less (10deg/s for rotary robots) in "SERVICE mode state" when the robot movement speed limit is enabled.
- If the hold-to-run function is enabled, robot movement stops upon releasing [F1] (yes) in step 3 in "SERVICE mode state". (You must hold down [F1] (yes) until return-to-origin is complete.)

⚠️ CAUTION
When performing return-to-origin by the stroke-end detection method, do not interrupt the return-to-origin operation while detecting the origin (while contacting the mechanical limit). Otherwise, the operation will stop due to a controller overload alarm and the power will have to be turned on again.

⚠️ CAUTION
If return-to-origin operation using the stroke-end detection method must be repeated, wait at least 5 seconds before repeating it.
**9-1-2 Return-to-origin by the mark method**

When the mark method is selected as the origin detection method (PRM13=2), perform return-to-origin with the procedure below.

1) Press [F2](OPRT) on the initial screen.

2) Next, press [F1](ORG).

3) To use the teaching playback method for setting a mark position, press [F1](TCH). To use the direct teaching method to set a mark position, press [F2](DTCH).

4) When the teaching playback method was selected, use [X−] and [X+] to move the robot to the mark position.
   The movement method is just the same as the teaching playback method for point data. (See “7-2 Teaching Playback”)
   When the robot reaches the mark position, press [ ]. (At this point, check that the machine reference is in a range from 25 to 75%. Otherwise, the origin point cannot be set correctly.)

5) When the direct teaching method is selected, a message appears asking you to press the emergency stop button. Press the emergency button on the TPB.

6) Move the robot by hand to the mark position, and then press [ ].
   (At this point, check that the machine reference is in a range from 25 to 75%. Otherwise, the origin point cannot be set correctly.)
7) When you press \( \text{F3} \) after moving the robot to the mark position by the teaching playback method or direct teaching method, the screen changes to allow entering the coordinate values for the mark position.

While checking the robot stays at the mark position, use the number keys to enter the coordinate values that you want the controller to recognize as the current position, and then press \( \text{F4} \).

8) Press \( \text{F1} \) (yes) to set the origin. To cancel, press \( \text{F2} \) (no).

9) If the machine reference is not in a range from 25 to 75%, the message at the right appears informing you that the origin cannot be set. In this case, press \( \text{ESC} \) and retry the above procedure.

10) When the origin has been set correctly, the message at the right appears. Pressing the \( \text{ESC} \) key returns to step 4 when the teaching playback method was used or to step 6 when the direct teaching method was used.

**NOTE**

When you check the robot position after setting the mark position coordinates, the robot position is not always at the coordinates specified as the mark position. This is because the mark position is synchronized to prevent positional shift and make an exact match when the motor's electrical angle is "0".

When the motor's electrical angle is "0", the machine reference is just 50%. This means that as the machine reference deviates from 50%, the robot position moves away from the coordinates specified as the mark position.

**CAUTION**

When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function ")

- Robot movement speed is limited to 10mm/s or less (10deg/s for rotary robots) in "SERVICE mode state" when the robot movement speed limit is enabled.
9-2 Using Step Operation

The following procedure explains how to perform step operation. In the case of a multi-task program, only the task currently selected is executed in step operation.

1) On the initial screen, press \[F2\] (OPRT).

2) Next, press \[F2\] (STEP).

3) If the program number displayed on the screen is not the one to be run, press \[F3\] (CHG).

4) Using the number keys, enter the number of the program to be executed, and then press \(\rightarrow\).

5) The first step of the selected program is displayed on the screen. To change the execution speed, press \[F1\] (SPD).

6) Enter the execution speed using the number keys, and press \(\rightarrow\).
7) The screen returns to step 5. Pressing RUN at this point executes the first step.

8) This screen is displayed while the program is being executed.

9) Pressing STOP during execution brings the robot to a halt and displays a message on the screen. To return to step 7, press the ESC key. Pressing RUN again executes the interrupted step.

10) When execution is finished, the second step is displayed. Each time RUN is pressed from this point on, the currently displayed step is executed. When the last step has been executed, the message "program end" is displayed. To return to the first step from the program end, press the ESC key.

11) To switch the execution task in a multi-task program, press F4 (next) to change the function menu display and then press F2 (CHGT).

12) Each time you press F2 (CHGT), the task currently in progress is switched. When the task you want to execute is selected, press RUN to execute the step displayed for the selected task.

13) To return to the first step of the program from any other step and initiate execution again, press F2 (RSET).
14) The screen returns to step 5, and the process is repeated from that point.

CAUTION

When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function ")

- Step operation cannot be performed in “SERVICE mode state” when automatic operation and step operation are prohibited.
- Robot movement speed is limited to 3% or less of maximum speed in "SERVICE mode state" when the robot movement speed limit is enabled.
- If the hold-to-run function is enabled, step operation stops upon releasing RUN in "SERVICE mode state".

When one step has been executed, you must release [RUN] and then press [RUN] again to execute the next step.
9-3 Using Automatic Operation

The following procedure explains how to perform automatic operation. All the tasks started in a multi-task program are executed by automatic operation.

1) On the initial screen, press [F2] (OPRT).

2) Next, press [F3] (AUTO).

3) If the program number displayed on the screen is not the one to be run, press [F3] (CHG).

4) Using the number keys, enter the number of the program to be executed and then press .

5) The first step of the selected program is displayed on the screen. To change the execution speed, press [F1] (SPD).

6) Enter the execution speed with the number keys and press .

7) The screen returns to step 5. Pressing [RUN] at this point executes the program all the way to the last step.
8) This is the screen displayed while the program is being executed.

9) Pressing [STOP] during execution brings the robot to a halt and displays the message "stop key". Press the [ESC] key to display the step where execution was interrupted. Pressing [RUN] will cause execution to resume from the step where it was interrupted. When the last step has been executed, the message "program end" is displayed. Pressing the [ESC] key returns the screen to that shown in Step 7.

10) To switch to the display of another task in a multi-task program, press [F4] (next) to change the function menu display and then press [F2] (CHGT).

11) Each time you press [F2] (CHGT), the task display is switched.

12) To return to the first step of the program from any other step and initiate execution again, press [F2] (RSET).

13) The screen returns to step 5, and the process is repeated from that point.

**CAUTION**
When the SERVICE mode function is enabled, the following safety control will function. (See "10-4 SERVICE mode function").
- Automatic operation cannot be performed in "SERVICE mode state" when automatic operation and step operation are prohibited.
- Robot movement speed is limited to 3% or less of maximum speed in "SERVICE mode state" when the robot movement speed limit is enabled.
- If the hold-to-run function is enabled, automatic operation stops upon releasing [RUN] in "SERVICE mode state".
9-4 Switching the Execution Program

The following procedure explains how to switch the program in automatic operation. Use the same procedure in step operation.

The program selected by this procedure will be the lead program to which the execution sequence always returns after program reset.

When the program is switched, reset is automatically performed, so all general-purpose outputs are turned off.

As exceptions, DO4 does not turn OFF when PRM33 (operation at return-to-origin complete parameter) is set to 1 or 3, and DO7 does not turn OFF when PRM46 (servo status output parameter) is set to 1.

1) If the program number displayed on the screen is not the one to be run, press \( \text{F3} \) (CHG).

\[
\begin{align*}
\text{[OPRT-AUTO]} & \quad 100 \ 0: \ 0 \\
001: \text{MOVA} & \quad 254,100 \\
\quad [ & \quad 0.00] \\
\text{1SPD 2RSET3CHG 4next}
\end{align*}
\]

2) Using the number keys, enter the number of the program to be executed and then press \( \rightarrow \).

\[
\begin{align*}
\text{[OPRT-AUTO]} & \quad 100 \ 0: \ 0 \\
\text{PGM No} & = _
\end{align*}
\]

3) The first step of the selected program is displayed on the screen. To change the execution speed, press \( \text{F1} \) (SPD).

\[
\begin{align*}
\text{[OPRT-AUTO]} & \quad 100 \ 0: \ 10 \\
001: \text{MOVA} & \quad 999,50 \\
\quad [ & \quad 0.00] \\
\text{1SPD 2RSET3CHG 4next}
\end{align*}
\]

4) Enter the execution speed using the number keys and press \( \rightarrow \).

\[
\begin{align*}
\text{[OPRT-AUTO]} & \quad 100 \ 0: \ 10 \\
\text{SPEED} & = _
\end{align*}
\]

5) The screen returns to step 3.

\[
\begin{align*}
\text{[OPRT-AUTO]} & \quad 50 \ 0: \ 10 \\
001: \text{MOVA} & \quad 999,50 \\
\quad [ & \quad 0.00] \\
\text{1SPD 2RSET3CHG 4next}
\end{align*}
\]
9-5  Emergency Stop Function

There are two ways to trigger emergency stop on the ERCX controller. One way is by using the push-button on the TPB. The other is to use the I/O emergency stop input. In either case for safety reasons, a contact B (normally closed) input is used (when the contact is opened, emergency stop is triggered). The ERCX controller can recover from an emergency stop condition without turning off the power so return-to-origin is not necessary.

This section explains how to initiate and recover from an emergency stop using the TPB.

9-5-1 Initiating an emergency stop

If for any reason you want to immediately stop the robot while operating it with the TPB, press the emergency stop button on the TPB. The emergency stop button locks in the depressed position, and can be released by turning it to the right.

In emergency stop, the robot assumes a “free” state so that commands initiating robot motion (for example, return-to-origin command) cannot be executed.

9-5-2 Recovering from an emergency stop

When recovery from an emergency stop is required during TPB operation, that procedure automatically appears on the TPB. Follow those instructions to reset the emergency stop condition.

Recovery from an emergency stop is required during TPB operation when you are going to:

- Perform return-to-origin.
- Run step operation.
- Run automatic operation.
- Edit point data using teaching playback.
- Exit the direct teaching mode.

The following steps explain the procedure for running step operation after emergency stop. As this example shows, the emergency stop condition cannot be cancelled by just releasing the emergency stop button.

1) Press \[RUN\] to start operation.

2) Following the message displayed on the screen, release the emergency stop button.
3) After the emergency stop is released, a message appears asking whether to turn the servo on. To turn the servo on, press [F1] (yes). To leave the servo off, press [F2] (no).


5) Operation starts when [F1] (yes) was pressed in step 4. If [F2] (no) was pressed, the screen returns to step 1.

⚠️ CAUTION

When the SERVICE mode function is enabled, the following safety control will function. (See “10-4 SERVICE mode function.”)

- If the hold-to-run function is enabled, the robot stops upon releasing [F1] (yes) in step 4 in “SERVICE mode state”.
9-6 Displaying the Memory I/O Status

The memory I/O status can be displayed on the screen.

1) On the initial screen, press [F2] (OPRT).

2) Press [F2] (STEP) or [F3] (AUTO).
The STEP or AUTO mode screen appears. The following steps are explained using the STEP mode screen.

3) Press [F4] (next) to change the menu display and then press [F3] (MIO).

4) The I/O status of each memory is displayed.
From the left, the top line shows the status from 115 to 100, the middle line from 131 to 116, and the bottom line from 147 to 132.

5) To return to the previous screen, press [ESC].
9-7 Displaying the Variables

The values of point data variable "P", counter array variable "C" and counter variable "D" can be displayed on the TPB screen. This monitor function can be used when the ERCX controller version is 13.23 or later and the TPB version is 12.18 or later.

1) On the initial screen, press \[F2\] (OPRT).

2) Press \[F2\] (STEP) or \[F3\] (AUTO). The following explains the procedure for displaying the variables on the screens in step operation.

3) Press \[F4\] (next) to change the menu display and then press \[F1\] (VAL).

4) Continue to indicate the value of each variable. The item enclosed by brackets \[\] is an array element number selected with the CSEL statement.

5) To return to the previous screen, press \[ESC\].

6) To display the variable of another task in a multi-task program, press \[F3\] (CHGT) in step 3 so that the task is switched before pressing \[F1\] (VAL).
Chapter 10 OTHER OPERATIONS

The TPB has many convenient functions in addition to those already covered. For example, memories can be initialized, and options such as memory cards can be used. This chapter will describe these additional functions.
10-1 Initialization

Initializing the programs and points erases all the program data and point data currently stored in the controller. Initializing the parameters resets the parameters to their initial values.

1) On the initial screen, press [F3] (SYS).

2) Next, press [F3] (INIT).

3) Select the data to be initialized.
   To initialize the program data, press [F1] (PGM).
   To initialize the point data, press [F2] (PNT).
   To initialize the parameter data, press [F3] (PRM).
   To initialize all of the program, point and parameter data, press [F4] (ALL).

4) If [F3] (PRM) or [F4] (ALL) was selected in step 3, the robot type must be specified.
   Enter the robot number with the number keys and then press \(\Rightarrow\).
   To find the robot number, see "15-1-2 Robot number list".

5) If a robot with multiple lead lengths available was selected in step 4, the lead length selection screen appears.
   Press [F1] to [F3] to select the lead length of the robot connected to the controller.
   If 4 or more lead lengths are available for the robot, then press \(\Rightarrow\) (next) to go to the next menu display.
6) A confirmation message appears after selecting the lead length.
Make sure the lead length is correct and press \[ F1 \] (yes).
To select another lead length, press \[ F2 \] (no).

7) Next, enter the robot stroke length.
Enter the stroke length with the number keys and then press стрелка вниз.

8) Finally, enter the robot payload.
Enter the payload with the number keys and then press стрелка вниз.

9) A confirmation message appears on the screen.
To execute the initialization, press \[ F1 \] (yes).
To cancel the initialization, press \[ F2 \] (no).

10) When initialization is complete, the screen returns to step 3.
10-2  DIO Monitor Display

Data indicating whether the I/O signals are on or off can be displayed on the screen. The operation procedure is explained below.

10-2-1 Display from the monitor menu

1) On the initial screen, press \[F4\] (MON).

2) The ON/OFF status of I/O signals is displayed.

For information about what the display shows, refer to "4-3-4 DIO monitor screen".

3) To return to the initial screen, press \[ESC\].
10-2-2 Display from the DIO key operation

1) Hold down the DIO key.

2) The ON/OFF status of I/O signals is displayed as long as the key is held down.
   For information about what the display shows, refer to "4-3-4 DIO monitor screen".

3) Releasing the key returns the screen to the previous screen.

--- CAUTION ---

The DIO Monitor key does not function during system operation.

10-3 System Information Display

1) On the initial screen, press the ESC key.

2) The controller version number, TPB version number, and robot type are displayed. The screen returns to the initial screen after approximately two seconds.
10-4 SERVICE mode function

The SERVICE mode function is explained in this section.

The robot operator or others sometimes need to enter the hazardous area in the robot safety enclosure and move the robot to perform maintenance or adjustment while using the TPB. This situation is referred to as "SERVICE mode state" and requires extra caution. Limits should be placed on controller operation at this time to ensure operator safety.

A safety function called "SERVICE mode function" places limits on controller operation when in "SERVICE mode state".

When the SERVICE mode function is enabled, the ERCX controller constantly monitors status to check whether "SERVICE mode state" occurs. In "SERVICE mode state", the SERVICE mode function does the following:

• Limits command input from any device other than the TPB.
• Limits robot movement speed.
• Prohibits automatic operation and step operation.
• Enables hold-to-run function.

The controller recognizes "SERVICE mode state" when the SERVICE mode function is enabled and the SERVICE mode input (SVCE) is OFF (contact is open). (See "3-2-3 SERVICE mode input (SVCE)".)

NOTE
The SERVICE mode function is protected by a password so that the settings cannot be changed easily.
10-4-1 Safety settings for SERVICE mode

Safety controls that work in "SERVICE mode state" are explained in detail below.

- **Limiting command input from any device other than TPB**

  When the operator is working within the robot safety enclosure using the TPB, permitting any command input from devices (such as via I/O) other than the TPB is very hazardous to the TPB operator.

  (For example, a hazardous situation may occur if someone outside the safety enclosure runs an automatic operation start command (AUTO-R) without letting the TPB operator know about it.)

  To avoid this kind of hazard, the TPB can only be used to operate the robot in "SERVICE mode state", and all other device command inputs are disabled.

  However, this limitation can be cancelled even in "SERVICE mode state" under the user’s responsibility.

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Only commands input from the TPB are permitted in SERVICE mode state.</td>
</tr>
<tr>
<td>1</td>
<td>Only commands input from the TPB and parallel I/O are permitted in SERVICE mode state.</td>
</tr>
<tr>
<td>2</td>
<td>Only commands input from the TPB and option unit are permitted in SERVICE mode state.</td>
</tr>
<tr>
<td>3</td>
<td>Command inputs are not limited even in SERVICE mode state.</td>
</tr>
</tbody>
</table>

- **Limiting the robot movement speed**

  Moving the robot at a high speed while an operator is working within the robot safety enclosure is very dangerous to that operator. Setting the robot movement speed to a safety speed of 250mm/s or less is advisable because most robot operation while the operator is working within the safety enclosure is for maintaining or adjusting the robot. In view of this, the robot movement speed in "SERVICE mode state" is limited to below 3% of maximum speed.

  However, this speed limitation can be cancelled even in "SERVICE mode state" under the user’s responsibility.

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The robot movement speed is limited to 3% or less of maximum speed in SERVICE mode state.</td>
</tr>
<tr>
<td>1</td>
<td>The robot movement speed is not limited even in SERVICE mode state.</td>
</tr>
</tbody>
</table>
### Prohibiting the automatic operation and step operation
Running an automatic operation or step operation while an operator is working within the robot safety enclosure is very dangerous to that operator.
(For example, when the operator is in the safety enclosure, a hazardous situation may occur if someone runs a robot program without letting the operator know about it.)
To avoid this kind of hazard, automatic operation and step operation are basically prohibited in "SERVICE mode state".
However, this limitation can be cancelled even in "SERVICE mode state" under the user’s responsibility.

