

FANUC Robot **series**

R-J3/R-J3iB/R-30iA CONTROLLER

CIMPLICITY HMI for Robots

OPERATOR'S MANUAL

MAROCCHMI04071E REV. A

This publication contains proprietary information of FANUC Robotics America, Inc. furnished for customer use only. No other uses are authorized without the express written permission of FANUC Robotics America, Inc.

FANUC Robotics America, Inc.
3900 W. Hamlin Road
Rochester Hills, Michigan 48309-3253

B-82604EN/01

The descriptions and specifications contained in this manual were in effect at the time this manual was approved for printing. FANUC Robotics America, Inc, hereinafter referred to as FANUC Robotics, reserves the right to discontinue models at any time or to change specifications or design without notice and without incurring obligations.

FANUC Robotics manuals present descriptions, specifications, drawings, schematics, bills of material, parts, connections and/or procedures for installing, disassembling, connecting, operating and programming FANUC Robotics' products and/or systems. Such systems consist of robots, extended axes, robot controllers, application software, the KAREL® programming language, INSIGHT® vision equipment, and special tools.

FANUC Robotics recommends that only persons who have been trained in one or more approved FANUC Robotics Training Course(s) be permitted to install, operate, use, perform procedures on, repair, and/or maintain FANUC Robotics' products and/or systems and their respective components. Approved training necessitates that the courses selected be relevant to the type of system installed and application performed at the customer site.

WARNING

This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instruction manual, may cause interference to radio communications. As temporarily permitted by regulation, it has not been tested for compliance with the limits for Class A computing devices pursuant to subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of the equipment in a residential area is likely to cause interference, in which case the user, at his own expense, will be required to take whatever measure may be required to correct the interference.

FANUC Robotics conducts courses on its systems and products on a regularly scheduled basis at its headquarters in Rochester Hills, Michigan. For additional information contact

FANUC Robotics America, Inc.
Training Department
3900 W. Hamlin Road
Rochester Hills, Michigan 48309-3253
www.fanucrobotics.com

Send your comments and suggestions about this manual to:
product.documentation@fanucrobotics.com

**Copyright ©2007 by FANUC Robotics America, Inc.
All Rights Reserved**

The information illustrated or contained herein is not to be reproduced, copied, downloaded, translated into another language, published in any physical or electronic format, including internet, or transmitted in whole or in part in any way without the prior written consent of FANUC Robotics America, Inc.

AccuStat®, ArcTool®, DispenseTool®, FANUC LASER DRILL®, KAREL®, INSIGHT®, INSIGHT II®, PaintTool®, PaintWorks®, PalletTool®, SOCKETS®, SOFT PARTS®, SpotTool®, TorchMate®, and YagTool® are Registered Trademarks of FANUC Robotics.

FANUC Robotics reserves all proprietary rights, including but not limited to trademark and trade name rights, in the following names:

AccuAir™ AccuCal™ AccuChop™ AccuFlow™ AccuPath™
AccuSeal™ ARC Mate™ ARC Mate Sr.™ ARC Mate System 1™
ARC Mate System 2™ ARC Mate System 3™ ARC Mate System
4™ ARC Mate System 5™ ARCWorks Pro™ AssistTool™
AutoNormal™ AutoTCP™ BellTool™ BODYWorks™ Cal Mate™ Cell
Finder™ Center Finder™ Clean Wall™ CollisionGuard™
DispenseTool™ F-100™ F-200i™ FabTool™ FANUC LASER
DRILL™ Flexibell™ FlexTool™ HandlingTool™ HandlingWorks™
INSIGHT™ INSIGHT II™ IntelliTrak™ Integrated Process Solution™
Intelligent Assist Device™ IPC -Integrated Pump Control™ IPD
Integral Pneumatic Dispenser™ ISA Integral Servo Applicator™ ISD
Integral Servo Dispenser™ Laser Mate System 3™ Laser Mate
System 4™ LaserPro™ LaserTool™ LR Tool™ MIG Eye™
MotionParts™ NoBots™ Paint Stick™ PaintPro™ PaintTool 100™
PAINTWorks™ PAINTWorks II™ PAINTWorks III™ PalletMate™
PalletMate PC™ PalletTool PC™ PayloadID™ RecipTool™
RemovalTool™ Robo Chop™ Robo Spray™ S-420i™ S-430i™
ShapeGen™ SoftFloat™ SOF™ PARTS™ SpotTool+™ SR Mate™
SR ShotTool™ SureWeld™ SYSTEM R-J2 Controller™ SYSTEM R-
J3 Controller™ SYSTEM R-J3iB Controller™ TCP Mate™
TurboMove™ TorchMate™ visLOC™ visPRO-3D™ visTRAC™
WebServer™ WebTP™ YagTool™

© FANUC LTD 2007

- No part of this manual may be reproduced in any form.
- All specifications and designs are subject to change without notice.

Conventions

This manual includes information essential to the safety of personnel, equipment, software, and data. This information is indicated by headings and boxes in the text.



WARNING

Information appearing under **WARNING** concerns the protection of personnel. It is boxed and in bold type to set it apart from other text.



CAUTION

Information appearing under **CAUTION** concerns the protection of equipment, software, and data. It is boxed to set it apart from other text.

NOTE Information appearing next to **NOTE** concerns related information or useful hints.

Before using the Robot, be sure to read the "FANUC Robot Safety Manual (B-80687EN)" and understand the content.

This manual can be used with controllers labeled R-30iA or R-J3iC. If you have a controller labeled R-J3iC, you should read R-30iA as R-J3iC throughout this manual.

- No part of this manual may be reproduced in any form.
- All specifications and designs are subject to change without notice.

The products in this manual are controlled based on Japan's "Foreign Exchange and Foreign Trade Law". The export from Japan may be subject to an export license by the government of Japan.

Further, re-export to another country may be subject to the license of the government of the country from where the product is re-exported. Furthermore, the product may also be controlled by re-export regulations of the United States government.

Should you wish to export or re-export these products, please contact FANUC for advice.

In this manual we have tried as much as possible to describe all the various matters.

However, we cannot describe all the matters which must not be done, or which cannot be done, because there are so many possibilities.

Therefore, matters which are not especially described as possible in this manual should be regarded as "impossible".

Safety

FANUC Robotics is not and does not represent itself as an expert in safety systems, safety equipment, or the specific safety aspects of your company and/or its work force. It is the responsibility of the owner, employer, or user to take all necessary steps to guarantee the safety of all personnel in the workplace.

The appropriate level of safety for your application and installation can best be determined by safety system professionals. FANUC Robotics therefore, recommends that each customer consult with such professionals in order to provide a workplace that allows for the safe application, use, and operation of FANUC Robotic systems.

According to the industry standard ANSI/RIA R15.06, the owner or user is advised to consult the standards to ensure compliance with its requests for Robotics System design, usability, operation, maintenance, and service. Additionally, as the owner, employer, or user of a robotic system, it is your responsibility to arrange for the training of the operator of a robot system to recognize and respond to known hazards associated with your robotic system and to be aware of the recommended operating procedures for your particular application and robot installation.

FANUC Robotics therefore, recommends that all personnel who intend to operate, program, repair, or otherwise use the robotics system be trained in an approved FANUC Robotics training course and become familiar with the proper operation of the system. Persons responsible for programming the system—including the design, implementation, and debugging of application programs—must be familiar with the recommended programming procedures for your application and robot installation.

The following guidelines are provided to emphasize the importance of safety in the workplace.