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Automatic operation and step operation are prohibited in SERVICE mode state.</td>
</tr>
<tr>
<td>1</td>
<td>Automatic operation and step operation are permitted even in SERVICE mode state.</td>
</tr>
</tbody>
</table>

### Hold-to-run function
If the robot continues to move while an operator is working within the robot safety enclosure without using the TPB, the operator may be exposed to a dangerous situation.
(For example, a hazardous situation may occur if the operator working within the safety enclosure should trip or fall by accident and blackout.)
To prevent this kind of hazard, the Hold-to-Run function allows the robot to move only during the time that the TPB key is kept pressed in "SERVICE mode state".
However, this function can be cancelled even in "SERVICE mode state" under the user’s responsibility.

<table>
<thead>
<tr>
<th>Setting value</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Hold-to-run function works in SERVICE mode state.</td>
</tr>
<tr>
<td>1</td>
<td>Hold-to-run function does not work even in SERVICE mode state.</td>
</tr>
</tbody>
</table>

⚠️ **CAUTION**
The above safety controls can be cancelled in part or in whole under the user’s responsibility. However, extra caution must be taken to maintain safety since hazardous situations may occur.

💡 **NOTE**
When parameter initialization is performed, all safety control settings are initialized. (All settings will be set to "0"). However, the SERVICE mode function setting will not change even after parameter initialization.
10-4-2 Enabling/disabling the SERVICE mode function

To enable or disable the SERVICE mode function, follow these steps.

1) On the initial screen, press \[F3\] (SYS).

2) Press \[F4\] (next) to change the menu display and then press \[F1\] (SAFE).

3) The password request screen appears. Enter the password and then press \[\rightarrow\].

4) When the password is correct, the screen shown on the right appears. Press \[F2\] (SVCE) here.

5) Press \[F1\] (SET).

6) The current SERVICE mode function setting appears.
   To disable the SERVICE mode function, enter 0 with the number key. To enable it, enter 1. Then, press \[\rightarrow\].
7) When writing is complete, the screen returns to step 6.

Note: The password is identical to the ERCX controller's version number. For example, if the controller version is 13.13, enter 13.13 as the password. Once the password is accepted, it will not be requested unless the TPB is disconnected from the controller or the controller power is turned off.
10-4-3 Setting the SERVICE mode functions

1) On the initial screen, press [F3] (SYS).

2) Press [F4] (next) to change the menu display and then press [F1] (SAFE).

3) The password request screen appears. Enter the password and then press " disregard the password.

4) When the password is correct, the screen shown on the right appears. Press [F2] (SVCE) here.

5) Select the item whose setting you want to change.
   To change the setting that limits the operation device, press [F2] (DEV).
   To change the setting that limits the speed, press [F3] (SPD).
   To change the setting that limits step operation and automatic operation, press [F4] (next) and then press [F1] (RUN).
   To change the setting for the hold-to-run function, press [F4] (next) and then press [F2] (HtoR).

6) The current setting for the selected item appears. To change the setting, enter the data with the number key and then press " disregard the password.
7) When the setting has been changed, the memory write screen appears.
   To save the change permanently (retain the change even after the controller power is turned off), press [F1](SAVE).
   To save the change temporarily (retain the change until the power is turned off), press [F2](CHG).
   To cancel changing the setting, press [F3](CANCEL).

8) When writing is complete, the screen returns to step 6.

```
[System-Safe-SVCE-DEV]
data = 1
PB/DI valid
[SYS-SAFE-SVCE-DEV]
data = 1
PB/DI valid
```

NOTE
The password is identical to the ERCX controller’s version number. For example, if the controller version is 13.13, enter 13.13 as the password. Once the password is accepted, it will not be requested unless the TPB is disconnected from the controller or the controller power is turned off.
10-5 System utilities

10-5-1 Viewing hidden parameters

Parameters hidden in the normal state can be viewed. Use extra caution to avoid accidentally changing the parameters when these hidden parameters are displayed.

1) On the initial screen, press F3 (SYS).

2) Press F4 (next) to change the menu display and then press F3 (UTL).

3) Press F1 (HDPR) here.

4) A confirmation message appears.
   To permit display of the hidden parameters, press F1 (yes)
   To hide the hidden parameters, press F2 (no).

5) The screen returns to step 3.
   Display of hidden parameters is permitted until you press F1 (HDPR) and then F2 (no), or the ERCX controller is turned off, or until the TPB is disconnected.

---

**NOTE**

The hidden parameter display is also permitted by turning on the power to the controller while holding down the ESC key on the TPB, or by connecting the TPB to the controller while holding down the ESC key.
10-6 Using a Memory Card

A memory card can be used with the TPB to back up the data in the ERCX controller. Refer to "16-2-1 Memory card" for the procedure for handling a memory card and for the number of data that can be stored.

10-6-1 Saving controller data to a memory card

1) Insert the memory card into the TPB.

2) On the initial screen, press **F3** (SYS).

3) Next, press **F2** (B.UP).

4) Press **F1** (SAVE).

5) Specify the format to save the data.
   Press **F1** (normal) to save the data in the ERCX standard format.
   Press **F2** (compati) to save the data in SRC and SRCA compatible format.

6) Specify the save area on the memory card.
   Enter the save area with the number keys and press ➡️.
7) The saved status of data on the memory card can be checked by pressing [F1](ID) in step 6. To check the saved status in AREA 3 onward, press [STEP UP] or [STEP DOWN] to scroll the screen. To return to the screen in step 6, press [ESC].

8) If data already exists in the area specified in step 6, a confirmation message appears. To overwrite the data in the selected area, press [F1] (yes). To change the selected area, press [F2] (no).

9) Set an ID number for the data being saved. Using the number keys (0 to 9), the "." (minus) key, and the "," (period) key, enter a number of up to eight characters and then press .


11) This screen is displayed while the data is being saved.

12) When saving is finished, the screen returns to step 4.

⚠️ CAUTION
Never eject the memory card during saving of data. Do not leave the memory card inserted into the TPB when not in use. This shortens the backup battery life.
10-6-2 Loading data from a memory card

1) Insert the memory card into the TPB.

2) On the initial screen, press [F3] (SYS).


4) Press [F2] (LOAD).

5) Specify the load area in the memory card. Enter the load area with the number keys and press $\Rightarrow$.

6) The saved status of data on the memory card can be checked by pressing [F1] (ID) in step 5. To check the saved status in AREA 3 onward, press [STEP UP] or [STEP DOWN] to scroll the screen. To return to the screen in step 5, press [ESC].
7) When the load area was selected in step 5, the data load screen appears. Select the data to be loaded.

To load the program data, press \[F1\] (PGM).
To load the point data, press \[F2\] (PNT).
To load the parameter data, press \[F3\] (PRM).
To load all of the program, point and parameter data, press \[F4\] (ALL).

8) When \[F1\] (PGM) or \[F2\] (PNT) was selected in step 7, a confirmation message appears asking whether to overwrite the data.

Pressing \[F1\] (yes) overwrites the data only with the same program numbers or point numbers. (The previous data remains if its program or point number differs from the program or point number to be loaded.)

Pressing \[F2\] (no) loads the data after initializing the data in the ERCX controller.

When \[F4\] (ALL) was selected in step 7, all data in the ERCX controller will be initialized and then loaded.

9) A confirmation message appears asking whether to load the data.

To load the data, press \[F1\] (yes)
To cancel, press \[F2\] (no).

10) This screen is displayed while data is being loaded.

11) When data loading is complete, the screen returns to step 7.

**CAUTION**

Never eject the memory card during loading of data.

Do not leave the memory card inserted into the TPB when not in use. This shortens the backup battery life.
10-6-3 Formatting a memory card

1) Insert the memory card into the TPB.

2) On the initial screen, press \[F3\] (SYS).

3) Next, press \[F2\] (B.UP).

4) Press \[F3\] (FMT).

5) A confirmation message appears.
   To format the memory card, press \[F1\] (yes).
   To cancel, press \[F2\] (no).

6) Select the format type.
   To perform the ERCX standard formatting, press \[F1\] (normal).
   To perform the SRC/SRCA compatible formatting, press \[F2\] (compati).

7) This screen is displayed while the memory card is being formatted.

8) When formatting is complete, the screen returns to step 4.

⚠️ CAUTION
Never eject the memory card during formatting.
Do not leave the memory card inserted into the TPB when not in use. This shortens the backup battery life.
10-6-4 Viewing the ID number for memory card data

1) Insert the memory card into the TPB.

2) On the initial screen, press [F3] (SYS).


5) The ID number of each area is displayed on the screen.
   To check the saved status in AREA 3 onward, press [STEP UP] or [STEP DOWN] to scroll the screen. To return to the screen in step 4, press [ESC].
The ERCX controller has a duty (load factor) monitor to allow you to operate the robot under the most optimal conditions. The duty monitor checks the robot's motor load factor and displays it in percent (%) versus the motor rating.

**NOTE**
The duty monitor function can be used when the controller version is 13.50 or later and the TPB version is 12.60 or later.

An overload error might appear if the duty exceeds 100% during robot operation. If this happens, either lower the robot acceleration or maximum speed, or increase the robot stop time (lower the duty ratio). On the other hand, if you want to shorten the cycle time even further, when there is currently no overload, you can raise the acceleration or maximum speed, or shorten the robot stop time (raise the duty ratio).

There are the following two methods to measure the duty.

**Method 1:** On the TPB, select DUTY mode and measure the duty during robot movement with a point movement command (ABS-PT, INC-PT) or a program start command (AUTO-R, STEP-R) via the I/O connector.

**Method 2:** Specify an interval in a program in which you want to measure the duty and run the program.

**[Method 1]**
1) Operate the robot with a point movement command (ABS-PT, INC-PT) or a program start command (AUTO-R, STEP-R) via the I/O connector.
2) On the TPB, select DUTY mode.
3) Measure the operation duty.
4) Check the measurement result.

Refer to "10-7-1 Measuring the duty (load factor)" for procedures to start and stop duty measurement and check the result.

**NOTE**
In method 1, the duty cannot be measured during robot movement by the TPB (RS-232C).
[Method 2]
1) Add the robot language command "DUTY 1" to the beginning of the interval in a program in which you want to measure the duty and also add the robot language command "DUTY 0" to the end of the interval.

```
005:  DO 0,1
006:  WAIT 1,1
007:  DO 0,0
008:  TIMR 100
009:  DUTY 1 ← Start operation duty measurement
010:  DO 0,0
011:  WAIT 0,1
012:  MOVA 2,100
013:  DO 0,1
014:  WAIT 1,1
015:  DO 1,0
016:  TIMR 100
017:  DUTY 0 ← Stop operation duty measurement
018:  DO 0,0
```

2) Run the program including the operation duty measurement interval.
3) Stop (end) the program.
4) On the TPB, select DUTY mode and check the measurement result.

Refer to "10-7-1 Measuring the duty (load factor)" for the procedure to check the measurement result.
10-7-1 Measuring the duty (load factor)

NOTE
The duty monitor function can be used when the controller version is 13.50 or later and the TPB version is 12.60 or later.

1) Press [F4] (MON) on the TPB initial screen while moving the robot with a point movement command (ABS-PT, INC-PT) or a program start command (AUTO-R, STEP-R) via the I/O connector.

2) Next, press [F2] (DUTY).

3) Press [F1] (RUN) to start measuring the operation duty.

4) Press [F2] (STOP) to stop measuring the operation duty.

5) Next, press [F3] (RSLT) to display the measurement result on the TPB screen.

6) The operation duty value appears as a percent on the TPB screen.

NOTE
The operation duty can also be monitored while the program is being executed with a program command. For more information, see "10-7 Duty (load factor) monitor". The method for displaying the measurement result is the same as described above.
Chapter 11 COMMUNICATION WITH PC

The ERCX controller allows you to edit the program data and point data or control the robot operation using a PC (personal computer) by RS-232C communication instead of using the TPB. This chapter describes how to set the communication parameters required to communicate between the PC and the ERCX controller, and also explains the communication command specifications.
The communication parameters on the PC should be set as follows. For the setting procedure, refer to the computer operation manual.

- **Baud rate**: 9600 bps
- **Data bit length**: 8 bits
- **Stop bit length**: 1 bit
- **Parity check**: On
- **Parity setting**: Odd
- **Control method**: XON/XOFF software control (X parameter) (Effective)
- **Communication method**: Full duplex
- **Sync method**: Asynchronous method
- **Return key transmission**: CR code
- **CR code reception**: For CR/LF reception: Return + line feed
  For CR reception: Return

* If the above parameter settings are not possible due to your equipment specifications, the robot controller settings can be changed by changing PRM47 (communication parameter settings) from the TPB.

**PRM47 settings (default value: 0)**

<table>
<thead>
<tr>
<th>bit</th>
<th>Function</th>
<th>Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>15 to 9</td>
<td>Reserved</td>
<td>Always set to 0.</td>
</tr>
<tr>
<td>8</td>
<td>Termination code</td>
<td>0: CR+LF ; 1: CR</td>
</tr>
<tr>
<td>7 to 4</td>
<td>Transmission speed</td>
<td>0: 9600bps ; 1: 300bps ; 2: 600bps ; 3: 1200bps ; 4: 2400bps ; 5: 4800bps ; 6: 9600bps ; 7 to 15: Cannot be set</td>
</tr>
<tr>
<td>3</td>
<td>Data bit length</td>
<td>0: 8 bits ; 1: 7 bits</td>
</tr>
<tr>
<td>2</td>
<td>Stop bit length</td>
<td>0: 1 bit ; 1: 2 bits</td>
</tr>
<tr>
<td>1 to 0</td>
<td>Parity check</td>
<td>0: Odd ; 1: Even ; 2 to 3: Non</td>
</tr>
</tbody>
</table>

Example: To set the data bit length to "7 bits" and the parity check to "Non", enter "10" for PRM47, which is given by 0000000000001010 (binary) = 10 (decimal)

⚠️ **CAUTION**

Be sure to use a cable which conforms to specifications listed in "11-2 Communication Cable Specifications". The settings will be invalid if other cables such as POPCOM communication cables or those having different specifications are used.
After changing the parameters, turn the power off and then turn it on again to enable the settings.
The TPB can be used even if the parameters have been changed.
11-2 Communication Cable Specifications

**CAUTION**

- Pins 10, 12, 18 and 21 of the controller's connector are specifically used for TPB connection. To avoid possible accidents do not connect other inputs to these pins.
- When using optional POPCOM software, make connections while referring to the POPCOM operation manual since it shows the different connection specifications.
- The personal computer may have its own connector specifications, so be sure to check the computer operation manual to ensure the connections are correct.

11-2-1 Connecting to the computer with a 25-pin D-sub connector

**Connector model**

- Mating connector type No.: XM2A-2501 (OMRON) or equivalent type
- Mating connector cover type No.: XM2S-2511 (OMRON) or equivalent type

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Pin No.</th>
<th>Controller side</th>
<th>Computer side</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F.G</td>
<td>1</td>
<td></td>
<td>1 F.G</td>
</tr>
<tr>
<td>TXD</td>
<td>2</td>
<td></td>
<td>2 TXD (SD)</td>
</tr>
<tr>
<td>RXD</td>
<td>3</td>
<td></td>
<td>3 RXD (RD)</td>
</tr>
<tr>
<td>RTS</td>
<td>4</td>
<td></td>
<td>4 RTS (RS)</td>
</tr>
<tr>
<td>CTS</td>
<td>5</td>
<td></td>
<td>5 CTS (CS)</td>
</tr>
<tr>
<td>D.G</td>
<td>7</td>
<td></td>
<td>7 D.G (SG)</td>
</tr>
</tbody>
</table>

11-2-2 Connecting to the computer with a 9-pin D-sub connector

**Connector model (controller side)**

- Mating connector type No.: XM2A-2501 (OMRON) or equivalent type
- Mating connector cover type No.: XM2S-2511 (OMRON) or equivalent type

**Connector model (computer side)**

- Mating connector type No.: XM2D-0901 (OMRON) or equivalent type
- Mating connector cover type No.: XM2S-0913 (OMRON) or equivalent type

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Pin No.</th>
<th>Controller side</th>
<th>Computer side</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F.G</td>
<td>1</td>
<td></td>
<td>SHELL F.G</td>
</tr>
<tr>
<td>TXD</td>
<td>2</td>
<td></td>
<td>1 DCD (CD)</td>
</tr>
<tr>
<td>RXD</td>
<td>3</td>
<td></td>
<td>2 RXD (RD)</td>
</tr>
<tr>
<td>RTS</td>
<td>4</td>
<td></td>
<td>3 TXD (SD)</td>
</tr>
<tr>
<td>CTS</td>
<td>5</td>
<td></td>
<td>4 DTR (ER)</td>
</tr>
<tr>
<td>D.G</td>
<td>7</td>
<td></td>
<td>5 D.G (SG)</td>
</tr>
</tbody>
</table>

The "SHELL" is the metallic casing of the connector.