CONSIDERING SAFETY FOR YOUR ROBOT INSTALLATION

Safety is essential whenever robots are used. Keep in mind the following factors with regard to safety:

- The safety of people and equipment
- Use of safety enhancing devices
- Techniques for safe teaching and manual operation of the robot(s)
- Techniques for safe automatic operation of the robot(s)
- Regular scheduled inspection of the robot and workcell
- Proper maintenance of the robot

Keeping People and Equipment Safe

The safety of people is always of primary importance in any situation. However, equipment must be kept safe, too. When prioritizing how to apply safety to your robotic system, consider the following:

- People
- External devices
- Robot(s)
- Tooling
- Workpiece

Using Safety Enhancing Devices

Always give appropriate attention to the work area that surrounds the robot. The safety of the work area can be enhanced by the installation of some or all of the following devices:

- Safety fences, barriers, or chains
- Light curtains
- Interlocks
- Pressure mats
- Floor markings
- Warning lights
- Mechanical stops
- EMERGENCY STOP buttons
- DEADMAN switches

Setting Up a Safe Workcell

A safe workcell is essential to protect people and equipment. Observe the following guidelines to ensure that the workcell is set up safely. These suggestions are intended to supplement and **not** replace existing federal, state, and local laws, regulations, and guidelines that pertain to safety.

- Sponsor your personnel for training in approved FANUC Robotics training course(s) related to your application. Never permit untrained personnel to operate the robots.

- Install a lockout device that uses an access code to prevent unauthorized persons from operating the robot.
- Use anti-tie-down logic to prevent the operator from bypassing safety measures.
- Arrange the workcell so the operator faces the workcell and can see what is going on inside the cell.
- Clearly identify the work envelope of each robot in the system with floor markings, signs, and special barriers. The work envelope is the area defined by the maximum motion range of the robot, including any tooling attached to the wrist flange that extend this range.
- Position all controllers outside the robot work envelope.
- Never rely on software as the primary safety element.
- Mount an adequate number of EMERGENCY STOP buttons or switches within easy reach of the operator and at critical points inside and around the outside of the workcell.
- Install flashing lights and/or audible warning devices that activate whenever the robot is operating, that is, whenever power is applied to the servo drive system. Audible warning devices shall exceed the ambient noise level at the end-use application.
- Wherever possible, install safety fences to protect against unauthorized entry by personnel into the work envelope.
- Install special guarding that prevents the operator from reaching into restricted areas of the work envelope.
- Use interlocks.
- Use presence or proximity sensing devices such as light curtains, mats, and capacitance and vision systems to enhance safety.
- Periodically check the safety joints or safety clutches that can be optionally installed between the robot wrist flange and tooling. If the tooling strikes an object, these devices dislodge, remove power from the system, and help to minimize damage to the tooling and robot.

- Make sure all external devices are properly filtered, grounded, shielded, and suppressed to prevent hazardous motion due to the effects of electro–magnetic interference (EMI), radio frequency interference (RFI), and electro–static discharge (ESD).
- Make provisions for power lockout/tagout at the controller.
- Eliminate *pinch points*. Pinch points are areas where personnel could get trapped between a moving robot and other equipment.
- Provide enough room inside the workcell to permit personnel to teach the robot and perform maintenance safely.
- Program the robot to load and unload material safely.
- If high voltage electrostatics are present, be sure to provide appropriate interlocks, warning, and beacons.
- If materials are being applied at dangerously high pressure, provide electrical interlocks for lockout of material flow and pressure.

Staying Safe While Teaching or Manually Operating the Robot

Advise all personnel who must teach the robot or otherwise manually operate the robot to observe the following rules:

- Never wear watches, rings, neckties, scarves, or loose clothing that could get caught in moving machinery.
- Know whether or not you are using an intrinsically safe teach pendant if you are working in a hazardous environment.
- Before teaching, visually inspect the robot and *work envelope* to make sure that no potentially hazardous conditions exist. The work envelope is the area defined by the maximum motion range of the robot. These include tooling attached to the wrist flange that extends this range.
- The area near the robot must be clean and free of oil, water, or debris. Immediately report unsafe working conditions to the supervisor or safety department.
- FANUC Robotics recommends that no one enter the work envelope of a robot that is on, except for robot teaching operations. However, if you must enter the work envelope, be sure all safeguards are in place, check the teach pendant DEADMAN switch for proper operation, and place the robot in teach mode. Take the teach pendant with you, turn it on, and be prepared to release the DEADMAN switch. Only the person with the teach pendant should be in the work envelope.



WARNING

Never bypass, strap, or otherwise deactivate a safety device, such as a limit switch, for any operational convenience. Deactivating a safety device is known to have resulted in serious injury and death.

- Know the path that can be used to escape from a moving robot; make sure the escape path is never blocked.
- Isolate the robot from all remote control signals that can cause motion while data is being taught.
- Test any program being run for the first time in the following manner:



WARNING

Stay outside the robot work envelope whenever a program is being run. Failure to do so can result in injury.

- Using a low motion speed, single step the program for at least one full cycle.
- Using a low motion speed, test run the program continuously for at least one full cycle.
- Using the programmed speed, test run the program continuously for at least one full cycle.
- Make sure all personnel are outside the work envelope before running production.

Staying Safe During Automatic Operation

Advise all personnel who operate the robot during production to observe the following rules:

- Make sure all safety provisions are present and active.
- Know the entire workcell area. The workcell includes the robot and its work envelope, plus the area occupied by all external devices and other equipment with which the robot interacts.
- Understand the complete task the robot is programmed to perform before initiating automatic operation.
- Make sure all personnel are outside the work envelope before operating the robot.

- Never enter or allow others to enter the work envelope during automatic operation of the robot.
- Know the location and status of all switches, sensors, and control signals that could cause the robot to move.
- Know where the EMERGENCY STOP buttons are located on both the robot control and external control devices. Be prepared to press these buttons in an emergency.
- Never assume that a program is complete if the robot is not moving. The robot could be waiting for an input signal that will permit it to continue activity.
- If the robot is running in a pattern, do not assume it will continue to run in the same pattern.
- Never try to stop the robot, or break its motion, with your body. The only way to stop robot motion immediately is to press an EMERGENCY STOP button located on the controller panel, teach pendant, or emergency stop stations around the workcell.

Staying Safe During Inspection

When inspecting the robot, be sure to

- Turn off power at the controller.
- Lock out and tag out the power source at the controller according to the policies of your plant.
- Turn off the compressed air source and relieve the air pressure.
- If robot motion is not needed for inspecting the electrical circuits, press the EMERGENCY STOP button on the operator panel.
- Never wear watches, rings, neckties, scarves, or loose clothing that could get caught in moving machinery.
- If power is needed to check the robot motion or electrical circuits, be prepared to press the EMERGENCY STOP button, in an emergency.
- Be aware that when you remove a servomotor or brake, the associated robot arm will fall if it is not supported or resting on a hard stop. Support the arm on a solid support before you release the brake.

Staying Safe During Maintenance

When performing maintenance on your robot system, observe the following rules:

- Never enter the work envelope while the robot or a program is in operation.
- Before entering the work envelope, visually inspect the workcell to make sure no potentially hazardous conditions exist.
- Never wear watches, rings, neckties, scarves, or loose clothing that could get caught in moving machinery.
- Consider all or any overlapping work envelopes of adjoining robots when standing in a work envelope.
- Test the teach pendant for proper operation before entering the work envelope.
- If it is necessary for you to enter the robot work envelope while power is turned on, you must be sure that you are in control of the robot. Be sure to take the teach pendant with you, press the DEADMAN switch, and turn the teach pendant on. Be prepared to release the DEADMAN switch to turn off servo power to the robot immediately.
- Whenever possible, perform maintenance with the power turned off. Before you open the controller front panel or enter the work envelope, turn off and lock out the 3-phase power source at the controller.
- Be aware that when you remove a servomotor or brake, the associated robot arm will fall if it is not supported or resting on a hard stop. Support the arm on a solid support before you release the brake.