**NOTE**

- Transmission stops while CTS on the controller side is off. If a robot alarm is issued while CTS is on, the controller keeps sending the message.
- RTS on the controller side is always on.
11-3 Communication Command Specifications

On the ERCX controller, a command interface resembling the BASIC programming language is provided as standard, to facilitate easy communication with a PC. Communication commands are divided into the following four categories:

1. Robot movements
2. Data handling
3. Utilities
4. Special codes

Format: (except for special codes)

```
@<operation code> [<operand 1>][,<operand 2>][,<operand 3>]c/r l/f
```

- Basically, all of the commands begin with the start code '@' (=40H) and end with the code c/r (=0DH) l/f (=0AH). These two codes signal the controller that the statements between them constitute one command line. (The special codes are the only ones that do not require a start or an end code.)

- A communication command is basically composed of an operation code and an operand. Depending on the command statement, either no operand is used, or up to three operands are used. Items in [ ] (brackets) can be omitted.

- The character codes used in the ERCX series, are the JIS8 unit system codes (ASCII codes with katakana characters added). Input characters can be upper case or lower case.

- One or more space must be inserted between the operation code and the operand.

- Items with the < > marks should be specified by the user. Check the description of each communication command and enter the appropriate data. (Refer to "11-5 Communication Command Description".)

- When two or more operands are entered, insert a comma (,) between them.

An example is shown below.

**Transmission example**

```
@MOVI 123,100c/r l/f
```

Start code
Opcode (Operation code)
Space
Comma
Operand 2
Operand 1
## 11-4 Communication Command List

### 1. Robot movement

<table>
<thead>
<tr>
<th>No.</th>
<th>Operation code</th>
<th>Operand 1</th>
<th>Operand 2</th>
<th>Operand 3</th>
<th>Command details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ORG</td>
<td></td>
<td></td>
<td></td>
<td>Returns to origin</td>
</tr>
<tr>
<td>2.</td>
<td>ORGN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>RESET</td>
<td></td>
<td></td>
<td></td>
<td>Resets program</td>
</tr>
<tr>
<td>4.</td>
<td>RUN</td>
<td></td>
<td></td>
<td></td>
<td>Starts automatic operation</td>
</tr>
<tr>
<td>5.</td>
<td>SRUN</td>
<td>0</td>
<td></td>
<td>1</td>
<td>Starts step operation</td>
</tr>
<tr>
<td>6.</td>
<td>SRVO</td>
<td></td>
<td></td>
<td>1</td>
<td>Turns servo on</td>
</tr>
<tr>
<td>8.</td>
<td>MOV D</td>
<td>X-axis position (mm)</td>
<td>speed</td>
<td></td>
<td>Directly moves to specified position</td>
</tr>
<tr>
<td>9.</td>
<td>MOVA</td>
<td>point number</td>
<td>speed</td>
<td></td>
<td>Moves to specified position</td>
</tr>
<tr>
<td>10.</td>
<td>MOVI</td>
<td>point number</td>
<td>speed</td>
<td></td>
<td>Moves specified movement amount</td>
</tr>
<tr>
<td>11.</td>
<td>MOV F</td>
<td>point number</td>
<td>D1 number</td>
<td>0 or 1</td>
<td>Moves in response to general-purpose input</td>
</tr>
<tr>
<td>12.</td>
<td>EO*</td>
<td>output number</td>
<td>0</td>
<td>1</td>
<td>Turns off general-purpose output or memory output</td>
</tr>
<tr>
<td>13.</td>
<td>WAIT</td>
<td>input number</td>
<td>0 or 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>TIMR</td>
<td>time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>P</td>
<td>point number</td>
<td></td>
<td></td>
<td>Defines point variable P</td>
</tr>
<tr>
<td>16.</td>
<td>P+</td>
<td>point number</td>
<td></td>
<td></td>
<td>Adds 1 to point variable P</td>
</tr>
<tr>
<td>17.</td>
<td>P-</td>
<td>point number</td>
<td></td>
<td></td>
<td>Subtracts 1 from point variable P</td>
</tr>
<tr>
<td>18.</td>
<td>MOV M</td>
<td>pallet work position</td>
<td>speed</td>
<td>number of rows</td>
<td>number of columns</td>
</tr>
<tr>
<td>19.</td>
<td>MSEL</td>
<td>pallet number</td>
<td>array element number</td>
<td></td>
<td>specifies array element of counter array variable C</td>
</tr>
<tr>
<td>20.</td>
<td>CSEL</td>
<td>array element number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21.</td>
<td>C</td>
<td>counter value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22.</td>
<td>C+</td>
<td>addition value</td>
<td></td>
<td></td>
<td>Adds specified value to counter array variable C</td>
</tr>
<tr>
<td>23.</td>
<td>C-</td>
<td>subtraction value</td>
<td></td>
<td></td>
<td>Subtracts specified value from counter array variable C</td>
</tr>
<tr>
<td>24.</td>
<td>D</td>
<td>counter value</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25.</td>
<td>D+</td>
<td>addition value</td>
<td></td>
<td></td>
<td>Adds specified value to counter variable D</td>
</tr>
<tr>
<td>26.</td>
<td>D-</td>
<td>subtraction value</td>
<td></td>
<td></td>
<td>Subtracts specified value from counter variable D</td>
</tr>
<tr>
<td>27.</td>
<td>D*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28.</td>
<td>SHIFT</td>
<td>point number</td>
<td></td>
<td></td>
<td>Performs point data shift</td>
</tr>
</tbody>
</table>
## 2. Data handling

<table>
<thead>
<tr>
<th>No.</th>
<th>Operation code</th>
<th>Operand 1</th>
<th>Operand 2</th>
<th>Operand 3</th>
<th>Command details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>?POS</td>
<td></td>
<td></td>
<td></td>
<td>Reads current position</td>
</tr>
<tr>
<td>2</td>
<td>?NO</td>
<td></td>
<td></td>
<td></td>
<td>Reads current program number</td>
</tr>
<tr>
<td>3</td>
<td>?SNO</td>
<td></td>
<td></td>
<td></td>
<td>Reads current step number</td>
</tr>
<tr>
<td>4</td>
<td>?TNO</td>
<td></td>
<td></td>
<td></td>
<td>Reads current task number</td>
</tr>
<tr>
<td>5</td>
<td>?PNO</td>
<td></td>
<td></td>
<td></td>
<td>Reads current point number</td>
</tr>
<tr>
<td>6</td>
<td>?TSTP</td>
<td>program number</td>
<td></td>
<td></td>
<td>Reads total number of steps in specified program</td>
</tr>
<tr>
<td>7</td>
<td>?MEM</td>
<td>history number</td>
<td>display count</td>
<td></td>
<td>Reads alarm history</td>
</tr>
<tr>
<td>8</td>
<td>?VER</td>
<td></td>
<td></td>
<td></td>
<td>Reads ROM version number</td>
</tr>
<tr>
<td>9</td>
<td>?ROBOT</td>
<td></td>
<td></td>
<td></td>
<td>Reads robot number</td>
</tr>
<tr>
<td>10</td>
<td>?TLOOK</td>
<td></td>
<td></td>
<td></td>
<td>Reads total operation time of controller</td>
</tr>
<tr>
<td>11</td>
<td>?ALM</td>
<td>history number</td>
<td>display count</td>
<td></td>
<td>Reads alarm history</td>
</tr>
<tr>
<td>12</td>
<td>?EMG</td>
<td></td>
<td></td>
<td></td>
<td>Confirms emergency stop status</td>
</tr>
<tr>
<td>13</td>
<td>?SRVO</td>
<td></td>
<td></td>
<td></td>
<td>Confirms servo status</td>
</tr>
<tr>
<td>14</td>
<td>?ORG</td>
<td></td>
<td></td>
<td></td>
<td>Confirms return-to-origin status</td>
</tr>
<tr>
<td>15</td>
<td>?MODE</td>
<td></td>
<td></td>
<td></td>
<td>Confirms operation mode</td>
</tr>
<tr>
<td>16</td>
<td>?PVA</td>
<td>input number</td>
<td></td>
<td></td>
<td>Reads current point variable P</td>
</tr>
<tr>
<td>17</td>
<td>?DI</td>
<td>output number</td>
<td></td>
<td></td>
<td>Reads general-purpose input or memory input status</td>
</tr>
<tr>
<td>18</td>
<td>?DO</td>
<td>parameter number</td>
<td></td>
<td></td>
<td>Reads general-purpose output or memory output status</td>
</tr>
<tr>
<td>19</td>
<td>?PROM</td>
<td>parameter number</td>
<td>parameter number</td>
<td></td>
<td>Reads specified parameter data</td>
</tr>
<tr>
<td>20</td>
<td>?P</td>
<td>point number</td>
<td>point number</td>
<td></td>
<td>Reads specified multiple point data</td>
</tr>
</tbody>
</table>
| 21  | ?READ          | program number | step number | number of steps | Reads specified program data
|     |                | PGM   |           |           | Reads all program data |
|     |                | PNT   |           |           | Reads all point data |
|     |                | PRM   |           |           | Batch reads all program, point and parameter data |
|     |                | ALL   |           |           | Reads input/output information |
|     |                | DIO   |           |           | Reads memory input/output information |
|     |                | MIO   |           |           | Reads registered program information |
|     |                | INF   |           |           | |
| 22  | WRITE          | PGM   |           |           | Writes program data |
|     |                | PNT   |           |           | Writes point data |
|     |                | PRM   |           |           | Writes parameter data |
|     |                | ALL   |           |           | Batch writes program, point and parameter data |
|     |                |       |           |           | |
| 23  | ?ERR           | history number | display count |           | Reads error history |
| 24  | ?MAT           | pallet number |           |           | Reads matrix definition contents |
| 25  | ?MSEL          |           |           |           | Reads currently specified matrix number |
| 26  | ?CSSEL         |           |           |           | Reads currently specified element number of counter array variable C |
| 27  | ?C            | array element number |           |           | Reads current counter array variable C |
| 28  | ?DO            |           |           |           | Reads current counter variable D |
| 29  | ?SHIFT         |           |           |           | Reads current shift data |
3. Utility

<table>
<thead>
<tr>
<th>No.</th>
<th>Operation code</th>
<th>Operand 1</th>
<th>Operand 2</th>
<th>Operand 3</th>
<th>Command details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>INIT</td>
<td>PGM</td>
<td>robot number</td>
<td></td>
<td>Initializes program data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PNT</td>
<td></td>
<td></td>
<td>Initializes point data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PRM</td>
<td></td>
<td></td>
<td>Initializes robot parameters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLOCK</td>
<td></td>
<td></td>
<td>Initializes timer that measures total operation time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ALM</td>
<td></td>
<td></td>
<td>Initializes alarm history</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ERR</td>
<td></td>
<td></td>
<td>Initializes error history</td>
</tr>
<tr>
<td>2.</td>
<td>SWI</td>
<td>program number</td>
<td></td>
<td></td>
<td>Switches program number to be run</td>
</tr>
<tr>
<td>3.</td>
<td>SWITSK</td>
<td>task number</td>
<td>step number</td>
<td></td>
<td>Switches task number to be run</td>
</tr>
<tr>
<td>4.</td>
<td>SINS</td>
<td>program number</td>
<td>step number</td>
<td></td>
<td>Inserts one program step</td>
</tr>
<tr>
<td>5.</td>
<td>SDEL</td>
<td>program number</td>
<td>step number</td>
<td></td>
<td>Deletes one program step</td>
</tr>
<tr>
<td>6.</td>
<td>SMOD</td>
<td>program number</td>
<td>step number</td>
<td></td>
<td>Modifies one program step</td>
</tr>
<tr>
<td>7.</td>
<td>COPY</td>
<td>program number</td>
<td>program number</td>
<td></td>
<td>Copies program</td>
</tr>
<tr>
<td>8.</td>
<td>PDEL</td>
<td>point number</td>
<td>number of points</td>
<td></td>
<td>Deletes specified program</td>
</tr>
<tr>
<td>9.</td>
<td>PDDEL</td>
<td></td>
<td></td>
<td></td>
<td>Deletes point data</td>
</tr>
</tbody>
</table>

4. Special codes

<table>
<thead>
<tr>
<th>No.</th>
<th>Code</th>
<th>Command details</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>^C (=03H)</td>
<td>Interrupts RUN, SRUN, ORG, etc.</td>
</tr>
<tr>
<td>2.</td>
<td>^Z (=1AH)</td>
<td>Ends data transmission</td>
</tr>
</tbody>
</table>
11-5 Communication Command Description

11-5-1 Robot movements

(1) @ORG
     @ORGN
This command performs return-to-origin when the search method is selected as the origin detection method and outputs the machine reference value when return-to-origin is completed normally. When the mark method is selected, this command checks the return-to-origin status and outputs the machine reference value if return-to-origin is complete but issues an error if incomplete.

Transmission example: @ORG c/r l/f .............................. Performs return-to-origin.

Response example 1: OK c/r l/f
                        52% c/r l/f
                        OK c/r l/f

Response example 2: NG c/r l/f .............................. The robot is running.
                       31: running   c/r l/f    Execute the command again after stopping the robot.

Response example 3: NG c/r l/f .............................. Return-to-origin by the mark method is not complete.

NOTE
Once return-to-origin is performed after the robot cable and absolute battery are connected, there is no need to repeat it even when the controller is turned off. (As an exception, return-to-origin becomes incomplete if the absolute backup function is disabled or a parameter relating to the origin is changed. Return-to-origin must be performed again in that case.)

CAUTION
When performing return-to-origin by the stroke-end detection method, do not interrupt the return-to-origin operation while detecting the origin (while contacting the mechanical limit). Otherwise, the operation will stop due to a controller overload alarm and the power will need to be turned on again.

CAUTION
If you must repeat return-to-origin using the stroke-end detection method, wait at least 5 seconds before repeating it.
(2) @RESET
This returns the program execution step to the first step of the program selected with the '@SWI' statement, and turns all general-purpose outputs (DO0 to DO12) and memory output off. The "current position in the program" used as a reference for the relative movement command (MOVI) is initialized to the current position of the robot, and the point variable P is also cleared to 0.

Transmission example : @RESET c/r l/f
Response example 1 : OK c/r l/f
Response example 2 : NG c/r l/f ............................. The robot is running.
                           31: running   c/r l/f  Execute the command again
                           after stopping the robot.

* When PRM33 ("operation at return-to-origin complete" parameter) is set to 1 or 3, DO4 does not turn off even if the @RESET command is executed. Likewise, when PRM46 (servo status output parameter) is set to 1, DO7 does not turn off even if the @RESET command is executed.

(3) @RUN
This executes a program all the way to the last step.
In the case of a multi-task program, all tasks are executed.

Transmission example : @RUN c/r l/f
Response example 1 : STOP c/r l/f ............................. The last step of the program
                           60: program end   c/r l/f  has been executed.
Response example 2 : NG c/r l/f ............................. Return-to-origin has not been
                           32: origin incomplete c/r l/f  performed. Execute the command again after performing

⚠️ CAUTION
When using an endless program (program that unconditionally returns to the head of the program at the last step), there will be no response.

(4) @SRUN
This executes only one step of a program.
In the case of a multi-task program, the selected task is executed.

Transmission example : @SRUN c/r l/f
Response example 1 : OK c/r l/f
Response example 2 : STOP c/r l/f ............................. The last step of the program
                           60: program end   c/r l/f  has been executed.
Response example 3 : NG c/r l/f ............................. Return-to-origin has not been
                           32: origin incomplete c/r l/f  performed. Execute the command again after performing
                           return-to-origin.
(5) @SRVO <servo status>
Turns the servo on or off.

Servo status : Specify 1 to turn the servo on or 0 to turn it off.
Transmission example : @SRVO 0 c/r l/f ......................... Turns the servo off.
Response example : OK c/r l/f

(6) @X+, (@X-)
@X+ moves the robot to the + side and @X- to the - side based on the following equation.

Movement distance = 1 × (PRM26/100) (mm)  PRM26: Teaching movement data (%)

⚠️ CAUTION
If the robot uses a rotary axis, then movement distance is expressed in deg. (degrees).

(7) @XINC, (@XDEC)
@XINC moves the robot to the + side and @XDEC to the - side at a speed calculated by the equation below. The robot continues moving until the ^C code is input or the robot reaches the soft limit.

Movement speed = 100 × (PRM26/100) (mm/sec.)  PRM26: Teaching movement data (%)

⚠️ CAUTION
If the robot uses a rotary axis, the movement distance is expressed in deg/sec.

⚠️ CAUTION
The soft limit will not work unless return-to-origin has been performed.

(8) @MOVD <X-axis position (mm)>,<speed>
Moves the robot to a specified coordinate position.

X-axis position : Directly specify the target position to move the robot to. If the robot uses a rotary axis, the coordinate position is expressed in deg. (degrees).
Speed : The speed can be set to any level between 1 and 100. If PRM30 (Maximum program speed) is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).
Transmission example : @MOVD 50.37,100 c/r l/f .......... Moves the robot to the position at 50.37 mm, at 100% speed.
Response example 1 : OK c/r l/f
Response example 2 : NG c/r l/f ................................. The target position exceeds the soft limit over 30: soft limit over c/r l/f
(9) @MOVA <point number>,<speed>
Moves the robot to a position specified by a point number at a specified speed.

Point number : This is a number assigned to each point (position data) and can be from 0 to 999 (a total of 1,000 points). Data for the point numbers can be edited with the @WRITE PNT statement. The point variable "P" can also be used.

Speed : The speed can be set to any level between 1 and 100. If PRM30 (Maximum program speed) is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).

Transmission example : @MOVA 123,100 c/r l/f ................ Moves the robot to point 123 at 100% speed.

Response example 1 : OK c/r l/f
Response example 2 : NG c/r l/f ............................... The target position exceeds the soft limit over c/r l/f soft limit. Change the point data or soft limit parameter.

(10) @MOVI <point number>,<speed>
Moves the robot a distance specified by a point number from the current position, at a specified speed.

Point number : This is a number assigned to each point (position data) and can be from 0 to 999 (a total of 1,000 points). Data for the point numbers can be edited with the @WRITE PNT statement. The point variable "P" can also be used.

Speed : The speed can be set to any level between 1 and 100. If PRM30 (Maximum program speed) is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).

Transmission example : @MOVI 123,100 c/r l/f ................ Moves the robot a distance defined by point 123, at 100% speed.

Response example 1 : OK c/r l/f
Response example 2 : NG c/r l/f ............................... The target position exceeds the soft limit over c/r l/f soft limit. Change the point data or soft limit parameter.

⚠️ CAUTION

When movement is interrupted with a stop (^C) statement, the current position in the program stays unchanged so that the movement can be resumed by executing the @MOVI command again. However, if the command is reset, the current position in the program is initialized to the actual robot position.
(11) \texttt{@MOVF <point number>,<DI number>,<DI status>}

This command moves the robot toward a position specified by a point number until a specified DI input condition is met. When the DI condition is met, the robot stops and the command terminates. Even if the DI condition is not met, the command terminates when the target point is reached.

- **Point number**: This is a number assigned to each point (position data) and can be from 0 to 999 (a total of 1,000 points). Data for the point numbers can be edited with the \texttt{@WRITE PNT} statement. The point variable "P" can also be used.

- **DI number**: Specify one of the general-purpose inputs DI0 to DI15.

- **DI status**: Specify 1 (ON) or 0 (OFF) as the input condition.

**Transmission example**: \texttt{@MOVF 2,10,l c/r l/f ...................} Moves to point 2 until DI10 becomes 1 (ON).

**Response example**: \texttt{OK c/r l/f}

\begin{center}
\textbf{CAUTION}
\end{center}

The movement speed is set with PRM9 (MOVF speed) and independent of the PRM30 setting (Maximum program speed).

(12) \texttt{@DO <general-purpose output or memory output number>,<output status>}

Turns a general-purpose output or memory output on or off.

- **Output number**: Specify one of the general-purpose outputs from 0 to 12 (13 points) or one of the memory outputs from 100 to 131 (32 points).

- **Output status**: Specify 1 (ON) or 0 (OFF).

**Transmission example**: \texttt{@DO 3,1 c/r l/f .............................} Turns on general-purpose output 3.

**Response example**: \texttt{OK c/r l/f}

(13) \texttt{@WAIT <general-purpose input or memory input number>,<input status>}

Waits until a specified general-purpose input or memory input is switched to a specified status.

- **Input number**: Specify one of the general-purpose inputs from 0 to 15 (16 points) or one of the memory inputs from 100 to 147 (48 points).

- **Input status**: Specify 1 (ON) or 0 (OFF).

**Transmission example**: \texttt{@WAIT l,l c/r l/f ..........................} Waits until DI1 becomes 1 (ON).

**Response example**: \texttt{OK c/r l/f}

(14) \texttt{@TIMR <time>}

Waits a specified amount of time.

- **Time**: Set the time between 1 and 65535 in units of 10ms.

**Transmission example**: \texttt{@TIMR 100 c/r l/f .........................} Waits one second.

**Response example**: \texttt{OK c/r l/f}
(15) \( @P \) <point number>
Sets the point variable P.

- **Point number**: This can be any value from 0 to 999.
- **Transmission example**: \( @P \ 100 \) c/r l/f ............................... Set the point variable P to 100.
- **Response example**: OK c/r l/f

⚠️ **CAUTION**
The contents of the point variable P are held even when the ERCX is turned off. However, when the program is reset or when the program reset is applied for example by switching the execution program, the point variable P will be initialized to 0.

(16) \( @P+ \)
Adds 1 to the point variable P.

- **Transmission example**: \( @P+ \) c/r l/f ............................... Increments the point variable P.
  \( (P \leftarrow P+1) \)
- **Response example**: OK c/r l/f

(17) \( @P- \)
Subtracts 1 from the point variable P.

- **Transmission example**: \( @P- \) c/r l/f ............................... Decrements the point variable P.
  \( (P \leftarrow P-1) \)
- **Response example**: OK c/r l/f
(18) @MOVM <pallet work position>,<speed>
Moves the robot to a specified pallet work position at a specified speed.
This command is available with controller version 13.23 or later.

Pallet work position : The pallet work position is a number used to identify each point on a matrix, and can be from 1 to 65025 (=255 \times 255). The counter array variable C or counter variable D can also be used.

Speed : The speed can be set to any level between 1 and 100. If PRM30 (Maximum program speed) is 100, then 100 will be equal to 3000 rpm (when PRM44=3000).

Transmission example : @MOVM 5,100 c/r l/f ............... When a 4 \times 3 matrix is defined, the robot moves to the point at "row 2, column 2" at 100% speed.

Response example 1 : OK c/r l/f
Response example 2 : NG c/r l/f ......................... Data error. The specified pallet work position is outside the matrix.

⚠️ CAUTION
- The MOVM statement performs calculation on the assumption that the robot operates on the Cartesian coordinate system.
- Because only a single-axis robot is controlled with the ERCX, the actual movement is linear even if a 2-dimensional matrix is defined.

(19) @MAT <number of rows>,<number of columns>,<pallet number>
Defines a matrix.
This command is available with controller version 13.23 or later.

Number of rows : Set the number of rows from 1 to 255.
Number of columns : Set the number of columns from 1 to 255.
Pallet number : The pallet number is a number used to identify each matrix (pallet) and can be from 0 to 31.

Transmission example : @MAT 5,2,1 c/r l/f ................. Defines a matrix of 5 \times 2 on pallet number 1.

Response example : OK c/r l/f

⚠️ CAUTION
Because only a single-axis robot is controlled with the ERCX, the actual movement is linear even if a 2-dimensional matrix is defined.

(20) @MSEL <pallet number>
Specifies a matrix where the robot moves with a MOVM statement.
This command is available with controller version 13.23 or later.

Pallet number : The pallet number is a number used to identify each matrix (pallet) and can be from 0 to 31.

Transmission example : @MSEL 0 c/r l/f ................. Specifies pallet number 0.

Response example : OK c/r l/f
(21) @CSEL <array element number>
Specifies an array element for the counter array variable C to be used.
This command is available with controller version 13.23 or later.

Array element number: This is a number used to designate an array element for the
counter array variable C, and can be from 0 to 31. The counter
variable D can also be specified here as the array element.

Transmission example: @CSEL 1 c/r l/f......................... Uses the counter array variable
C of element number 1 in the subsequent steps.

Response example: OK c/r l/f

(22) @C <counter value>
Sets a specified value in the counter array variable C specified with the CSEL statement.
This command is available with controller version 13.23 or later.

Counter value: This can be any value from 0 to 65535.

Transmission example: @C 100 c/r l/f......................... Sets the counter array variable
C to 100.

Response example: OK c/r l/f

(23) @C+ [<addition value>]
Adds a specified value to the counter array variable C.
This command is available with controller version 13.23 or later.

Addition value: This can be any value from 1 to 65535. If this value is omitted,
then 1 is added to the counter array variable.

Transmission example: @C+ c/r l/f............................. Increments the counter array variable C. (C ← C+1)

Response example: OK c/r l/f

(24) @C- [<subtraction value>]
Subtracts a specified value from the counter array variable C.
This command is available with controller version 13.23 or later.

Subtraction value: This can be any value from 1 to 65535. If this value is omitted,
then 1 is subtracted from the counter array variable.

Transmission example: @C- c/r l/f............................. Decrements the counter array variable C. (C ← C-1)

Response example: OK c/r l/f

(25) @D <counter value>
Sets a specified value in the counter variable D.
This command is available with controller version 13.23 or later.

Counter value: This can be any value from 0 to 65535.

Transmission example: @D 100 c/r l/f......................... Sets the counter variable D to
100.

Response example: OK c/r l/f
(26) @D+ [<addition value>]
Add a specified value to the counter variable D.
This command is available with controller version 13.23 or later.

Addition value : This can be any value from 1 to 65535. If this value is omitted, then 1 is added to the counter variable.
Transmission example : @D+ c/r l/f ......................................... Increments the counter variable D. (D ← D+1)
Response example : OK c/r l/f

(27) @D- [<subtraction value>]
Subtracts a specified value from the counter variable D.
This command is available with controller version 13.23 or later.

Subtraction value : This can be any value from 1 to 65535. If this value is omitted, then 1 is subtracted from the counter variable.
Transmission example : @D- c/r l/f ......................................... Decrements the counter variable D. (D ← D-1)
Response example : OK c/r l/f

(28) @SHFT <point number>
Shifts the position data by an amount equal to the distance defined by a specified point number.
The shifted data is valid until the SHFT statement is executed again or until the program is reset.
This command is available with controller version 13.23 or later.

Point number : This is a number used to identify each point (position data) and can be from 0 to 999 (a total of 1,000 points). Data for the point numbers can be edited with the @WRITE PNT statement. The point variable P can also be used.
Transmission example : @SHFT 1 c/r l/f ............................ Shifts the point data by an amount defined by point number 1 and the shifted data is used with the subsequent movement commands.
Response example : OK c/r l/f

⚠️ CAUTION
When the program is reset, the shift data will be initialized to 0.00.
The SHFT statement affects MOVA, MOVF and MOVM, but does not affect MOVD and MOVI.
11-5-2 Data handling

(1) @?POS
Reads the current position.

Transmission example : @?POS c/r l/f
Response example : 321.05 c/r l/f
OK c/r l/f

(2) @?NO
Reads the current program number. In multi-task operation, this command reads the program information on the task currently selected.

Transmission example : @?NO c/r l/f
Response example 1 : 31 c/r l/f ........................................ Program No.31 is being executed.
Response example 2 : 10/l c/r l/f ...................................... Program No.1 is the lead program (program selected with @SWI statement), and program No.10 is currently being executed with the JMP or CALL statement, etc.

(3) @?SNO
Reads the current step number. The @RUN and @SRUN commands are executed from the step read here. In multi-task operation, this command reads the program information on the task currently selected.

Transmission example : @?SNO c/r l/f
Response example : 170 c/r l/f
OK c/r l/f

(4) @?TNO
Reads the current task number.

Transmission example : @?TNO c/r l/f
Response example : 0 c/r l/f ........................................ Task 0 (main task) is currently selected.

(5) @?PNO
Reads the currently selected point number. This is used to find which point data is being used for movement, or to find the point that caused an error if it occurs. In multi-task operation, this command reads the program information on the task currently selected.

Transmission example : @? PNO c/r l/f
Response example : 57 c/r l/f
OK c/r l/f
(6) @?STP <program number>
Reads the total number of steps in the specified program.

Program number : This is a number used to identify each program and can be 0 to 99 (a total of 100).

Transmission example : @?STP 10 c/r l/f ..................... Reads the total number of steps for program No. 10.

Response example : 140 c/r l/f
OK c/r l/f

(7) @?MEM
Reads the number of steps that can be added.

Transmission example : @?MEM c/r l/f

Response example : 1001 c/r l/f
OK c/r l/f

⚠️ CAUTION
In addition to the number of existing steps, the steps equivalent to the number of programs are used internally as the program control steps. For example, if one program with 50 steps is registered, the number of the available remaining steps will be 2949 steps (3000 - 1 - 50 = 2949).

(8) @?VER
Reads the ROM version in the ERCX controller.

Transmission example : @?VER c/r l/f

Response example : 13.13 c/r l/f
OK c/r l/f

(9) @?ROBOT
Reads the type of the robot currently specified.

Transmission example : @?ROBOT c/r l/f

Response example : 90 c/r l/f
OK c/r l/f

(10) @?CLOCK
Reads the total operation time of the ERCX controller.

Transmission example : @?CLOCK c/r l/f

Response example : 00101,05:11:12 c/r l/f .................. Indicates that the total operation time is 101 days, 5 hours, 11 minutes and 12 seconds.
(11) @?ALM <history number>[,<display count>]
Displays a specified number of past alarms, starting from a specified history number.
A maximum of 100 past alarms can be displayed.
This alarm history shows the time (total elapsed time from controller start-up) that each alarm occurred and a description of the alarm.

History number
: This number is assigned to each alarm sequentially from 0 to 99 in the order the alarms occurred. History number 0 indicates the most recent alarm that occurred. A larger history number indicates it is an older alarm.

Display count
: Specify the number of alarms you want to display from 1 to 100. If this entry is omitted, only one alarm is displayed.

Transmission example
: @?ALM 0,2 c/r l/f ......................... Displays the two most recent alarms that occurred.

Response example
: 00101,05:11:12,X04: POWER DOWN c/r l/f
00096,18:10:02,X04: POWER DOWN c/r l/f
OK c/r l/f ............................... The most recent alarm that occurred was a voltage drop alarm occurring 101 days, 5 hours, 11 minutes and 12 seconds after the ERCX controller has started. The next most recent alarm was a voltage drop alarm occurring 96 days, 18 hours, 10 minutes and 2 seconds after the ERCX controller has started.

(12) @?EMG
Reads the emergency stop status.

Transmission example
: @?EMG c/r l/f

Response example 1
: 0 c/r l/f ................................. Emergency stop is off.
OK c/r l/f

Response example 2
: 1 c/r l/f ................................. Emergency stop is on.
OK c/r l/f

(13) @?SRVO
Reads the servo status.

Transmission example
: @?SRVO c/r l/f

Response example 1
: 0 c/r l/f ................................. Servo is off.
OK c/r l/f

Response example 2
: 1 c/r l/f ................................. Servo is on.
OK c/r l/f
(14) @?ORG
Reads whether or not return-to-origin has been completed.

Transmission example : @?ORG c/r l/f
Response example 1 : 0 c/r l/f ........................................ Return-to-origin not completed. OK c/r l/f
Response example 2 : 1 c/r l/f ........................................ Return-to-origin completed. OK c/r l/f

(15) @?MODE
Reads the robot status.

Transmission example : @?MODE c/r l/f
Response example 1 : 0 c/r l/f ........................................ Robot is stopped. OK c/r l/f
Response example 2 : 1 c/r l/f ........................................ Program is being executed from TPB or PC. OK c/r l/f
Response example 3 : 2 c/r l/f ........................................ Program is being executed by I/O command.

(16) @?PVA
Reads the point variable P. In multi-task operation, this command reads the program information on the task currently selected.

Transmission example : @?PVA c/r l/f
Response example : 0 c/r l/f OK c/r l/f

⚠️ CAUTION
The contents of the point variable P are held even when the ERCX is turned off. However, when the program is reset or when the program reset is applied for example by switching the execution program, the point variable P will be initialized to 0.

(17) @?DI <general-purpose input or memory input number>
Reads the status of a general-purpose input or memory input.

Input number : Specify one of the general-purpose inputs 0 to 15 (16 points) or one of the memory inputs 100 to 147 (48 points).

Transmission example : @?DI 1 c/r l/f
Response example 1 : 0 c/r l/f ........................................ Input status is off. OK c/r l/f
Response example 2 : 1 c/r l/f ........................................ Input status is on. OK c/r l/f
(18) \texttt{@DO <general-purpose output or memory output number>}  
Reads the status of a general-purpose output or memory output.

Output number : Specify one of the general-purpose outputs 0 to 12 (13 points) or one of the memory outputs 100 to 131 (32 points).

Transmission example : \texttt{@DO 2 c/r l/f}

Response example 1 : 0 c/r l/f .......................................... Output status is off.  
                     OK c/r l/f

Response example 2 : 1 c/r l/f .......................................... Output status is on.  
                     OK c/r l/f

(19-1) \texttt{@PRM <parameter number>}  
Reads the data from a specified parameter.

Parameter number : This is a number used to identify each parameter and can be from 0 to 63.

Transmission example : \texttt{@PRM1 c/r l/f}  
                     Reads the data from PRM1 (parameter 1).

Response example 1 : 350 c/r l/f  
                     OK c/r l/f

Response example 2 : c/r l/f .......................................... No data is registered in PRM1 (parameter 1).  
                     OK c/r l/f

(19-2) \texttt{@PRM <parameter number>,<parameter number>}  
Reads multiple parameter data from the first parameter number to the second parameter number. If unregistered parameters exist, they will be skipped.

Parameter number : This is a number used to identify each parameter and can be from 0 to 63.

Transmission example : \texttt{@PRM1,5 c/r l/f}  
                     Reads the data from PRM1 to PRM5 (parameters 1 to 5).

Response example : PRM1=350 c/r l/f  
                     PRM2=0 c/r l/f  
                     PRM3=30 c/r l/f  
                     PRM4=100 c/r l/f  
                     PRM5=0 c/r l/f  
                     OK c/r l/f
(20-1) @?P <point number>
Reads the data of a specified point.

Point number : This is a number used to identify each point data and can be from 0 to 999.

Transmission example : @?P 254 c/r l/f .............................. Reads the data of point 254.

Response example 1 : -0.05 c/r l/f
                      OK c/r l/f

Response example 2 : c/r l/f ........................................... No data is registered in the
                      OK c/r l/f specified point.

(20-2) @?P <point number>,<point number>
Reads multiple point data from the first point number to the second point number. If unregistered
points exist, they will be skipped.

Point number : This is a number used to identify each point data and can be from 0 to 999.

Transmission example : @?P15,22 c/r l/f ........................... Reads the data from points 15
to 22.