**WARNING**

Lethal voltage is present in the controller WHENEVER IT IS CONNECTED to a power source. Be extremely careful to avoid electrical shock.

HIGH VOLTAGE IS PRESENT at the input side whenever the controller is connected to a power source. Turning the disconnect or circuit breaker to the OFF position removes power from the output side of the device only.

- Release or block all stored energy. Before working on the pneumatic system, shut off the system air supply and purge the air lines.

- Isolate the robot from all remote control signals. If maintenance must be done when the power is on, make sure the person inside the work envelope has sole control of the robot. The teach pendant must be held by this person.
- Make sure personnel cannot get trapped between the moving robot and other equipment. Know the path that can be used to escape from a moving robot. Make sure the escape route is never blocked.
- Use blocks, mechanical stops, and pins to prevent hazardous movement by the robot. Make sure that such devices do not create pinch points that could trap personnel.



WARNING

Do not try to remove any mechanical component from the robot before thoroughly reading and understanding the procedures in the appropriate manual. Doing so can result in serious personal injury and component destruction.

- Be aware that when you remove a servomotor or brake, the associated robot arm will fall if it is not supported or resting on a hard stop. Support the arm on a solid support before you release the brake.
- When replacing or installing components, make sure dirt and debris do not enter the system.
- Use only specified parts for replacement. To avoid fires and damage to parts in the controller, never use nonspecified fuses.
- Before restarting a robot, make sure no one is inside the work envelope; be sure that the robot and all external devices are operating normally.

KEEPING MACHINE TOOLS AND EXTERNAL DEVICES SAFE

Programming Safety Precautions

Certain programming and mechanical measures are useful in keeping the machine tools and other external devices safe. Some of these measures are outlined below. Make sure you know all associated measures for safe use of such devices.

Implement the following programming safety measures to prevent damage to machine tools and other external devices.

- Back-check limit switches in the workcell to make sure they do not fail.
- Implement “failure routines” in programs that will provide appropriate robot actions if an external device or another robot in the workcell fails.
- Use *handshaking* protocol to synchronize robot and external device operations.
- Program the robot to check the condition of all external devices during an operating cycle.

Mechanical Safety Precautions

Implement the following mechanical safety measures to prevent damage to machine tools and other external devices.

- Make sure the workcell is clean and free of oil, water, and debris.
- Use software limits, limit switches, and mechanical hardstops to prevent undesired movement of the robot into the work area of machine tools and external devices.

KEEPING THE ROBOT SAFE

Operating Safety Precautions

Observe the following operating and programming guidelines to prevent damage to the robot.

The following measures are designed to prevent damage to the robot during operation.

- Use a low override speed to increase your control over the robot when jogging the robot.
- Visualize the movement the robot will make before you press the jog keys on the teach pendant.
- Make sure the work envelope is clean and free of oil, water, or debris.
- Use circuit breakers to guard against electrical overload.

Programming Safety Precautions

The following safety measures are designed to prevent damage to the robot during programming:

- Establish *interference zones* to prevent collisions when two or more robots share a work area.

- Make sure that the program ends with the robot near or at the home position.
- Be aware of signals or other operations that could trigger operation of tooling resulting in personal injury or equipment damage.
- In dispensing applications, be aware of all safety guidelines with respect to the dispensing materials.

NOTE Any deviation from the methods and safety practices described in this manual must conform to the approved standards of your company. If you have questions, see your supervisor.

ADDITIONAL SAFETY CONSIDERATIONS FOR PAINT ROBOT INSTALLATIONS

Process technicians are sometimes required to enter the paint booth, for example, during daily or routine calibration or while teaching new paths to a robot. Maintenance personnel also must work inside the paint booth periodically.

Whenever personnel are working inside the paint booth, ventilation equipment must be used. Instruction on the proper use of ventilating equipment usually is provided by the paint shop supervisor.

Although paint booth hazards have been minimized, potential dangers still exist. Therefore, today's highly automated paint booth requires that process and maintenance personnel have full awareness of the system and its capabilities. They must understand the interaction that occurs between the vehicle moving along the conveyor and the robot(s), hood/deck and door opening devices, and high-voltage electrostatic tools.

Paint robots are operated in three modes:

- Teach or manual mode
- Automatic mode, including automatic and exercise operation
- Diagnostic mode

During both teach and automatic modes, the robots in the paint booth will follow a predetermined pattern of movements. In teach mode, the process technician teaches (programs) paint paths using the teach pendant.

In automatic mode, robot operation is initiated at the System Operator Console (SOC) or Manual Control Panel (MCP), if available, and can be monitored from outside the paint booth. All personnel must remain outside of the booth or in a designated safe

area within the booth whenever automatic mode is initiated at the SOC or MCP.

In automatic mode, the robots will execute the path movements they were taught during teach mode, but generally at production speeds.

When process and maintenance personnel run diagnostic routines that require them to remain in the paint booth, they must stay in a designated safe area.

Paint System Safety Features

Process technicians and maintenance personnel must become totally familiar with the equipment and its capabilities. To minimize the risk of injury when working near robots and related equipment, personnel must comply strictly with the procedures in the manuals.

This section provides information about the safety features that are included in the paint system and also explains the way the robot interacts with other equipment in the system.

The paint system includes the following safety features:

- Most paint booths have red warning beacons that illuminate when the robots are armed and ready to paint. Your booth might have other kinds of indicators. Learn what these are.
- Some paint booths have a blue beacon that, when illuminated, indicates that the electrostatic devices are enabled. Your booth might have other kinds of indicators. Learn what these are.
- EMERGENCY STOP buttons are located on the robot controller and teach pendant. Become familiar with the locations of all E–STOP buttons.
- An intrinsically safe teach pendant is used when teaching in hazardous paint atmospheres.
- A DEADMAN switch is located on each teach pendant. When this switch is held in, and the teach pendant is on, power is applied to the robot servo system. If the engaged DEADMAN switch is released during robot operation, power is removed from the servo system, all axis brakes are applied, and the robot comes to an EMERGENCY STOP. Safety interlocks within the system might also E–STOP other robots.



WARNING

An EMERGENCY STOP will occur if the DEADMAN switch is released on a bypassed robot.

- Overtravel by robot axes is prevented by software limits. All of the major and minor axes are governed by software limits. Limit switches and hardstops also limit travel by the major axes.
- EMERGENCY STOP limit switches and photoelectric eyes might be part of your system. Limit switches, located on the entrance/exit doors of each booth, will EMERGENCY STOP all equipment in the booth if a door is opened while the system is operating in automatic or manual mode. For some systems, signals to these switches are inactive when the switch on the SCC is in teach mode.

When present, photoelectric eyes are sometimes used to monitor unauthorized intrusion through the entrance/exit silhouette openings.

- System status is monitored by computer. Severe conditions result in automatic system shutdown.

Staying Safe While Operating the Paint Robot

When you work in or near the paint booth, observe the following rules, in addition to all rules for safe operation that apply to all robot systems.



WARNING

Observe all safety rules and guidelines to avoid injury.



WARNING

Never bypass, strap, or otherwise deactivate a safety device, such as a limit switch, for any operational convenience. Deactivating a safety device is known to have resulted in serious injury and death.

- Know the work area of the entire paint station (workcell).
- Know the work envelope of the robot and hood/deck and door opening devices.
- Be aware of overlapping work envelopes of adjacent robots.
- Know where all red, mushroom-shaped EMERGENCY STOP buttons are located.