Response example : P15=100.00 c/r l/f
                    P16=32.11 c/r l/f
                    P20=220.00 c/r l/f
                    P22=0.50 c/r l/f
                    OK c/r l/f

(21-1) @READ <program number>,<step number>,<number of steps>
Reads a specified number of step data from a specified step in a program. If the number of steps
from the specified step to the final step is less than the number of steps specified here, the com-
mand execution will end when the final step is read out.

Program number : This is a number used to identify each program and can be from 0 to 99.

Step number : This is a number assigned to each step and can be from 1 to 255.

Number of steps : Any number between 1 and 255 can be specified.

Transmission example : @READ 3,50,1 c/r l/f .................. Reads one step of data from
                      step 50 in program No. 3.

Response example 1 : MOV A 29,100 c/r l/f
                      ^Z(=1AH)
                      OK c/r l/f

Response example 2 : NG c/r l/f ...................................... The specified step number is
                      42: cannot find step c/r l/f not registered
(21-2) @READ PGM
Reads all of the program data.

Transmission example : @READ PGM c/r l/f

Response example : NO0 c/r l/f
                   MOV A 0,100 c/r l/f
                   JMPF 0,31,13 c/r l/f
                   NO31 c/r l/f
                   STOP c/r l/f
                   ^Z (=1AH)
                   OK c/r l/f

(21-3) @READ PNT
Reads all point data.

Transmission example : @READ PNT c/r l/f

Response example : P0=0.00 c/r l/f
                    P1=350.00 c/r l/f
                    P2=196.47 c/r l/f
                    P254=-0.27 c/r l/f
                    ^Z (=1AH)
                    OK c/r l/f

(21-4) @READ PRM
Reads all parameter data.

Transmission example : @READ PRM c/r l/f

Response example : PRM0=20 c/r l/f
                   PRM1=350 c/r l/f
                   PRM2=2 c/r l/f
                   :                   :                   :
                   PRM40=2 c/r l/f
                   ^Z (=1AH)
                   OK c/r l/f
(21-5) @READ ALL
Reads all data (parameters, programs, points) at one time. Each data group (parameters, programs, points) is separated by an empty line (a carriage return only).

Transmission example: @READ ALL c/r l/f

Response example:
PRM0=20 c/r l/f
PRM1=350 c/r l/f
...  
PRM40=2 c/r l/f  
c/r l/f
NO0 c/r l/f
MOVA 0,100 c/r l/f
MOVA 1,100 c/r l/f
NO10 c/r l/f
CALL 0,10 c/r l/f
STOP c/r l/f
C/r l/f
P0=0.00 c/r l/f
P1=550.00 c/r l/f
^Z (=1AH)
OK c/r l/f

(21-6) @READ DIO
Reads the on/off status of DIO. Refer to "4-3-4 DIO monitor screen".

Transmission example: @READ DIO c/r l/f

Response example:
DI 00000000 00000000 c/r l/f
10000000 c/r l/f
DO 00000000 11000000 c/r l/f
XO:0 XS:1 c/r l/f
OK c/r l/f

(21-7) @READ MIO
Reads the on/off status of memory I/O. From the left, the top line shows MIO numbers from 115 to 100, the middle line from 131 to 116, and the bottom line from 147 to 132.

Transmission example: @READ MIO c/r l/f

Response example:
M 00000000 00000000 c/r l/f
00000000 00000000 c/r l/f
00000000 00000001 c/r l/f
OK c/r l/f
(21-8) @READ INF
Reads the status of the registered programs. The registered program numbers and number of steps are displayed.

Transmission example : @READ INF c/r l/f

Response example : NO0- 43 steps c/r l/f
                  NO1- 52 steps c/r l/f
                  NO31- 21 steps c/r l/f
                  ^Z (=1AH)
                  OK c/r l/f

(22-1) @WRITE PGM
Writes the program data. The controller will transmit READY when this command is received. Confirm that READY is received and then transmit the program data. Always transmit ^Z (=1AH) at the end of the data.

Transmission example : Send
@WRITE PGM c/r l/f
Receive
READY c/r l/f
NO0 c/r l/f
MOVA 0,100 c/r l/f
JMPF 0,31,12 c/r l/f
NO31 c/r l/f
STOP c/r l/f
^Z(=1AH)

OK c/r l/f

⚠ CAUTION
When @WRITE PGM is executed, the previous data of the same program number is overwritten. (The previous data remains as long as its program number differs from the program number to be written.)

(22-2) @WRITE PNT
Writes the point data. The controller will transmit READY when this command is received. Confirm that READY is received and then transmit the point data. Always transmit ^Z (=1AH) at the end of the data.

Transmission example : Send
@WRITE PNT c/r l/f
Receive
READY c/r l/f
P0=0.00 c/r l/f
P1=350.00 c/r l/f
P254=-0.27 c/r l/f
^Z(=1AH)

OK c/r l/f

⚠ CAUTION
When @WRITE PNT is executed, the previous data of the same point number is overwritten. (The previous data remains as long as its point number differs from the point number to be written.)
(22-3) @WRITE PRM
W riten the parameter data. The controller will transmit READY when this command is received. Confirm that READY is received and then transmit the parameter data. Always transmit ^Z (=1AH) at the end of the data.

Transmission example:
Send: @WRITE PRM c/r l/f

Receive:
READY c/r l/f

PRM1=550 c/r l/f
PRM2=10 c/r l/f
^Z(=1AH)

CAUTION
Loading unsuitable robot data to the ERCX can inhibit the robot controller performance, possibly resulting in failures, malfunctions, and errors.

(22-4) @WRITE ALL
Writes all data (parameters, programs and points) at one time. The controller will transmit READY when this command is received. Confirm that READY is received and then transmit all data. Always transmit ^Z (=1AH) at the end of the data.

Transmission example:
Send: @WRITE ALL c/r l/f

Receive:
READY c/r l/f

PRM0=20 c/r l/f
PRM1=350 c/r l/f
c/r l/f
NO10 c/r l/f
CALL 0, 20 c/r l/f
STOP c/r l/f
c/r l/f
P1=550.00 c/r l/f
^Z(=1AH)

CAUTION
- Always place one or more empty line to separate between each data group (parameters, programs, points).
- There is no specific rule in the data group sequence. There can be data groups that are not written in.
- When @WRITE ALL is executed, the previous data of the same program number or point number is overwritten. (The previous data remains as long as its program number or point number differs from the program number or point number to be written.)
- Loading unsuitable robot data to the ERCX can inhibit the robot controller performance, possibly resulting in failures, malfunctions, and errors.
(23) @?ERR <history number>[, <display count>]
Displays a specified number of past errors, starting from a specified history number.
A maximum of 100 past errors can be displayed.
This error history shows the time (total elapsed time from controller start-up) that each error occurred and a description of the error.
This command is available with controller version 13.50 or later.

| History number | This number is assigned to each error sequentially from 0 to 99 in the order the errors occurred. History number 0 indicates the most recent error that occurred. A larger history number indicates it is an older error. |
| Display count | Specify the number of errors you want to display from 1 to 100. If this entry is omitted, only one error is displayed. |

Transmission example: @?ERR 0,2 c/r l/f ......................... Displays the two most recent errors that occurred.

Response example: 00:00101,05:11:12,PIO,52 : NO POINT DATA c/r l/f
01:00096,18:10:02,CMU,30: SOFT LIMIT OVER c/r l/f
OK c/r l/f ...................................... The most recent error that occurred was a "no point data" error in a parallel I/O command occurring 101 days, 5 hours, 11 minutes and 12 seconds after the ERCX controller has started. The next most recent error was a "soft limit over" error during TPB or RS-232C operation occurring 96 days, 18 hours, 10 minutes and 2 seconds after the ERCX controller has started.

(24) @?MAT <pallet number>
Reads the matrix data on a specified pallet.
This command is available with controller version 13.23 or later.

| Pallet number | This is a number used to identify each matrix (pallet) and can be from 0 to 31. |

Transmission example: @?MAT 1 c/r l/f ........................... Reads the matrix data on pallet number 1.

Response example: 20,30 c/r l/f
OK c/r l/f

(25) @?MSEL
Reads the pallet number for the currently specified matrix. In multi-task operation, this command reads the program information on the task currently selected.
This command is available with controller version 13.23 or later.

Transmission example: @?MSEL c/r l/f

Response example: 0 c/r l/f
OK c/r l/f
(26) @?CSEL
Reads the currently specified element number of the counter array variable C. In multi-task operation, this command reads the program information on the task currently selected. This command is available with controller version 13.23 or later.

Transmission example : @?CSEL c/r l/f
Response example : 0 c/r l/f OK c/r l/f

(27) @?C [<array element number>]
Reads the value in the counter array variable C of the specified element number. This command is available with controller version 13.23 or later.

Element number : This is a number used to specify each array element and can be from 0 to 31. If this entry is omitted, the element number selected with the @CSEL command is used.

Transmission example : @?C c/r l/f
                     OK c/r l/f
Response example : 21202 c/r l/f
                     OK c/r l/f

(28) @?D
Reads the counter variable D. This command is available with controller version 13.23 or later.

Transmission example : @?D c/r l/f
Response example : 21202 c/r l/f OK c/r l/f

(29) @?SHFT
Reads the shift data currently set. In multi-task operation, this command reads the program information on the task currently selected. This command is available with controller version 13.23 or later.

Transmission example : @?SHFT c/r l/f
Response example : 150.00 c/r l/f OK c/r l/f
11-5-3 Utilities

(1-1) @INIT PGM
Initializes all program data.

Transmission example : @INIT PGM c/r l/f
Response example : OK c/r l/f

(1-2) @INIT PNT
Initializes all point data.

Transmission example : @INIT PNT c/r l/f
Response example : OK c/r l/f

(1-3) @INIT PRM <robot number>
Initializes the parameter data to match the specified robot.
For robot numbers, refer to "15-1-2 Robot number list".

Transmission example : @INIT PRM 90 c/r l/f .................. Parameter data is initialized to match the model T4 robot.
Response example : OK c/r l/f

⚠️ CAUTION
After initialization, change the lead length parameter (PRM12) to match the robot lead length.

(1-4) @INIT CLOCK
Initializes the timer to 0, which is used to measure the total operation time of the ERCX controller.
The alarm history and error history are also initialized at this point.

Transmission example : @INIT CLOCK c/r l/f
Response example : OK c/r l/f

(1-5) @INIT ALM
Initializes the alarm history.

Transmission example : @INIT ALM c/r l/f
Response example : OK c/r l/f

(1-6) @INIT ERR
Initializes the error history.
This command is available with controller version 13.50 or later.

Transmission example : @INIT ERR c/r l/f
Response example : OK c/r l/f
(2) @SWI  <program number>
This command switches the execution program number. When a program is reset, program execution will always return to the first step of the program selected here. The program is reset when the @SWI command is executed.

Program number : This is a number used to identify each program and can be from 0 to 99.

Transmission example : @SWI 31 c/r l/f

Response example : OK c/r l/f

(3) @SWITSK  <task number>
Switches the task number to be executed. In the subsequent step run, the program of the task selected here is executed. When the command such as @?NO or @?SNO is issued, the contents of this task replies to it.

Task number : This is a number used to identify each task and can be from 0 to 3.

Transmission example : @SWITSK 1 c/r l/f

Response example 1 : OK c/r l/f

Response example 2 : NG c/r l/f ................................. The specified task was not found. 72: not execute task c/r l/f

(4) @SINS  <program number>,<step number>
Inserts data in a specified step of a specified program. All data below the inserted data will shift down one line. If the step following the last step is specified, a new step will be added. If the first step of a program that does not exist is specified, a new program will be created. The ERCX controller will transmit READY when this command is received. Confirm that READY is received and then transmit the insertion data.

Program number : This is a number used to identify each program and can be from 0 to 99.

Step number : This is a number used to identify each step and can be from 1 to 255.

Transmission example 1 : Send
@SINS 19,4 c/r l/f
TIMR 50 c/r l/f

Receive
READY c/r l/f
OK c/r l/f

Transmission example 2 : Send
@SINS 19,4 c/r l/f

Receive
NG c/r l/f
43: cannot find PGM c/r l/f
(5) @SDEL <program number>,<step number>
Deletes a specified step.

**Program number**: This is a number used to identify each program and can be from 0 to 99.

**Step number**: This is a number used to identify each step and can be from 1 to 255.

**Transmission example**: @SDEL 31,99 c/r l/f ................... Deletes step 99 of program No. 31.

**Response example 1**: OK c/r l/f

**Response example 2**: NG c/r l/f ...................................... The specified step number is not registered.

(6) @SMOD <program number>,<step number>
Modifies data in a specified step. The ERCX controller will transmit READY when this command is received. Confirm that READY is received and then transmit the modification data.

**Program number**: This is a number used to identify each program and can be from 0 to 99.

**Step number**: This is a number used to identify each step and can be from 1 to 255.

**Transmission example 1**: Send
@SMOD 0,5 c/r l/f
TIMR 50 c/r l/f
OK c/r l/f

**Transmission example 2**: Send
@SMOD 0,5 c/r l/f
NG c/r l/f
43: cannot find PGM c/r l/f

(7) @COPY <program number (copy source)>,<program number (copy destination)>
Copies a program. If a program exists in the copy destination, the program will be rewritten.

**Program number**: This is a number used to identify each program and can be from 0 to 99.

**Transmission example**: @COPY 0,1 c/r l/f ....................... Copies program No. 0 to program No. 1.

**Response example 1**: OK c/r l/f

**Response example 2**: NG c/r l/f ...................................... The program to be copied is not registered.
(8) @DEL <program number>
Deletes a program.

Program number : This is a number used to identify each program and can be from 0 to 99.

Transmission example : @DEL 10 c/r l/f ......................... Deletes program No. 10.

Response example 1 : OK c/r l/f

Response example 2 : NG c/r l/f ................................. The program to be deleted is not registered.

(9) @PDEL <point number>,<number of points>
Deletes point data. Deletes the specified number of points starting with the point number specified here.

Point number : This is a number assigned to each point and can be from 0 to 999.

Number of points : Any number between 1 and 999 can be specified.

Transmission example : @PDEL 16,10 c/r l/f .................. Deletes 10 points starting from point 16 (up to point 25).

Response example : OK c/r l/f
This section lists all of the messages that are displayed on the TPB or sent to the PC (personal computer) to inform the operator of an error in operation or a current status. For a list of the alarm messages displayed if any trouble occurs, refer to "13-2 Alarm and Countermeasures".
12-1 Error Messages

12-1-1 Error message specifications

The error message transmission format is as follows.

<Error No.> : <Error message> c/r l/f

The length of the <error message> character string is 17 characters. (Spaces are added until the message contains 17 characters.) Thus, the character string length containing the c/r l/f will be 22 characters.

12-1-2 Command error message

<table>
<thead>
<tr>
<th>Error No.</th>
<th>Message</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>no start code</td>
<td>The start code (@) has not been added at the beginning of the command.</td>
<td>Always make sure the command begins with a start code (@).</td>
</tr>
<tr>
<td>21</td>
<td>illegal type</td>
<td>The command is erroneous.</td>
<td>Use the correct command.</td>
</tr>
<tr>
<td>22</td>
<td>line buf overflow</td>
<td>The number of characters in one line exceeds 80.</td>
<td>Limit the number of characters per line to 80 or less.</td>
</tr>
<tr>
<td>23</td>
<td>data error</td>
<td>There is an error in numeric data.</td>
<td>Correct the data.</td>
</tr>
<tr>
<td>24</td>
<td>cannot access</td>
<td>Execution is limited by the password or access level (operation level).</td>
<td>Cancel the limit.</td>
</tr>
</tbody>
</table>
### 12-1-3 Operation error message

<table>
<thead>
<tr>
<th>Error No.</th>
<th>Message</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>soft limit over</td>
<td>Executing the command will move the robot to a position that exceeds the soft limit set by parameter.</td>
<td>Review the point data or soft limit parameter.</td>
</tr>
<tr>
<td>31</td>
<td>running</td>
<td>Another command is already being executed, so the command cannot be accepted.</td>
<td>Wait until execution of the current command finishes before inputting another command.</td>
</tr>
<tr>
<td>32</td>
<td>origin incomplete</td>
<td>The command cannot be executed because a return to origin has not yet been completed.</td>
<td>Complete a return to origin first. Charge the absolute battery if not charged.</td>
</tr>
<tr>
<td>33</td>
<td>emergency stop</td>
<td>The command cannot be executed because an emergency stop is triggered.</td>
<td>Cancel the emergency stop.</td>
</tr>
<tr>
<td>34</td>
<td>servo off</td>
<td>The command cannot be executed because the servo is off.</td>
<td>Turn servo on.</td>
</tr>
<tr>
<td>35</td>
<td>system error 2</td>
<td>An error interruption occurred due to noise or an unknown cause, so the status changed to servo off.</td>
<td>Turn servo on.</td>
</tr>
<tr>
<td>36</td>
<td>cannot restart</td>
<td>Restart of the interpolation operation program was attempted.</td>
<td>Reset the program.</td>
</tr>
<tr>
<td>37</td>
<td>SVCE-port changed</td>
<td>Execution was forcibly terminated because the SERVICE mode input state was changed.</td>
<td>Restart execution.</td>
</tr>
<tr>
<td>38</td>
<td>net link error</td>
<td>The connection was forcibly disconnected because an error occurred in the network connection.</td>
<td>Remedy the network connection error, and then restart.</td>
</tr>
</tbody>
</table>
### 12-1-4 Program error message

<table>
<thead>
<tr>
<th>Error No.</th>
<th>Message</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
</table>
| 40        | stack overflow                 | ① Seven or more successive CALL statements were used within a CALL statement.  
                        |                                                                 | ② In the program called as a subroutine by a CALL statement, a jump was made to another program by a JMP or JMPF statement. |
|           |                                | ① Reduce the number of CALL statements used in a CALL statement to 6 or less.  
                        | ② Review the program.                                                                                       |                                                                       |
| 41        | cannot find label              | The specified label cannot be found.                                  | Create the required label.                                             |
| 42        | cannot find step               | The specified step cannot be found.                                   | Check whether the step number is correct.                              |
| 43        | cannot find PGM                | The specified program cannot be found.                                | Check whether the program number is correct.                           |
| 44        | PGM memory full                | The total number of steps in all programs has exceeded 3000.          | Delete unnecessary programs or steps.                                  |
| 45        | step over                      | The total number of steps in one program has exceeded 255.             | Delete unnecessary steps or divide the program into two parts.          |
### 12-1-5 System error message

<table>
<thead>
<tr>
<th>Error No.</th>
<th>Message</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>system error</td>
<td>An unexpected error occurred.</td>
<td>Contact YAMAHA and describe the problem.</td>
</tr>
<tr>
<td>51</td>
<td>illegal opcode</td>
<td>There is an error in a registered program.</td>
<td>Check the program.</td>
</tr>
<tr>
<td>52</td>
<td>no point data</td>
<td>No data has been registered for the specified point number.</td>
<td>Register the point data.</td>
</tr>
<tr>
<td>53</td>
<td>PRM0 data error</td>
<td>This error will not occur in the ERCX controller.</td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>PRM8 data error</td>
<td>The number of conditional input points is set to something other than 1 to 8.</td>
<td>Correct the setting for the PRM8 parameter.</td>
</tr>
<tr>
<td>59</td>
<td>robot type error</td>
<td>Unsuitable parameter data was transmitted to the controller.</td>
<td>Initialize the parameters. Transmit the correct parameter data.</td>
</tr>
</tbody>
</table>

### 12-1-6 Multi-task error message

<table>
<thead>
<tr>
<th>Error No.</th>
<th>Message</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>70</td>
<td>task running</td>
<td>An attempt was made to start the task which is already in progress.</td>
<td>Check the program.</td>
</tr>
<tr>
<td>71</td>
<td>can't select task</td>
<td>An attempt was made by a task to finish itself. An attempt was made to switch a task which is suspended.</td>
<td>Check the program. Check the task status.</td>
</tr>
<tr>
<td>72</td>
<td>not execute task</td>
<td>An attempt was made to switch a task which has not started.</td>
<td>Check the task status.</td>
</tr>
</tbody>
</table>
### 12-2 TPB Error Messages

<table>
<thead>
<tr>
<th>Message</th>
<th>Cause</th>
<th>Action</th>
</tr>
</thead>
</table>
| SIO error | 1. Parity error in data received from controller.  
             2. TPB was connected when dedicated command input was on. | 1. Contact YAMAHA for consultation.  
             2. Turn all dedicated command inputs off before connecting the TPB. |
| bad format| The memory card is not formatted.                                    | Format the memory card.                                                 |
| save error| Error in writing to the memory card.                                 | Replace the memory card.                                                |
| load error| The memory card data is damaged.                                     | Format or replace the memory card.                                     |
| checksum error| The memory card data is damaged.                                   | Format or replace the memory card.                                     |
| battery error| The memory card battery voltage dropped.                         | Replace the memory card battery.                                       |
| printer busy!!| The printer is not ready.                                   | Set the printer to print-ready state.                                  |
12-3  Stop Messages

12-3-1 Message specifications

The stop message transmission format is as follows.

<Stop No.> : <Stop message> c/r l/f

The length of the <stop message> character string is 17 characters. (Spaces are added until the message contains 17 characters.) Thus, the character string length containing the c/r l/f will be 22 characters.

12-3-2 Stop messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Message</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>60</td>
<td>program end</td>
<td>Execution has stopped because the program has ended.</td>
</tr>
<tr>
<td>61</td>
<td>stop key</td>
<td>Execution has stopped because the Stop key on the TPB was pressed.</td>
</tr>
<tr>
<td>62</td>
<td>interlock</td>
<td>Execution has stopped because an I/O interlock was applied.</td>
</tr>
<tr>
<td>63</td>
<td>stop command</td>
<td>Execution has stopped because the STOP command was carried out.</td>
</tr>
<tr>
<td>64</td>
<td>key release</td>
<td>Execution has stopped by the hold-to-run function.</td>
</tr>
</tbody>
</table>
12-4 Displaying the Error History

A history of past errors can be displayed. Up to 100 errors can be stored in the controller. This function is available when the controller version is 13.50 or later and the TPB version is 12.18 or later.

1) On the initial screen, press [F3] (SYS).

2) Next, press [F4] (next) to change the menu display and then press [F3] (UTL).


5) History numbers, time that errors occurred (total elapsed time from controller start-up) and error descriptions are displayed. One screen displays the past 4 errors in the order from the most recent error. Pressing the \(-\) and \(+)\) keys displays the hidden items. Press the \[STEP\] and \[STEP\] keys to sequentially scroll through the error list.

<table>
<thead>
<tr>
<th>History number</th>
<th>Time the error occurred</th>
<th>Movement command control mode immediately before the error occurred</th>
<th>Error description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00 : 00101, 05:11:12, CMU</td>
<td>00:00101, 05:11:12, CM</td>
<td>CMU: TPB or RS-232C control</td>
<td>62: Interlock</td>
</tr>
<tr>
<td>01:00096, 18:10:02, PI</td>
<td>01:00096, 18:10:02, PI</td>
<td>PIO: Parallel I/O control</td>
<td></td>
</tr>
<tr>
<td>02:00080, 10:07:33, CM</td>
<td>02:00080, 10:07:33, CM</td>
<td>SIO: Serial I/O control</td>
<td></td>
</tr>
<tr>
<td>03:00015, 20:35:45, CM</td>
<td>03:00015, 20:35:45, CM</td>
<td>WIO: Remote command (CC-Link) control</td>
<td></td>
</tr>
</tbody>
</table>

6) To return to the previous screen, press \[ESC\].
This chapter explains how to take corrective action when a problem or breakdown occurs, by categorizing it into one of two cases depending on whether or not an alarm is output from the controller.
### 13-1 If A Trouble Occurs

If trouble or breakdown occurs, contact YAMAHA or your YAMAHA dealer, providing us with the following information in as much detail as possible.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description (example)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>What you were using</strong></td>
<td>- Controller model name : ERCX-B1&lt;br&gt;- Robot model name : T4-12-300&lt;br&gt;- Controller version : Ver. 13.13&lt;br&gt;- Power supply for controller : 24V/3A&lt;br&gt;- 24V power supply for I/O : Shared with power supply for ERCX controller, or used separately from it.</td>
</tr>
<tr>
<td><strong>When</strong></td>
<td>- When purchased&lt;br&gt;- How long used, how often used&lt;br&gt;- Problem happened at power on? One hour after power was turned on.</td>
</tr>
<tr>
<td><strong>Under what conditions</strong></td>
<td>- During automatic operation&lt;br&gt;- While writing a program&lt;br&gt;- When the robot was at a specific position</td>
</tr>
<tr>
<td><strong>What happened</strong></td>
<td>- Servo does not lock.&lt;br&gt;- Alarm (No. and message) is issued.&lt;br&gt;- Motor makes an unusual sound.&lt;br&gt;- A program was lost.</td>
</tr>
<tr>
<td><strong>How often</strong></td>
<td>- Always occurs.&lt;br&gt;- Occurs once an hour.&lt;br&gt;- Cannot be made to occur again.</td>
</tr>
</tbody>
</table>
13-2 Alarm and Countermeasures

If the READY signal is turned off except in cases of emergency stop, then an alarm has probably been issued. The status LED on the front panel of the controller lights up in red.

13-2-1 Alarm specifications

■ If an alarm is issued:
If an alarm is issued, keep the power turned on and connect the TPB or set the POPCOM on-line to check the contents of the alarm. An alarm message appears on the screen.

The transmission format for alarm messages is as follows.

<alarm number> : <alarm message> c/r l/f

The <alarm number> is displayed in two digits, so a one-digit number is prefixed with 0 like 01. The <alarm message> is displayed in a 17 character string length. (Spaces are added until the message contains 17 characters.) Therefore, an message including c/r and l/f consists of 22 characters.

■ To cancel the alarm:
To cancel the alarm, turn the power off and after first eliminating the problem, turn it back on again.

If an alarm is still issued while the power is turned on, then try turning the power on while the robot is in emergency stop. No alarm detection is performed with this method, so that the data can be checked, corrected or initialized. Normal alarm detection is performed when the servo is turned on after cancelling emergency stop.
### 13-2-2 Alarm message list

<table>
<thead>
<tr>
<th>Alarm No.</th>
<th>Alarm Message</th>
<th>Meaning</th>
<th>Possible Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>OVER LOAD</td>
<td>Excessive load on motor</td>
<td>① Improper operation</td>
<td>① Lower the operation duty on the robot or reduce the acceleration parameter, or correct the payload parameter. ② If the motor armature resistance is too low or the motor movement is sluggish when turned by hand, then replace the motor assembly. ③ Initialize the parameters and check the robot type setting. ④ Supply 24V to the brake wire to check whether the brake is released. ⑤ Check the power supply capacity. If too low, use a power supply of larger capacity. ⑥ Check whether the robot moving parts work sluggishly. If sluggish, then adjust the mechanical alignment.</td>
</tr>
<tr>
<td>02</td>
<td>OVER CURRENT</td>
<td>Motor drawing too much current</td>
<td>① Motor wire shorted</td>
<td>① Check the motor wires for electrical continuity, and replace the motor assembly if abnormality is found. ② Replace the motor if internally shorted. ③ If the resistance between motor terminals U and V, U and W, or V and W is lower than 100 ohms, the output transistor is defective, so replace the ERCX controller. ④ Correct the ambient environmental conditions. (Install a cooling fan.) ⑤ Lower the operation duty on the robot. ⑥ If the controller is being used correctly, the transistor is probably defective, so replace the ERCX controller.</td>
</tr>
<tr>
<td>03</td>
<td>OVER HEAT</td>
<td>Transistor has heated to 90°C or more.</td>
<td>① Rise in ambient temperature (above 40°C)</td>
<td>① Correct the ambient environmental conditions. (Install a cooling fan.) ② Lower the operation duty on the robot. ③ If the controller is being used correctly, the transistor is probably defective, so replace the ERCX controller.</td>
</tr>
<tr>
<td>04</td>
<td>POWER DOWN</td>
<td>Power supply voltage has dropped to less than 70% of rated value.</td>
<td>① Insufficient power supply capacity</td>
<td>① Check the power supply capacity. If insufficient, use a power supply having larger capacity. (Power is consumed mostly during return-to-origin of stroke end detection, robot start-up and acceleration/deceleration.)</td>
</tr>
<tr>
<td>05</td>
<td>BATT.LOW-VOLTAGE</td>
<td>System backup battery voltage is low</td>
<td>① Battery worn out.</td>
<td>① Replace the battery. (If not possible to replace it immediately, then temporarily set bit 3 of PRM34 to “1”.)</td>
</tr>
<tr>
<td>06</td>
<td>24V POWER OFF</td>
<td>24V power is not supplied.</td>
<td>① 24V power supply is not connected to A13 and B13 of the I/O connector. ② Fuse has blown due to short-circuit or excessive current flow in the 24V circuit.</td>
<td>① Check the 24V power supply. ② Check for short-circuit using a multimeter or recheck the I/O connections.</td>
</tr>
<tr>
<td>Alarm No.</td>
<td>Alarm Message</td>
<td>Meaning</td>
<td>Possible Cause</td>
<td>Action</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------</td>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| 07       | P.E. COUNTER OVER   | Overflow in position deviation counter       | ① Mechanical lock  
② Motor wire is broken or connected wrong.  
③ Electromagnetic brake failure or wire broken  
④ Parameter error | ① Check whether robot moving parts are locked.  
② Check the motor wire and resolver signal wire connections.  
③ Supply 24V to the brake wire to check whether the brake is released.  
④ Initialize the parameters. |
| 08       | PNT DATA DESTROY    | Point data has been damaged.                 | ① Backup circuit failure  
② Power was turned off while writing data.  
③ Data was destroyed by external noise. | ①② In emergency stop, turn power on and check point data. If part of the data is defective, correct the data. If all data are defective, initialize the point data and then reload the data. If there is no problem with the data, perform rewriting on any data.  
③ Check the surrounding environment for noise. |
| 09       | PRM DATA DESTROY    | Parameter data has been corrupted.           | ① Backup circuit failure  
② Power was turned off while writing data.  
③ Data was destroyed by external noise. | ①② In emergency stop, turn power on and initialize the parameters.  
③ Check the surrounding environment for noise. |
| 10       | PGM DATA DESTROY    | Program data has been corrupted.             | ① Backup circuit failure  
② Power was turned off while writing data.  
③ Data was destroyed by external noise. | ①② In emergency stop, turn power on and check program data. If part of the program is defective, correct the data. If all data are defective, initialize the program data and then reload the data. If there is no problem with the data, perform rewriting on any data.  
③ Check the surrounding environment for noise. |
| 11       | SYSTEM FAULT        | Software problem                             | ① External noise or other factors have disrupted software program  
② Overflow in receiving buffer. When communicating with a PC, the XON/XOFF control communication parameter was not selected on the PC. | ① Check the surrounding environment for noise.  
② Select the XON/XOFF control. |
| 12       | Not used            |                                              |                                                                               |                                                                       |
| 13       | Not used            |                                              |                                                                               |                                                                       |
| 14       | FEEDBACK ERROR 1    | Incorrect parameter setting                  | ① Parameter error  
② Motor is miswired.  
③ Resolver signal wire is disconnected. | ① Initialize the parameters.  
② Check the motor wire connection.  
③ Check the resolver signal wire connection. |
<p>| 15       | FEEDBACK ERROR 2    | Resolver signal discontinuity                | ① Resolver signal wire is broken. | ① Check the resolver signal wire connection. |</p>
<table>
<thead>
<tr>
<th>Alarm No.</th>
<th>Alarm Message</th>
<th>Meaning</th>
<th>Possible Cause</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>ABNORMAL VOLTAGE</td>
<td>Excessive voltage (higher than 30V) generated</td>
<td>1. Rise in regenerative absorption resistor temperature (above 120°C)&lt;br&gt;2. Incorrect power supply voltage</td>
<td>1. Lower the operation duty on the robot, or install a cooling fan.&lt;br&gt;2. Check the power supply voltage.</td>
</tr>
<tr>
<td>17</td>
<td>SYSTEM FAULT 2</td>
<td>Controller’s internal LSI error</td>
<td>1. Internal LSI failure or malfunction</td>
<td>1. If the error occurs frequently, then the LSI is probably defective, so replace the ERCX controller.</td>
</tr>
<tr>
<td>18</td>
<td>FEEDBACK ERROR 3</td>
<td>Motor cable is disconnected, improperly wired or overload</td>
<td>1. Motor wire is broken or disconnected.&lt;br&gt;2. The robot slider struck on an obstacle or mechanical damper.&lt;br&gt;3. Defective or disconnected electromagnetic brake&lt;br&gt;4. Parameter error&lt;br&gt;5. Insufficient power supply capacity&lt;br&gt;6. Drop in voltage at stopper origin</td>
<td>1. Check the motor wire connection.&lt;br&gt;2. Remove the obstacle or correct the point data or origin position.&lt;br&gt;3. Apply 24V to the brake line and check brake release.&lt;br&gt;4. Initialize the parameters.&lt;br&gt;5.6. Check the power supply capacity and increase if necessary.</td>
</tr>
<tr>
<td>19</td>
<td>SYSTEM FAULT 3</td>
<td>CPU error</td>
<td>1. External noise or other factors have disrupted software program.&lt;br&gt;2. CPU failure or malfunction</td>
<td>1. Check the environment for noise.&lt;br&gt;2. If the error occurs frequently, then the CPU is defective. Replace the ERCX controller.</td>
</tr>
<tr>
<td>20</td>
<td>Not used</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Not used</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>VERSION MISMATCH</td>
<td>Wrong combination of PB and controller</td>
<td>1. The PB used does not match the controller.</td>
<td>1. Replace the PB.</td>
</tr>
<tr>
<td>23</td>
<td>ABS BAT L-VOLTAGE</td>
<td>The absolute battery voltage is low.</td>
<td>1. The absolute battery voltage is low.&lt;br&gt;2. The absolute battery is not connected.&lt;br&gt;3. End of the absolute battery service life.&lt;br&gt;4. The absolute battery is defective.</td>
<td>1. Charge the absolute battery.&lt;br&gt;2. Connect the absolute battery. (If using the controller with the absolute battery disconnected, set bit 4 of PRM 34 to “0”.)&lt;br&gt;3. When the backup time becomes short even after fully charging the battery, this is probably the end of service life. Replace the battery.&lt;br&gt;4. When this alarm does not disappear even after fully charging the battery, the battery is probably defective. Replace the battery.</td>
</tr>
<tr>
<td>24</td>
<td>ABS DATA ERROR</td>
<td>Absolute data error was detected.</td>
<td>1. Movement amount has exceeded the limit (approx. ±4000 turns) that can be retained during power off.&lt;br&gt;2. The absolute battery was disconnected during power off.&lt;br&gt;3. Discontinuity of absolute battery wire.&lt;br&gt;4. Discontinuity or misconnection of resolver signal line&lt;br&gt;5. Circuit defect or malfunction</td>
<td>1. Limit the movement range during power off.&lt;br&gt;2. Do not disconnect the absolute battery when the position data is backed up.&lt;br&gt;3. Check the absolute line connection.&lt;br&gt;4. Check the resolver signal line connection.&lt;br&gt;5. If the fault occurs frequently, replace the controller.</td>
</tr>
<tr>
<td>25</td>
<td>ABS DATA ERROR 2</td>
<td>Abnormal reset</td>
<td>1. Malfunction due external noise, etc.&lt;br&gt;2. Circuit defect or malfunction</td>
<td>1. Check the environment for noise.&lt;br&gt;2. If the fault occurs frequently, replace the controller.</td>
</tr>
<tr>
<td>26</td>
<td>FEEDBACK ERROR 4</td>
<td>Motor wire breakage or misconnection</td>
<td>1. Motor wire is broken or connected wrong.&lt;br&gt;2. Parameter error&lt;br&gt;3. Wrong power supply voltage setting&lt;br&gt;4. Insufficient power supply capacity</td>
<td>1. Check the motor wire connection.&lt;br&gt;2. Initialize the parameters.&lt;br&gt;3. Check the voltage setting (100V/200V).&lt;br&gt;4. Check the power supply capacity. If too low, use a power supply of larger capacity.</td>
</tr>
</tbody>
</table>
13-3 Troubleshooting for Specific Symptom

If any problems develop while the controller is being used, check the items below for the appropriate way to handle them. If the problem cannot be corrected using the steps listed below, please contact our sales office or sales representative right away.