- Know the location and status of all switches, sensors, and/or control signals that might cause the robot, conveyor, and opening devices to move.
- Make sure that the work area near the robot is clean and free of water, oil, and debris. Report unsafe conditions to your supervisor.
- Become familiar with the complete task the robot will perform BEFORE starting automatic mode.
- Make sure all personnel are outside the paint booth before you turn on power to the robot servo system.
- Never enter the work envelope or paint booth before you turn off power to the robot servo system.
- Never enter the work envelope during automatic operation unless a safe area has been designated.
- Never wear watches, rings, neckties, scarves, or loose clothing that could get caught in moving machinery.
- Remove all metallic objects, such as rings, watches, and belts, before entering a booth when the electrostatic devices are enabled.
- Stay out of areas where you might get trapped between a moving robot, conveyor, or opening device and another object.
- Be aware of signals and/or operations that could result in the triggering of guns or bells.
- Be aware of all safety precautions when dispensing of paint is required.
- Follow the procedures described in this manual.

Staying Safe While Operating Paint Application Equipment

When you work with paint application equipment, observe the following rules, in addition to all rules for safe operation that apply to all robot systems.



WARNING

When working with electrostatic paint equipment, follow all national and local codes as well as all safety guidelines within your organization. Also reference the following standards: NFPA 33 Standards for Spray Application Using Flammable or Combustible Materials, and NFPA 70 National Electrical Code.

- Grounding: All electrically conductive objects in the spray area must be grounded. This includes the spray booth, robots, conveyors, workstations, part carriers, hooks, paint pressure pots, as well as solvent containers. Grounding is defined as the object or objects shall be electrically connected to ground with a resistance of not more than 1 megohms.
- High Voltage: High voltage should only be on during actual spray operations. Voltage should be off when the painting process is completed. Never leave high voltage on during a cap cleaning process.
- Avoid any accumulation of combustible vapors or coating matter.
- Follow all manufacturer recommended cleaning procedures.
- Make sure all interlocks are operational.
- No smoking.
- Post all warning signs regarding the electrostatic equipment and operation of electrostatic equipment according to NFPA 33 Standard for Spray Application Using Flammable or Combustible Material.
- Disable all air and paint pressure to bell.
- Verify that the lines are not under pressure.

Staying Safe During Maintenance

When you perform maintenance on the painter system, observe the following rules, and all other maintenance safety rules that apply to all robot installations. Only qualified, trained service or maintenance personnel should perform repair work on a robot.

- Paint robots operate in a potentially explosive environment. Use caution when working with electric tools.
- When a maintenance technician is repairing or adjusting a robot, the work area is under the control of that technician. All personnel not participating in the maintenance must stay out of the area.
- For some maintenance procedures, station a second person at the control panel within reach of the EMERGENCY STOP button. This person must understand the robot and associated potential hazards.

- Be sure all covers and inspection plates are in good repair and in place.
- Always return the robot to the “home” position before you disarm it.
- Never use machine power to aid in removing any component from the robot.
- During robot operations, be aware of the robot’s movements. Excess vibration, unusual sounds, and so forth, can alert you to potential problems.
- Whenever possible, turn off the main electrical disconnect before you clean the robot.
- When using vinyl resin observe the following:
 - Wear eye protection and protective gloves during application and removal
 - Adequate ventilation is required. Overexposure could cause drowsiness or skin and eye irritation.
 - If there is contact with the skin, wash with water.
- When using paint remover observe the following:
 - Eye protection, protective rubber gloves, boots, and apron are required during booth cleaning.
 - Adequate ventilation is required. Overexposure could cause drowsiness.
 - If there is contact with the skin or eyes, rinse with water for at least 15 minutes.

I. SAFETY

1

SAFETY PRECAUTIONS

For the safety of the operator and the system, follow all safety precautions when operating a robot and its peripheral devices installed in a work cell.

1.1 OPERATOR SAFETY

Operator safety is the primary safety consideration. Because it is very dangerous to enter the operating space of the robot during automatic operation, adequate safety precautions must be observed.

The following lists the general safety precautions. Careful consideration must be made to ensure operator safety.

- (1) Have the robot system operators attend the training courses held by FANUC.

FANUC provides various training courses. Contact our sales office for details.

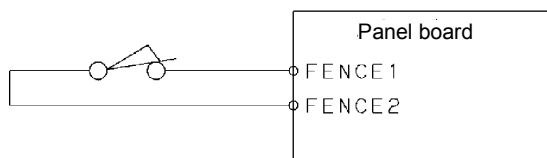
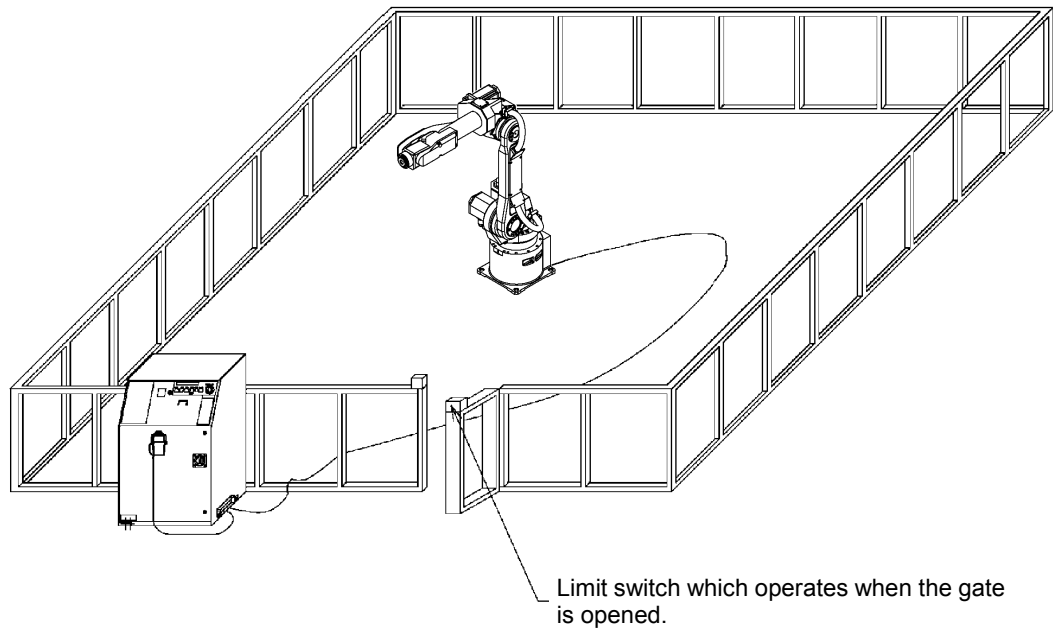
- (2) Even when the robot is stationary, it is possible that the robot is still ready to move state and is waiting for a signal. In this state, the robot is regarded as still in motion. To ensure operator safety, provide the system with an alarm to indicate visually or aurally that the robot is in motion.
- (3) Install a safety fence with a gate so that no operator can enter the work area without passing through the gate. Equip the gate with an interlock that stops the robot when the gate is opened.

The controller is designed to receive this interlock signal. When the gate is opened and this signal received, the controller stops the robot in an emergency. For connection, see Fig.1.1.

- (4) Provide the peripheral devices with appropriate grounding (Class 1, Class 2, or Class 3).
- (5) Try to install the peripheral devices outside the work area.
- (6) Draw an outline on the floor, clearly indicating the range of the robot motion, including the tools such as a hand.
- (7) Install a mat switch or photoelectric switch on the floor with an interlock to a visual or aural alarm that stops the robot when an operator enters the work area.
- (8) If necessary, install a safety lock so that no one except the operator in charge can turn on the power of the robot.

The circuit breaker installed in the controller is designed to disable anyone from turning it on when it is locked with a padlock.

- (9) When adjusting each peripheral device independently, be sure to turn off the power of the robot.



Note) Terminals FENCE1 and FENCE2 are on the PC board in the operator's panel.