13-3-1 Relating to the robot movement

<table>
<thead>
<tr>
<th>No.</th>
<th>Symptom</th>
<th>Possible Cause</th>
<th>Items to Check</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Servo of robot does not lock even after power is turned on.</td>
<td>1) Power is not being supplied. 2) Emergency stop is activated. 3) The servo is off. 4) An alarm has occurred.</td>
<td>• Check that the status LED on the front panel of the controller lights up or flashes. • If the READY signal of the I/O connector is off and no alarm has been issued, an emergency stop is in effect. • Check whether the status LED is flashing. • Check whether the servo has been turned off in the program, and whether the TPB has been plugged or unplugged. • Check whether the status LED is flashing. • Connect the TPB and check whether an alarm is displayed. • Check whether the status LED is lit in red. • Check whether bit 4 of PRM34 is set to “1”.</td>
<td>• Check the voltage at the power cable plug. If the voltage is correct, replace the ERCX controller. • Turn the servo on with the I/O servo recovery input or from the TPB operation. • Take corrective action according to the instructions in the alarm message list. • Correct the program if necessary. • Change the program to select the desired program. • Change the program to select the desired program. • If an origin-related parameter has been changed, perform return-to-origin again. • Remove the cause of alarm and then perform return-to-origin. • Set bit 4 of PRM34 to &quot;1&quot;.</td>
</tr>
<tr>
<td>No.</td>
<td>Symptom</td>
<td>Possible Cause</td>
<td>Items to Check</td>
<td>Action</td>
</tr>
<tr>
<td>-----</td>
<td>---------</td>
<td>----------------</td>
<td>----------------</td>
<td>--------</td>
</tr>
<tr>
<td>4</td>
<td>Abnormal noise or vibration occurs.</td>
<td>1) Coupling is not securely tightened.</td>
<td>• Check the coupling bolts.</td>
<td>• Tighten if loose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2) A screw on the cover is loose.</td>
<td>• Check the screws used to secure the cover.</td>
<td>• Tighten if loose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3) Robot installation surface is not flat or even.</td>
<td>• Measure the degree of leveling.</td>
<td>• Correct the leveling if outside the tolerance limit.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4) Linear guide abnormality</td>
<td>• Check for debris intrusion, damage or deformation.</td>
<td>• Clean or replace the linear guide. • Check to make sure the linear guide is being used properly.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5) Ball screw abnormality</td>
<td>• Check for debris intrusion, damage or deformation.</td>
<td>• Clean or replace the ball screw. • Check to make sure the ball screw is being used properly.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6) Bearing abnormality</td>
<td>• Check for noise or vibration around the axis.</td>
<td>• Correct the assembled condition.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7) Motor failure</td>
<td>• Try replacing the motor with another one.</td>
<td>• If another motor works normally, then the currently used motor is defective so replace it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8) Improper grounding of motor case</td>
<td>• Measure to see if the resistance between the motor case and the controller's FG terminal is 1 ohm or less.</td>
<td>• If the resistance is too high, find and repair the poor connection.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9) Parameter setting error</td>
<td>• Check the parameter data.</td>
<td>• Initialize the parameters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10) Controller failure</td>
<td>• Try using another controller if available.</td>
<td>• If another controller operates normally, then the currently used controller is defective, so replace it. • Use the correct controller and robot combination.</td>
</tr>
<tr>
<td>5</td>
<td>Position deviation or offset occurs.</td>
<td>1) Coupling or pulley is not securely tightened.</td>
<td>• Check the coupling or pulley bolts.</td>
<td>• Tighten if loose.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2) Ball screw is loose.</td>
<td>• Check the ball screw.</td>
<td>• Replace the ball screw if necessary.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3) Belt is not properly engaged.</td>
<td>• Check the acceleration. • Check the amount of belt slack.</td>
<td>• Correct the parameter setting. • Adjust the belt tension.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4) Robot is not securely installed.</td>
<td>• Make sure there is no loose parts where the robot is installed.</td>
<td>• Reinstall the robot securely.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5) Malfunction caused by noise</td>
<td>• Check whether the motor case is properly grounded. • Check that the resistance between the motor case and the controller's FG terminal is 1 ohm or less, and also that the controller is properly grounded.</td>
<td>• If the controller is used near a unit that generates noise such as welding machines and electric discharge machines, move it as far away as possible. If the entire unit cannot be moved, then at least move the power supply away. It might be necessary to use a noise filter or isolating transformer depending on the trouble.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6) The robot was moved at high speeds during power off. (higher than 3000rpm)</td>
<td></td>
<td>• Do not move the robot at high speeds while the position data is retained.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7) Controller failure</td>
<td>• Try using another controller if available.</td>
<td>• If another controller operates normally, then the currently used controller is defective so replace it.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8) Motor failure</td>
<td>• Try replacing the motor with another one.</td>
<td>• If another motor works normally, then the currently used motor is defective so replace it.</td>
</tr>
<tr>
<td>6</td>
<td>During return-to-origin, the robot stops due to alarm after striking on the stroke end (overload).</td>
<td>1) Wrong robot type number setting</td>
<td>• Connect the TPB and check the robot type number.</td>
<td>• When the parameter setting is “1” (stroke-end detection method), “2” (mark method), initialize the parameter. • When the parameter setting is “0” (sensor method), set the parameter to “1”.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2) Parameter setting error</td>
<td>• Check the origin detection method parameter (PRM13) setting.</td>
<td>• When the parameter setting is “1” (stroke-end detection method), “2” (mark method), initialize the parameter. • When the parameter setting is “0” (sensor method), set the parameter to “1”.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3) The origin position is inappropriate so the robot slider makes contact with the damper when at the origin.</td>
<td>• Use the TPB to check whether the alarm occurs before or after return-to-origin is complete. If the alarm occurs after return-to-origin is complete, the damper position is inappropriate.</td>
<td>• Adjust the origin position.</td>
</tr>
</tbody>
</table>
13-3 Troubleshooting for Specific Symptom

### Items to Check

#### 7 Robot starts moving at high speed when the power is turned on.

- **Items to Check**
  1. Motor and/or resolver are miswired.
  2. Parameter error

- **Possible Cause**
  1. Motor and/or resolver are miswired.
  2. Parameter error

- **Action**
  1. Correct the connections.
  2. Try initializing the parameters.

#### 8 Robot speed is abnormally fast or slow.

- **Items to Check**
  1. Parameter setting error
  2. Speed setting was changed.

- **Possible Cause**
  1. Parameter setting error
  2. Speed setting was changed.

- **Action**
  1. Correct the parameter.

13-3-2 Relating to the I/O

### Items to Check

#### 1 Output signal cannot be controlled.

- **Possible Cause**
  1. Wiring to external devices is incorrect.
  2. Misprogramming
  3. Output transistor is defective.

- **Items to Check**
  1. Check the wiring.
  2. Connect the TPB and check the program.
  3. Measure the voltage at the PLC input terminal.
  4. Connect the TPB and check the operation.
  5. Check that the signal pulse width is 50ms or more.

- **Action**
  1. Make the correct wiring by referring to the connection diagram.
  2. Correct the program.
  3. Replace the ERCX controller if the output transistor is defective.
  4. Increase the signal pulse width ("on" duration).
  5. Increase the delay time.

#### 2 Robot will not move even with dedicated command input.

- **Possible Cause**
  1. Return-to-origin has not yet been completed.
  2. Program cannot be run.
  3. Signal pulse width is too narrow.
  4. Time interval before inputting a dedicated command after canceling emergency stop is too short.
  5. Interlock signal remains off.
  6. Another dedicated command input is on.

- **Items to Check**
  1. Connect the TPB and check the operation.
  2. Connect the TPB and check the operation.
  3. Check that the signal pulse width is 50ms or more.
  4. After canceling emergency stop, allow at least 200ms before inputting a dedicated command before inputting a dedicated command.
  5. Check the signal by using the TPB DIO monitor.
  6. Check the signal input (by using a PLC monitor, etc.).

- **Action**
  1. Reperform return-to-origin.
  2. Eliminate the cause of error by referring to the error message.
  3. Increase the signal pulse width ("on" duration).
  4. Increase the delay time.
  5. Turn on the interlock signal.
  6. Turn off the dedicated command input.
### 13-3 Troubleshooting for Specific Symptom

#### 13-3-3 Other

<table>
<thead>
<tr>
<th>No.</th>
<th>Symptom</th>
<th>Possible Cause</th>
<th>Items to Check</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>An error occurs when the TPB is connected. The TPB cannot be used.</td>
<td>1. A dedicated I/O command input is on.</td>
<td>• Check the signal input (by using a PLC monitor, etc.).</td>
<td>Always turn off dedicated command input signals when connecting the TPB to the controller.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21. The TPB cable is broken.</td>
<td>• Check the cable wiring.</td>
<td>Replace the TPB if defective.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Try connecting another TPB if available.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Program can be input only up to No. 31, or point data can be specified only up to P254, or D9 to D15 and D05 to D012 cannot be monitored.</td>
<td>11. An old version DPB was used as the teaching box.</td>
<td>• Check that DPB version is 1.50 or later.</td>
<td>Replace the ROM to upgrade the version.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21. Communication cable specifications are wrong.</td>
<td>• Check whether the wrong cable (POPCOM cable, etc.) is being used.</td>
<td>Use the specified communication cable. (POPCOM cable is different from the communication cable.) As an alternative, transmit <code>@DPBVER 210</code> in advance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Use the specified communication cable. (POPCOM cable is different from the communication cable.) As an alternative, transmit <code>@DPBVER 210</code> in advance.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Return-to-origin cannot be performed by the mark method.</td>
<td>11. The TPB version is obsolete.</td>
<td>• Check whether the TPB version is 2.10 or later.</td>
<td>Replace the ROM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>21. The DPB version is obsolete.</td>
<td>• Check whether the DPB version is 1.60 or later.</td>
<td>Replace the ROM.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Check whether the TPB version is 1.8 or later.</td>
<td>Upgrade the version.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Use POPCOM/WIN.</td>
<td></td>
</tr>
</tbody>
</table>

### Possible Cause

1. A dedicated I/O command input is on.
2. The TPB cable is broken.
3. An old version DPB was used as the teaching box.
4. Communication cable specifications are wrong.
5. POPCOM/WIN version is obsolete.
6. POPCOM/DOS is being used.
7. POPCOM/DOS is being used.
8. The TPB version is obsolete.
9. The DPB version is obsolete.
10. The POPCOM/WIN version is obsolete.
11. Return-to-origin cannot be performed by the mark method.
12. Communication cable specifications are wrong.
13. POPCOM/WIN version is obsolete.
14. POPCOM/DOS is being used.
13-4 Displaying the Alarm History

A history of past alarms can be displayed. Up to 100 alarms can be stored in the controller. This function is available with TPB version 12.18 or later.

1) On the initial screen, press [F3] (SYS).

2) Next, press [F4] (next) to change the menu display and then press [F3] (UTL).


4) Press [F1] (ALM).
5) History numbers, time that alarms occurred (total elapsed time from controller start-up) and alarm descriptions are displayed. One screen displays the past 4 alarms in the order from the most recent alarm. Pressing the [↓] and [↑] keys displays the hidden items. Press the [STEP UP] and [STEP DOWN] keys to sequentially scroll through the alarm list.

<table>
<thead>
<tr>
<th>History number</th>
<th>Time the alarm occurred</th>
<th>Alarm description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00101,05:11:12, X04</td>
<td>POWER DOWN</td>
<td>13-4 Displaying the Alarm History</td>
</tr>
</tbody>
</table>

00 : 00101,05:11:12, X04: POWER DOWN
↓ ↓ ↓
① ② ③
① History number
② Time the alarm occurred
(The above example means that the alarm occurred 101 days, 5 hours, 11 minutes and 12 seconds after controller start-up.)
③ Alarm description
(See "13-2-2 Alarm message list".)

6) To return to the previous screen, press [ESC].
For safety purposes, always turn the power off before starting robot maintenance, cleaning or repairs, etc.
14-1 Warranty

For information on the product warranty, please contact your local agent where you purchased your product.

14-2 Replacing the System Backup Battery

If an alarm is issued indicating that the system backup battery voltage is low, replace the battery using the procedure listed below.

1. First, make a backup copy of all necessary data using a memory card or POPCOM software, because that data in the controller might be lost or destroyed during battery replacement.
2. Unplug all connectors from the controller and then remove the top cover.
3. You can now see the control board. Remove it from the controller.
4. The lithium battery is soldered to the control board. Use a desoldering tool or similar tool to remove the solder and then remove the battery from the control board.
5. Solder the new battery to the control board.
   Battery product number: CR2450THE (Toshiba Battery)
6. Install the control board back in its original position.
7. Reattach the top cover.
8. Initialize all data and then return the data you backed up into the controller.

Please note that the state of California USA has legal restrictions on the handling of manganese dioxide lithium batteries. See the following website for more information:
http://www.dtsc.ca.gov/hazardouswaste/perchlorate

14-3 Replacing the Absolute Battery

The absolute battery will wear down and must be replaced. Replace the battery when its service life has expired or when problems with backing up data occur even when the battery charge time was long enough.
Though battery wear depends on the number of charges and the ambient temperature, both B1 type and B2 type batteries should generally be replaced one and a half years after being connected to the controller.
Always charge the new battery after it is installed. The battery is automatically charged while the controller is turned on. Keep the battery charged for longer than the time listed in the table below. Since the battery charging time does not affect robot operation, the controller can be used to perform teaching, program editing and robot operation while the battery is still being charged.

<table>
<thead>
<tr>
<th>Type</th>
<th>Hours until full charge</th>
<th>Backup time</th>
</tr>
</thead>
<tbody>
<tr>
<td>B1</td>
<td>15h</td>
<td>120h</td>
</tr>
<tr>
<td>(3.6V/700mAh)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B2</td>
<td>48h</td>
<td>340h</td>
</tr>
<tr>
<td>(3.6V/2000mAh)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1) At ambient temperature of 20°C
*2) After power is off with absolute battery fully charged.

* When the absolute battery is disconnected from the controller, an alarm (24:ABS. DATA ERROR) is issued. So an alarm is always issued when the absolute battery is replaced, but this is not an error. (An alarm "23:ABS. BATL-VOLTAGE" may occur in some cases.)

Absolute batteries are recycled items. Please contact our sales office for proper disposal of used batteries.
14-4 Updating the System

YAMAHA may request, on occasion, that you update the system in your equipment. The following steps describe how to update the system. Before updating the system, you must set up a system that allows communications between the controller and a PC (personal computer). Use a communication cable which conforms to the specifications listed in “11-2 Communication Cable Specifications”.

1. First, make a backup copy of the necessary data using a memory card or POPCOM software, because the data in the controller might be lost or destroyed while updating the system.
2. With the controller started up, type “@SETUP” and press the Return (Enter) key.
3. When a response “OK” is returned from the controller, turn off the power to the controller.
4. Unplug the I/O connector from the controller.
5. With the I/O connector still unplugged, turn on the power to the controller again.
6. The controller enters the system setup mode and the YAMAHA copyright message appears on the PC screen.
7. Type “@UPDATA” and press the Return (Enter) key.
8. The controller then returns READY message, so transfer the new system data. (It will take about 5 minutes to transfer all the data.)
9. An “OK” response is returned when the system transfer is complete. Now turn off the power to the controller.
11. Turn on the power to the controller again. Type “@?VER” and press the Return (Enter) key. Then make sure that the controller version is updated.
12. Initialize all data and then return the data you backed up into the controller.

⚠️ CAUTION

- The controller must remain in emergency stop until updating of the system is finished. (Specifically, terminals A24 (EMG1) and B24 (EMG2) of the I/O connector should be left open.)
- Before starting the system updating, we strongly recommend for safety reasons that the robot cable be disconnected from the controller.
15-1 ERCX series

15-1-1 Basic specifications

<table>
<thead>
<tr>
<th>Specification Item</th>
<th>Model</th>
<th>ERCX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic specifications</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applicable motor capacitance</td>
<td>DC24V, 30W or less</td>
<td></td>
</tr>
<tr>
<td>External dimensions</td>
<td>W30 x H250 x D157mm</td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>0.9kg</td>
<td></td>
</tr>
<tr>
<td>Power supply voltage</td>
<td>DC24V±10%, 3A to 4.5A (Depends on robot type)</td>
<td></td>
</tr>
<tr>
<td>No. of controllable axes</td>
<td>1 axis</td>
<td></td>
</tr>
<tr>
<td>Control method</td>
<td>AC full digital servo PTP</td>
<td></td>
</tr>
<tr>
<td>Position detection method</td>
<td>Resolver with multi-turn data backup function</td>
<td></td>
</tr>
<tr>
<td>Speed setting</td>
<td>100-step setting possible per program step</td>
<td></td>
</tr>
<tr>
<td>Acceleration setting</td>
<td>Automatically set according to robot type and payload. 100-step setting is also possible with acceleration parameter.</td>
<td></td>
</tr>
<tr>
<td>Servo adjustment</td>
<td>Handled with parameters (special). Servo gain, current limit, etc.</td>
<td></td>
</tr>
<tr>
<td>No. of encoder pulses</td>
<td>16384P/R</td>
<td></td>
</tr>
<tr>
<td>Lead length</td>
<td>Lead length is selectable during initialization or by parameter setting</td>
<td></td>
</tr>
<tr>
<td>Axis control</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of multi tasks</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Origin detection method</td>
<td>Origin search by stroke end detection or absolute reset</td>
<td></td>
</tr>
<tr>
<td>Teaching method</td>
<td>MDI (coordinate value input), teaching playback, direct teaching</td>
<td></td>
</tr>
<tr>
<td>Auxiliary memory unit</td>
<td>IC memory card is available as TPB option</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROM</td>
<td>256K bytes (with CPU incorporated)</td>
<td></td>
</tr>
<tr>
<td>RAM</td>
<td>128K bytes with 64K lithium battery backup (5-year life)</td>
<td></td>
</tr>
<tr>
<td>No. of program steps</td>
<td>3000 steps or less in total, 255 steps/program</td>
<td></td>
</tr>
<tr>
<td>No. of programs</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>No. of points</td>
<td>1000 points</td>
<td></td>
</tr>
<tr>
<td>I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O input</td>
<td>General-purpose: 16 points, dedicated input: 8 points</td>
<td></td>
</tr>
<tr>
<td>I/O output</td>
<td>General-purpose: 13 points, dedicated output: 3 points, open collector output (0.1A/24V maximum per output)</td>
<td></td>
</tr>
<tr>
<td>Drive power supply</td>
<td>External DC24V±10%, more than 50mA*2</td>
<td></td>
</tr>
<tr>
<td>Brake output</td>
<td>Relay output (for 24V/300mA brake), one channel</td>
<td></td>
</tr>
<tr>
<td>Emergency stop input</td>
<td>Normally closed contact input (origin return not required after emergency stop is released)</td>
<td></td>
</tr>
<tr>
<td>Serial interface</td>
<td>One RS-232C channel (for communication with TPB or PC)</td>
<td></td>
</tr>
<tr>
<td>Network (option)</td>
<td>CC-Link, DeviceNet, Ethernet, PROFIBUS</td>
<td></td>
</tr>
<tr>
<td>General specification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ambient temperature</td>
<td>0 to 40°C</td>
<td></td>
</tr>
<tr>
<td>Storage temperature</td>
<td>-10 to 65°C</td>
<td></td>
</tr>
<tr>
<td>Ambient humidity</td>
<td>35 to 85%RH (no condensation)</td>
<td></td>
</tr>
<tr>
<td>Noise immunity</td>
<td>Conforms to IEC61000-4-4 Level 2</td>
<td></td>
</tr>
</tbody>
</table>

*1: When controlling a brake and I/O operations, a separate 24V power supply with the necessary capacity must be connected to the I/O connectors. A 24V power supply with higher capacity can boost robot performance. Please consult us for more details.

*2: When no brake and I/O control are used. If a brake and I/O control are used, an additional power supply with the necessary capacity will be required.

⚠️ CAUTION

Specifications and external appearance are subject to change without prior notice.
15-1-2 Robot number list

Each robot model has an identification number as listed in the table below. After you initialize the parameters, enter the correct robot number that matches the robot model actually connected to the controller.

**Single-axis robot**

<table>
<thead>
<tr>
<th></th>
<th>T4</th>
<th>T5</th>
<th>C4</th>
<th>C5</th>
<th>YMS45</th>
<th>YMS55</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>90</td>
<td>94</td>
<td>90</td>
<td>94</td>
<td>910</td>
<td>916</td>
</tr>
<tr>
<td>-BK (vertical installation model)</td>
<td>91</td>
<td>95</td>
<td>91</td>
<td>95</td>
<td>911</td>
<td>917</td>
</tr>
</tbody>
</table>

15-1-3 LED display

The table below shows the specifications of the operation status LED on the front panel of the controller.

<table>
<thead>
<tr>
<th>LED display</th>
<th>Robot or controller operation status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not lit</td>
<td>The power is off or fuse is blown.</td>
</tr>
<tr>
<td>Lit in green</td>
<td>Servo motor is on. (Ready to operate)</td>
</tr>
<tr>
<td>Lit in red</td>
<td>Error occurred. (Alarm is being issued.)</td>
</tr>
<tr>
<td>Flashes green (0.5 sec.) and red (0.5 sec.)</td>
<td>Emergency stop</td>
</tr>
<tr>
<td>Flashes green (1.5 sec.) and red (0.5 sec.)</td>
<td>Emergency stop is canceled. (Servo off)</td>
</tr>
</tbody>
</table>

15-1-4 Absolute Battery Unit

The absolute battery unit has the following basic specifications. (Either B1 or B2 type is supplied depending on the request by the user.)

<table>
<thead>
<tr>
<th>Specification item</th>
<th>Type</th>
<th>B1</th>
<th>B2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic specifications</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Battery type</td>
<td>Ni-Cd battery</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Charge method</td>
<td>Trickle charge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Battery capacity</td>
<td>3.6V/700mAh</td>
<td>3.6V/2000mAh</td>
<td></td>
</tr>
<tr>
<td>Dimensions</td>
<td>W47 × H52 × D15mm</td>
<td>L152 × φ29mm</td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>80g</td>
<td>280g</td>
<td></td>
</tr>
<tr>
<td><strong>Characteristics</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data backup time *1</td>
<td>120h</td>
<td>340h</td>
<td></td>
</tr>
<tr>
<td>Hours until full charge *2</td>
<td>15h</td>
<td>48h</td>
<td></td>
</tr>
<tr>
<td>Service life</td>
<td>About 1.5 years</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>General</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cable length</td>
<td>300mm (standard)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accessories</td>
<td>Snap band (plastic fastening tie) and band fastener, 2 pieces each</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1) After power is off with the absolute battery fully charged.
*2) At ambient temperature of 20°C
## 15-2 TPB

### 15-2-1 Basic specifications

<table>
<thead>
<tr>
<th>Specification item</th>
<th>Model</th>
<th>TPB</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic specifications</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>External dimensions</td>
<td>W107 ( \times ) H235 ( \times ) D47mm</td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>590g</td>
<td></td>
</tr>
<tr>
<td>Power consumption</td>
<td>5 V, 200 mA max.</td>
<td></td>
</tr>
<tr>
<td>Power supply</td>
<td>DC12V (supplied from the controller)</td>
<td></td>
</tr>
<tr>
<td>Cable length</td>
<td>Standard 3.5m</td>
<td></td>
</tr>
<tr>
<td><strong>I/O</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial interface</td>
<td>RS-232C, one channel, for communications with controller</td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td>Liquid crystal, 20 characters ( \times ) 4 lines</td>
<td></td>
</tr>
<tr>
<td>Keyboard</td>
<td>29 keys, membrane switch + emergency stop button</td>
<td></td>
</tr>
<tr>
<td>Emergency stop button</td>
<td>Normally-closed contact (with lock function)</td>
<td></td>
</tr>
<tr>
<td>Auxiliary memory device</td>
<td>IC memory card</td>
<td></td>
</tr>
<tr>
<td><strong>General specification</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ambient temperature</td>
<td>0°C to 40°C</td>
<td></td>
</tr>
<tr>
<td>Storage temperature</td>
<td>-10°C to 65°C</td>
<td></td>
</tr>
<tr>
<td>Ambient humidity</td>
<td>35 to 85% RH (no condensation)</td>
<td></td>
</tr>
<tr>
<td>Noise immunity</td>
<td>Conforms to IEC61000-4-4 Level 2</td>
<td></td>
</tr>
</tbody>
</table>
16-1 Operation When Not Using Absolute Function

An absolute backup function is standard on the ERCX controllers. This means return-to-origin is unnecessary each time the controller is turned on.
In some cases, however, the absolute function is not needed because the customer is accustomed to always performing return-to-origin when the power is turned on. In other cases the absolute function may not be needed because the current sequence system can be used to automatically perform return-to-origin when the power is turned on.
In such kind of cases where the absolute function is not needed, use the following procedure to operate the controller with the absolute battery removed.

(1) Set bit 4 to 0 in PRM34 (system mode selection parameter) to disable the absolute backup function.
   Bit 4 is set to 1 as the default value so that PRM34 = 16. If bits other than bit 4 can stay the default value without causing problems, just enter 0 in PRM34.
   (See "PRM34: System mode selection"
(2) Turn off power to the controller and remove the absolute battery from the controller.

* Once the absolute backup function has been disabled, always be sure to perform return-to-origin when the controller power is turned on.
* Even if the absolute battery is removed, the various data such as the program, point data, and parameters stored inside the controller will not be lost.
(The various data such as the program, point data, and parameters are retained by a system backup battery inside the controller. The absolute battery is used to hold the robot position data when the controller power is off.)
16-2 How to Handle Options

16-2-1 Memory card

A memory card (option) can be used with the TPB to back up the ERCX controller data.

- Using the memory card
  1. Insert the memory card into the TPB as shown in Fig. 16-1.
  2. Back up the data by referring to section "10-6 Using a Memory Card" in Chapter 10.

- Precautions when using the memory card
  1. Insert the memory card all the way inwards until you feel it makes contact.
  2. Be careful not to insert the memory card facing the wrong direction. The mark "Δ" should be facing upward. (A pin for preventing reverse insertion is provided.)
  3. Insert or pull out the memory card only when the power is supplied to the TPB.
  4. Never eject the memory card while backing up data.
  5. The memory card should be used under the following environmental conditions:
     - Ambient temperature range : -10 to 40°C
     - Ambient humidity range : Below 85% RH
     - Storage temperature range : -20 to 60°C
  6. Do not leave the memory card inserted in the TPB when not in use, since this will shorten the battery life.

    The battery life is about 5 years (at ambient temperature of 25°C).
    If the battery voltage drops, an alert message appears on the TPB, so replace the battery by referring to Fig. 16-2.
    Battery product number : Panasonic BR2325 or CR2325 (64KB card) or equivalent type
Data size that can be saved

Data size that can be saved on one memory card is as follows:

<table>
<thead>
<tr>
<th>Memory card capacity</th>
<th>Save format</th>
<th>DPB Ver. 1.52 or later</th>
<th>TPB Ver. 2.04 to 2.18</th>
<th>TPB Ver. 12.12 or later</th>
</tr>
</thead>
<tbody>
<tr>
<td>8KB</td>
<td>Standard</td>
<td>Cannot be used.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compatible*</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64KB</td>
<td>Standard</td>
<td>Up to 3 units of ERCX</td>
<td>Up to 3 units of ERCX</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compatible*</td>
<td>Up to 10 units of ERCX</td>
<td>Up to 10 units of ERCX</td>
<td></td>
</tr>
<tr>
<td>1024KB (1MB)</td>
<td>Standard</td>
<td>Up to 3 units of ERCX</td>
<td>Up to 3 units of ERCX</td>
<td>Up to 48 units of ERCX</td>
</tr>
<tr>
<td></td>
<td>Compatible*</td>
<td>Up to 10 units of ERCX</td>
<td>Up to 10 units of ERCX</td>
<td>Up to 160 units of ERCX</td>
</tr>
</tbody>
</table>

* ERCX data can be shared with SRC and SRCA by making the save format compatible with each other. Please note, however, that there are the following limitations.

- Number of steps in total: 1024 steps
- Maximum number of points: 255 points (P0 to P254)
- Maximum number of programs: 32 programs (program No. 0 to No. 31)
16-2-2 Handling the I/O Checker

This device connects to the I/O connector of the ERCX controller and is used for pseudo-input (emulation) by means of switches and for input/output monitoring by LED display.

- **Connecting the I/O checker**
  1. Plug the connector marked "ROBOT I/O" into the I/O connector of the ERCX controller.
  2. Plug the connector marked "JAE 50P" into connector CN1 on the right of the I/O Checker and make sure it locks.
  3. Plug the connector, which is normally connected to the I/O connector of the ERCX controller, into connector CN2 on the left of the I/O Checker.

- **Operation method**
  1. The LED monitor turns on (lights) and off (goes out) in conjunction with the input and output.
  2. The pseudo-input switch is on when set to the upper side and off when set to the lower side. However, the INTERLOCK and EMG inputs are opposite; they are on when set to the lower side and off when set to the upper side.
    Thus, if all of the switches are set to the lower side at first, the unit can be used for pseudo-input and as an I/O monitor.
  3. The input changeover switch should be set to the EXTERNAL (upper) side to receive external input from a PLC or a similar unit.
    If the switch is set to the INTERNAL (lower) side, the switch signals on the I/O checker are input to the controller. In either case, the input monitor is handled by means of LEDs.
16-2 How to Handle Options

16-2-3 POPCOM communication cable

This cable is used to operate the ERCX controller from POPCOM software which runs on a PC and allows easy and efficient robot programming and operation. This POPCOM cable is different from typical communication cables, so do not use it for other purposes.

Pins 18 and 21 on the ERCX controller are used for emergency stop input. Install a normally closed (contact B) switch of at least 50mA capacity between these pins when using emergency stop from the PC. Emergency stop is triggered when the switch opens the contact between pins 18 and 21.

Input response: 5ms or less
Input current: 33.3mA (DC24V)

When the PC has a D-sub 25-pin connector:

<table>
<thead>
<tr>
<th>Controller</th>
<th>PC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal name</td>
<td>Pin No.</td>
</tr>
<tr>
<td>F.G</td>
<td>1</td>
</tr>
<tr>
<td>TXD</td>
<td>2</td>
</tr>
<tr>
<td>RXD</td>
<td>3</td>
</tr>
<tr>
<td>RTS</td>
<td>4</td>
</tr>
<tr>
<td>CTS</td>
<td>5</td>
</tr>
<tr>
<td>D.G</td>
<td>7</td>
</tr>
<tr>
<td>HSTCK</td>
<td>12</td>
</tr>
<tr>
<td>HSES1</td>
<td>18</td>
</tr>
<tr>
<td>HSES2</td>
<td>21</td>
</tr>
</tbody>
</table>

When the PC has a D-sub 9-pin connector:

<table>
<thead>
<tr>
<th>Controller</th>
<th>PC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal name</td>
<td>Pin No.</td>
</tr>
<tr>
<td>F.G</td>
<td>1</td>
</tr>
<tr>
<td>TXD</td>
<td>2</td>
</tr>
<tr>
<td>RXD</td>
<td>3</td>
</tr>
<tr>
<td>RTS</td>
<td>4</td>
</tr>
<tr>
<td>CTS</td>
<td>5</td>
</tr>
<tr>
<td>D.G</td>
<td>7</td>
</tr>
<tr>
<td>HSTCK</td>
<td>12</td>
</tr>
<tr>
<td>HSES1</td>
<td>18</td>
</tr>
<tr>
<td>HSES2</td>
<td>21</td>
</tr>
</tbody>
</table>

The SHELL means a metallic casing of the connector.

⚠️ CAUTION
Pin 10 of the connector on the controller is used exclusively for connecting to the TPB. To prevent problems, do not attempt to wire anything to pin 10.
Revision record

<table>
<thead>
<tr>
<th>Manual version</th>
<th>Issue date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver. 6.02</td>
<td>Jan. 2007</td>
<td>Clerical error corrections.</td>
</tr>
<tr>
<td>Ver. 6.03</td>
<td>Jul. 2007</td>
<td>Cautions about loading the data from the controller were added. Clerical error corrections, etc.</td>
</tr>
<tr>
<td>Ver. 6.04</td>
<td>Oct. 2007</td>
<td>Addition of error message and caution about loading of unsuitable robot data. Addition of customize function for END output timing in execution of dedicated I/O command. Expansion of I/O assignment function (addition of movement point zone output function, etc.). Clerical error corrections, etc.</td>
</tr>
<tr>
<td></td>
<td>Jun. 2011</td>
<td>The manual's version number was changed to match that for the Japanese manual.</td>
</tr>
<tr>
<td>Ver. 7.08</td>
<td>Jun. 2011</td>
<td>The description regarding &quot;Warranty&quot; was changed.</td>
</tr>
</tbody>
</table>

User's Manual

YAMAHA ERCX

Jun. 2011
Ver. 7.08
This manual is based on Ver. 7.08 of Japanese manual.

© YAMAHA MOTOR CO., LTD.
IM Operations

All rights reserved. No part of this publication may be reproduced in any form without the permission of YAMAHA MOTOR CO., LTD. Information furnished by YAMAHA in this manual is believed to be reliable. However, no responsibility is assumed for possible inaccuracies or omissions. If you find any part unclear in this manual, please contact YAMAHA or YAMAHA sales representatives.