Fig.1.1 (a) Safety Fence and Safety gate (For R-J3iB CONTROLLER)

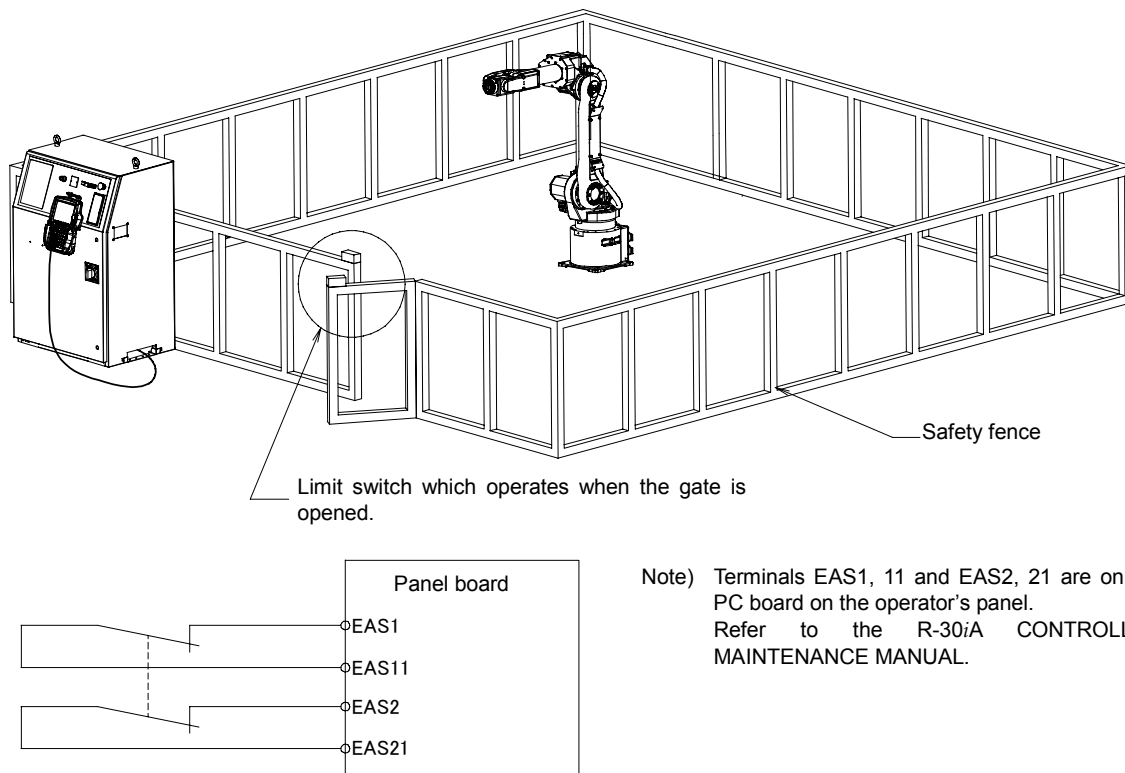


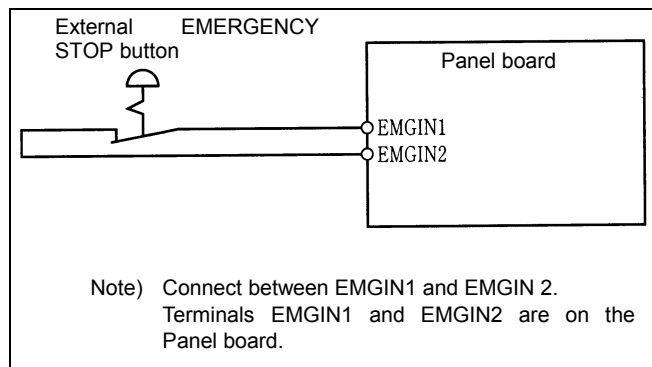
Fig.1.1(b) Safety Fence and Safety (For R-30iA CONTROLLER)

1.1.1 Operator Safety

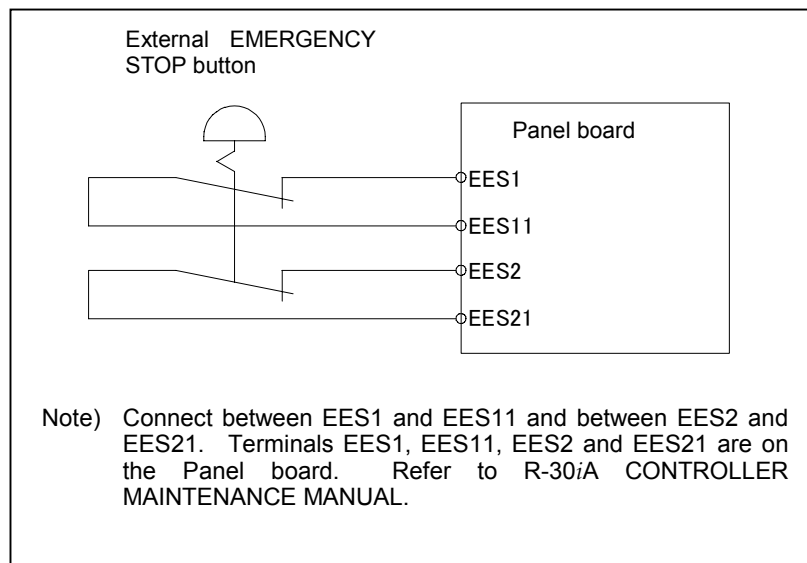
The operator is a person who operates the robot system. In this sense, a worker who operates the teach pendant is also an operator. However, this section does not apply to teach pendant operators.

- (1) If it is not necessary for the robot to operate, turn off the power of the robot controller or press the EMERGENCY STOP button, and then proceed with necessary work
- (2) Operate the robot system at a location outside the work area.
- (3) Install a safety fence with a safety gate to prevent any worker other than the operator from entering the work area unexpectedly and also to prevent the worker from entering a dangerous area.
- (4) Install an EMERGENCY STOP button within the operator's reach.

The robot controller is designed to be connected to an external EMERGENCY STOP button. With this connection, the controller stops the robot operation when the external EMERGENCY STOP button is pressed. See the diagram below for connection.



**Fig.1.1.1(a) Connection Diagram for External Emergency Stop Switch
(For R-J3iB CONTROLLER)**



**Fig.1.1.1(b) Connection Diagram for External Emergency Stop Switch
(For R-30iA CONTROLLER)**

1.1.2 Safety of the Teach Pendant Operator

While teaching the robot, it is necessary for the operator to enter the work area of the robot. It is particularly necessary to ensure the safety of the teach pendant operator.

- (1) Unless it is specifically necessary to enter the robot work area, carry out all tasks outside the area.
- (2) Before teaching the robot, check that the robot and its peripheral devices are all in the normal operating condition.
- (3) When entering the robot work area and teaching the robot, be sure to check the location and condition of the safety devices (such as the EMERGENCY STOP button and the deadman's switch on the teach pendant).

The teach pendant supplied by FANUC is provided with a teach pendant enable switch and a deadman's switch in addition to the EMERGENCY STOP button. The functions of each switch are as follows.

EMERGENCY STOP button : Pressing this button stops the robot in an emergency, irrespective to the condition of the teach pendant enable switch.

Deadman's switch : The function depends on the state of the teach pendant enable switch.

When the enable switch is on - Releasing the finger from the dead man's switch stops the robot in an emergency.

When the enable switch is off-The deadman's switch is ineffective

NOTE

The deadman's switch is provided so that the robot operation can be stopped simply by releasing finger from the teach pendant in case of emergency.

- (4) The teach pendant operator should pay careful attention so that no other workers enter the robot work area.

NOTE

In addition to the above, the teach pendant enable switch and the deadman's switch also have the following function. By pressing the deadman's switch while the enable switch is on, the emergency stop factor (normally the safety gate) connected to the controller is invalidated. In this case, it is possible for an operator to enter the fence during teach operation without pressing the EMERGENCY STOP button. In other words, the system understands that the combined operations of pressing the teach pendant enable switch and pressing the deadman's switch indicates the start of teaching.

The teach pendant operator should be well aware that the safety gate is not functional under this condition and bear full responsibility to ensure that no one enters the fence during teaching.

- (5) When entering the robot work area, the teach pendant operator should enable the teach pendant whenever he or she enters the robot work area. In particular, while the teach pendant enable switch is off, make certain that no start command is sent to the robot from any operator's panel other than the teach pendant.

The teach pendant, operator panel, and peripheral device interface send each robot start signal. However the validity of each signal changes as follows depending on the ON/OFF switch on the Teach pendant and the three modes switch on the Operator's panel and Remote condition on the software.

Operator 's panel Three modes switch	Teach pendant ON/OFF switch	Software remote condition	Teach pendant	Operator's panel	Peripheral devices
T1/T2 AUTO (Except RIA)	On	Independent	Allowed to start	Not allowed	Not allowed
AUTO	Off	Remote OFF	Not allowed	Allowed to start	Not allowed
AUTO	Off	Remote ON	Not allowed	Not allowed	Allowed to start

NOTE) When starting the system using the teach pendant in the RIA specification, the three modes switch should be T1/T2.

- (6) To start the system using the operator's box, make certain that nobody is in the robot work area and that there are no abnormal conditions in the robot work area.
- (7) When a program is completed, be sure to carry out a test run according to the procedure below.
- Run the program for at least one operation cycle in the single step mode at low speed.
 - Run the program for at least one operation cycle in the continuous operation mode at low speed.
 - Run the program for one operation cycle in the continuous operation mode at the intermediate speed and check that no abnormalities occur due to a delay in timing.
 - Run the program for one operation cycle in the continuous operation mode at the normal operating speed and check that the system operates automatically without trouble.
 - After checking the completeness of the program through the test run above, execute it in the automatic operation mode.
- (8) While operating the system in the automatic operation mode, the teach pendant operator should leave the robot work area.

1.1.3 Safety During Maintenance

For the safety of maintenance personnel, pay utmost attention to the following.

- (1) Except when specifically necessary, turn off the power of the controller while carrying out maintenance. Lock the power switch, if necessary, so that no other person can turn it on.
- (2) When disconnecting the pneumatic system, be sure to reduce the supply pressure.
- (3) Before the start of teaching, check that the robot and its peripheral devices are all in the normal operating condition.
- (4) If it is necessary to enter the robot work area for maintenance when the power is turned on, the worker should indicate that the machine is being serviced and make certain that no one starts the robot unexpectedly.
- (5) Do not operate the robot in the automatic mode while anybody is in the robot work area.
- (6) When it is necessary to maintain the robot alongside a wall or instrument, or when multiple workers are working nearby, make certain that their escape path is not obstructed.
- (7) When a tool is mounted on the robot, or when any moving device other than the robot is installed, such as belt conveyor, pay careful attention to its motion.
- (8) If necessary, have a worker who is familiar with the robot system stand beside the operator's panel and observe the work being performed. If any danger arises, the worker should be ready to press the EMERGENCY STOP button at any time.
- (9) When replacing or reinstalling components, take care to prevent foreign matter from entering the system.
- (10) When handling each unit or printed circuit board in the controller during inspection, turn off the power of the controller and also turn off the circuit breaker to protect against electric shock.
- (11) When replacing parts, be sure to use those specified by FANUC. In particular, never use fuses or other parts of non-specified ratings. They may cause a fire or result in damage to the components in the controller.

1.2 SAFETY OF THE TOOLS AND PERIPHERAL DEVICES

1.2.1 Precautions in Programming

- (1) Use a limit switch or other sensor to detect a dangerous condition and, if necessary, design the program to stop the robot when the sensor signal is received.
- (2) Design the program to stop the robot when an abnormal condition occurs in any other robots or peripheral devices, even though the robot itself is normal.
- (3) For a system in which the robot and its peripheral devices are in synchronous motion, particular care must be taken in programming so that they do not interfere with each other.
- (4) Provide a suitable interface between the robot and its peripheral devices so that the robot can detect the states of all devices in the system and can be stopped according to the states.

1.2.2 Precautions for Mechanism

- (1) Keep the component cells of the robot system clean, and operate the robot in an environment free of grease, water, and dust.
- (2) Employ a limit switch or mechanical stopper to limit the robot motion so that the robot does not come into contact with its peripheral devices or tools.

1.3 SAFETY OF THE ROBOT MECHANISM

1.3.1 Precautions in Operation

- (1) When operating the robot in the jog mode, set it at an appropriate speed so that the operator can manage the robot in any eventuality.
- (2) Before pressing the jog key, be sure you know in advance what motion the robot will perform in the jog mode.

1.3.2 Precautions in Programming

- (1) When the work areas of robots overlap, make certain that the motions of the robots do not interfere with each other.
- (2) Be sure to specify the predetermined work origin in a motion program for the robot and program the motion so that it starts from the origin and terminates at the origin. Make it possible for the operator to easily distinguish at a glance that the robot motion has terminated.

1.3.3 Precautions for Mechanisms

- (1) Keep the work area of the robot clean, and operate the robot in an environment free of grease, water, and dust.

1.4 SAFETY OF THE END EFFECTOR

1.4.1 Precautions in Programming

- (1) To control the pneumatic, hydraulic and electric actuators, carefully consider the necessary time delay after issuing each control command up to actual motion and ensure safe control.
- (3) Provide the end effector with a limit switch, and control the robot system by monitoring the state of the end effector.

1.5 SAFETY IN MAINTENANCE

- (1) Never enter the robot work area while the robot is operating. Turn off the power before entering the robot work area for inspection and maintenance.
- (2) If it is necessary to enter the robot work area with the power turned on, first press the EMERGENCY STOP button on the operator's box.
- (3) When replacing or reinstalling components, take care to prevent foreign matter from entering the system. When replacing the parts in the pneumatic system, be sure to reduce the pressure in the piping to zero by turning the pressure control on the air regulator.
- (4) When handling each unit or printed circuit board in the controller during inspection, turn off the power of the controller and turn off the circuit breaker to protect against electric shock.
- (5) When replacing parts, be sure to use those specified by FANUC. In particular, never use fuses or other parts of non-specified ratings. They may cause a fire or result in damage to the components in the controller.
- (6) Before restarting the robot, be sure to check that no one is in the robot work area and that the robot and its peripheral devices are all in the normal operating state.

1.6 WARNING LABEL

(1) Greasing and degreasing label

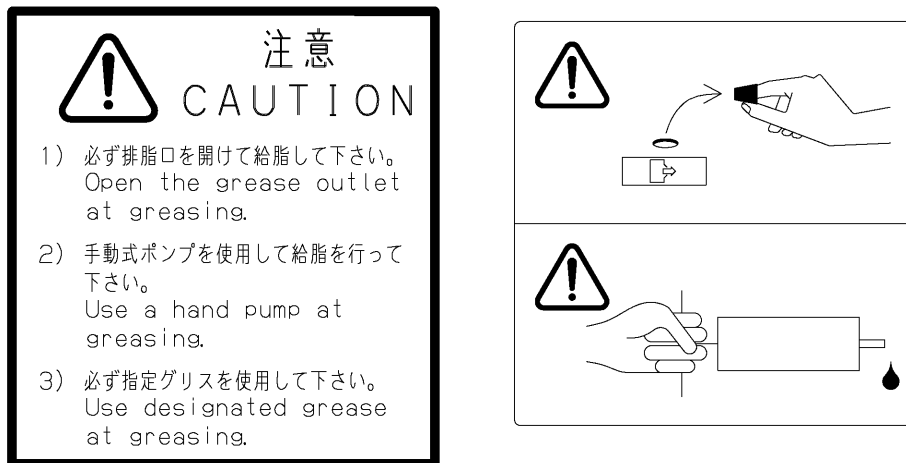


Fig. 1.6 (a) Greasing and Degreasing Label

Description

When greasing and degreasing, observe the instructions indicated on this label.

- 1) When greasing, be sure to keep the grease outlet open.
- 2) Use a manual pump to grease.
- 3) Be sure to use a specified grease.

(2) Step-on prohibitive label

**Fig. 1.6 (b) Step-on Prohibitive Label****Description**

Do not step on or climb the robot or controller as it may adversely affect the robot or controller and you may get hurt if you lose your footing as well.

(3) High-temperature warning label

**Fig. 1.6 (c) High-temperature warning label****Description**

Be cautious about a section where this label is affixed, as the section generates heat. If you have to inevitably touch such a section when it is hot, use a protective provision such as heat-resistant gloves.

TABLE OF CONTENTS

SAFETY

1	SAFETY PRECAUTIONS.....	s-3
1.1	OPERATOR SAFETY	s-4
1.1.1	Operator Safety	s-7
1.1.2	Safety of the Teach Pendant Operator	s-8
1.1.3	Safety During Maintenance.....	s-11
1.2	SAFETY OF THE TOOLS AND PERIPHERAL DEVICES	s-12
1.2.1	Precautions in Programming	s-12
1.2.2	Precautions for Mechanism.....	s-12
1.3	SAFETY OF THE ROBOT MECHANISM.....	s-13
1.3.1	Precautions in Operation.....	s-13
1.3.2	Precautions in Programming	s-13
1.3.3	Precautions for Mechanisms	s-13
1.4	SAFETY OF THE END EFFECTOR.....	s-14
1.4.1	Precautions in Programming	s-14
1.5	SAFETY IN MAINTENANCE	s-14
1.6	WARNING LABEL.....	s-15

OPERATION

1	PREFACE	1
2	ENVIRONMENT.....	2
2.1	REQUIRED SOFTWARE	3
2.2	RESTRICTION ON USE WITH OTHER OPTIONAL FUNCTIONS.....	4
2.3	NETWORK.....	5
3	CAUTIONS (BE SURE TO READ THE FOLLOWING:).....	6
3.1	CABLING AND CONNECTION	7
4	SETTINGS ON THE ROBOT SIDE	8
4.1	NETWORK-RELATED SETTINGS	9
4.1.1	Setting the Host Names, Internet (IP) Addresses, and Subnet Mask	9
4.1.2	Setting Full-Duplex Mode (on the Robot Side)	10
4.1.3	Setting Full-Duplex Mode (on the Hub Side).....	10
4.2	CHECKING NETWORK-RELATED SETTINGS.....	11
5	SETTING UP CIMPLICITY HMI	12

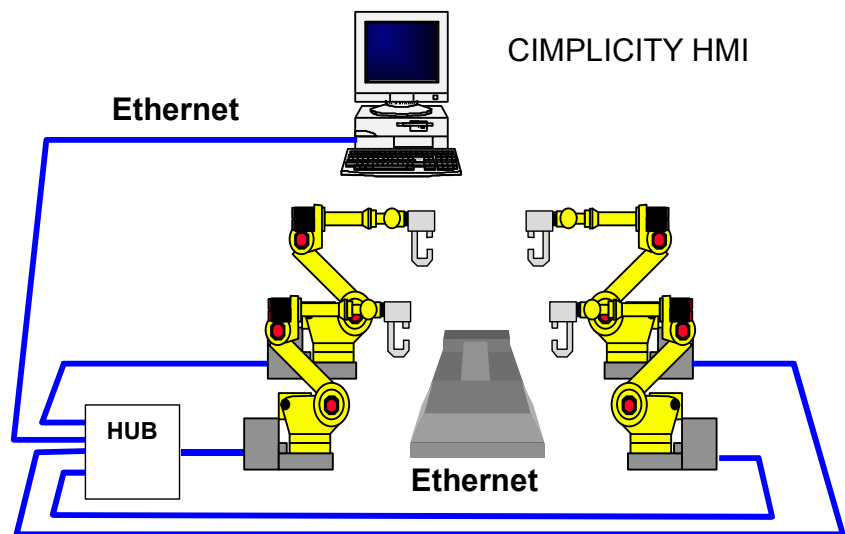
5.1	CREATING A NEW PROJECT	13
5.2	REGISTERING A PORT	15
5.3	REGISTERING DEVICES	17
5.4	REGISTERING POINTS	19
6	ADDRESS ASSIGNMENT TO POINTS	21
6.1	READING AND WRITING I/O SIGNALS (%I, %Q, %M, %AI, %AQ)	22
6.2	READING FROM AND WRITING TO REGISTERS (%R).....	24
6.3	READING FROM AND WRITING TO POSITION REGISTERS (%R).....	27
6.4	READING AND WRITING THE CURRENT POSITION (%R).....	34
6.5	READING ALARM HISTORY (%R).....	36
6.6	READING THE PROGRAM EXECUTION STATUS (%R)	39
6.7	READING FROM AND WRITING INTO SYSTEM VARIABLES (%R)	41
6.8	READING AND WRITING THE COMMENT OF REGISTERS, POSITION REGISTERS, AND I/O (%R)	44
6.9	READING AND WRITING THE VALUE AND SIM STATUS OF I/O (%R) ..	46
6.10	SETTING \$SNPX_ASG FROM CIMPLICITY (%G).....	48
6.11	NOTES AND TIPS ON USAGE.....	50
6.11.1	\$SNPX_ASG Is Set, but Data That Should Have Been Assigned Cannot Be Read/Written.	50
6.11.2	Improving Communication Efficiency.....	51
6.11.3	Version Of Communication Function With CIMPLICITY	51
6.11.4	Multi Connection And Multiplex \$SNPX_ASG.....	52
7	APPENDIX.....	53
7.1	SAMPLE PROJECT OF "HMI FOR ROBOT"	54
7.1.1	Sample Project Overview.....	54
7.1.2	Registered Devices.....	54
7.1.3	Registered Points.....	54
7.1.4	\$SNPX_ASG Settings (em_init.bcl).....	57
7.1.5	Running ROBOGUIDE To Monitor The Robot Motion (TOP.CIM).....	57
7.1.6	Getting The Alarm Information (ALARM1.CIM).....	61

II . OPERATION

1

PREFACE

CIMPLICITY HMI allows the user at a personal computer on which the CIMPLICITY HMI software is installed to remotely monitor, record, analyze, and control data of robot control units installed in a factory. To do this, CIMPLICITY uses an Ethernet communication feature that is provided as a standard feature of the robot control unit.



This document handles both CIMPLICITY version 4.01 with service pack 10 or later, and version 5.5.

And the pictures of CIMPLICITY screen in this document are captured on CIMPLICITY version 5.5.

The difference between CIMPLICITY version 4.01 and 5.5 is that, CIMPLICITY version 4.01 can connect to robot control unit with the device model type both GE Fanuc Series 90-30 and 90-70, but CIMPLICITY version 5.5 can connect with the device model type only GE Fanuc Series 90-30.

If you update CIMPLICITY version 4.01 to 5.5 and use the old project, check the model type of all devices registered in the project.

2

ENVIRONMENT

2.1 REQUIRED SOFTWARE

CIMPLICITY HMI does not require any special options.

R-J3 and R-J3iB do not require any special option software, either. R-30iA(R-J3iC) requires “HMI Device (SNPX)” option software if “FRA Params” option software is installed.

The applicable series and edition of the R-J3 system software is 7D70, A1 or later. And R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC) support the new functions such as Multiplex Communicating.

	Standard functions	New functions	Required software option
R-J3	7D70, A1 or later	Not supported	None
R-J3iB	All versions	7D80, 45 or later 7D81, 09 or later	None
R-30iA(R-J3iC) Standard	All versions	All versions	None
R-30iA(R-J3iC) FRA Params	All versions	All versions	HMI Device (SNPX)

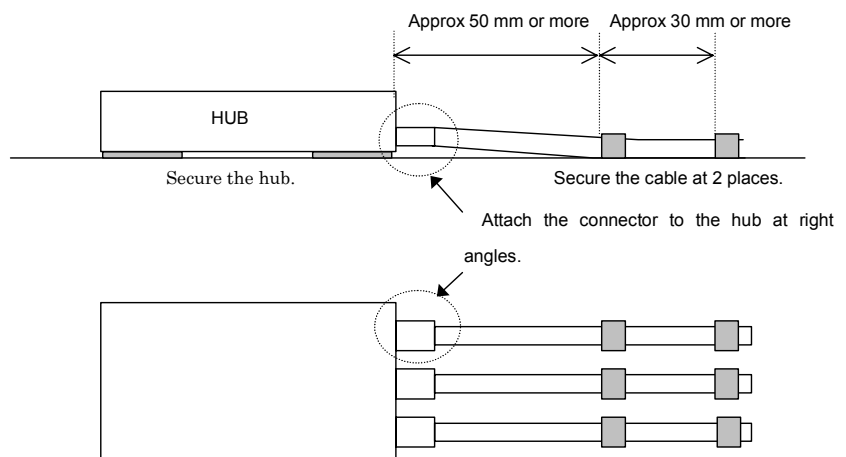
2.2 RESTRICTION ON USE WITH OTHER OPTIONAL FUNCTIONS

CIMPLICITY HMI cannot be used together with the robot link function.

2.3 NETWORK

- For the Ethernet Cable, use noise-resistant twisted-pair cable for 10BASE-T.
- For the hub (line concentrator) required for implementing a network, use a switching hub where possible. FANUC recommends a switching hub of which modular connector has no metal shield. This is prevent noise from entering the network from the backbone.
- Perform cabling so that the hub and Ethernet cables make good contact.

For example, install the hub and cables as follows:



- The installation environment of the hub must conform the specification defined by the manufacturer of the hub. For example, the hub must be installed in a vibration-free, dust-proof place.
- Sufficient measures should be taken to prevent noise.



CAUTION

The customer is responsible for purchasing cables and hubs. Make Provisions to protect yourself in the event of a hub failure, by purchasing spare parts.

3

CAUTIONS (BE SURE TO READ THE FOLLOWING:)

The following provides basic items the customer should bear in mind when using this function. The customer's system must at least satisfy these items. Accordingly, the customer may need to take additional safety measures according to the customer's safety standards for the system.

3.1 CABLING AND CONNECTION

- Run the Ethernet cable so that an operator can move safely and freely about with no tripping hazard.
- Run the Ethernet cable so as to avoid subjecting it to noise. Keep the cable clear of any noise source.
- Run the hub power cable clear of any traffic so as to avoid a tripping hazard and to avoid inadvertently disconnecting the power cable.
- Consider the location of cabling and installation of the switching hub so that the hub and the cable connectors make good contact.

The following provides very important items. Be sure to observe these items.

- After connecting one end of the Ethernet cable to the hub, and attaching the other end of the cable to the Ethernet cable jack on the main board of the robot controller, the LED on the printed circuit board located on the far side of the cable jack lights in orange. The hub and robot controller must both have been turned on. If the LED does not light, the main board may have a problem. Contact the FANUC Service Center.
- On the front of the hub, there is an LED indicator that provides the status of communications. If communication cannot be performed normally, the LED must be checked. For this reason, the hub must be installed so that the user can easily check the LED.

4

SETTINGS ON THE ROBOT SIDE

Before this function can be used, settings related to the Ethernet network must be made on the robot side.

4.1 NETWORK-RELATED SETTINGS

This function uses Ethernet. First, make settings for using Ethernet.

4.1.1 Setting the Host Names, Internet (IP) Addresses, and Subnet Mask

Display the protocol setup screen of the host communication setup screen.

On this setup screen, set the following items:

- Node name of the controller
- Node name of the router (Even when there is no router, be sure to enter a dummy node name.)
- Subnet mask
- Node names and IP addresses of all controllers (including this controller) that involve robot link communication

For example, the following settings are made:

```

SETUP Host Comm          JOINT 10 %
TCP/IP                   6/31
Node name:               ROBOT
Router name:             ROUTER
Board address: 00:E0:E4:FB:BA:DD
Subnet Mask:             255.255.255.0

Host Name (LOCAL) Internet Address
1  ROBOT                192.168.0.1
2  ROUTER               192.168.0.99
3  *****             *****
4  *****             *****

[ TYPE ]                LIST
  
```

R-J3

```

SETUP Host Comm          JOINT 10 %
TCP/IP                   5/32
Robot name:              ROBOT
Robot IP addr:           192.168.0.1
Router IP addr:          192.168.0.99
Subnet Mask:             255.255.255.0
Board address: 00:E0:E4:FB:BA:DD

Host Name (LOCAL) Internet Address
1  *****             *****
2  *****             *****
3  *****             *****

[ TYPE ]                LIST  PING  HELP
  
```

R-J3iB

```

SETUP Host Comm          JOINT 10 %
TCP/IP                   5/40
Robot name:              ROBOT
Port#1 IP addr:          192.168.0.1
Subnet Mask:             255.255.255.0
Board address: 00:E0:E4:FB:BA:DD
Router IP addr:          192.168.0.99

Host Name (LOCAL) Internet Address
1  *****             *****
2  *****             *****
3  *****             *****

[ TYPE ]                PORT  PING  HELP >
  
```

R-30iA(R-J3iC)

Set system-specific node names and Internet (IP) addresses. Any names may be used, but they should be intelligible.

**CAUTION**

The settings on this screen are made by entering characters. If a space exists in any character string (be aware of a space at the beginning of a character string), correct communication is impossible. In such a case, delete the entire line, and reenter the correct character string.

4.1.2 Setting Full-Duplex Mode (on the Robot Side)

When using a switching hub, set the robot in full-duplex mode. When using an ordinary hub, there is no need to change the setting. Set system variable \$ENETMODE.\$FULL_DUPLEX to TRUE.

4.1.3 Setting Full-Duplex Mode (on the Hub Side)

When the hub is equipped with a DIP switch or other means to switch between full-duplex and half-duplex modes, set the full-duplex mode, set the autonegotiate mode. (Or set the full-duplex mode if the autonegotiate mode cannot be set.)

4.2 CHECKING NETWORK-RELATED SETTINGS

When you have completed making the settings as described earlier, check the following items to confirm that the settings are correct:

1. Is an Ethernet cable attached between the hub and robot controller?
-> If not, attach it now.
2. When the power to the hub and robot controller is on, is the orange LED on? This LED is located on the printed circuit board on the far side of the Ethernet cable jack on the main board of the robot controller.
-> If the LED is not on, reconnect the Ethernet cable and turn on the power to the hub and controller again. If the LED is still off, the main board may be faulty. Contact the FANUC Service Center.

5

SETTING UP CIMPLICITY HMI

Create a project that communicates with robots on CIMPLICITY HMI.

The Process is as follows:

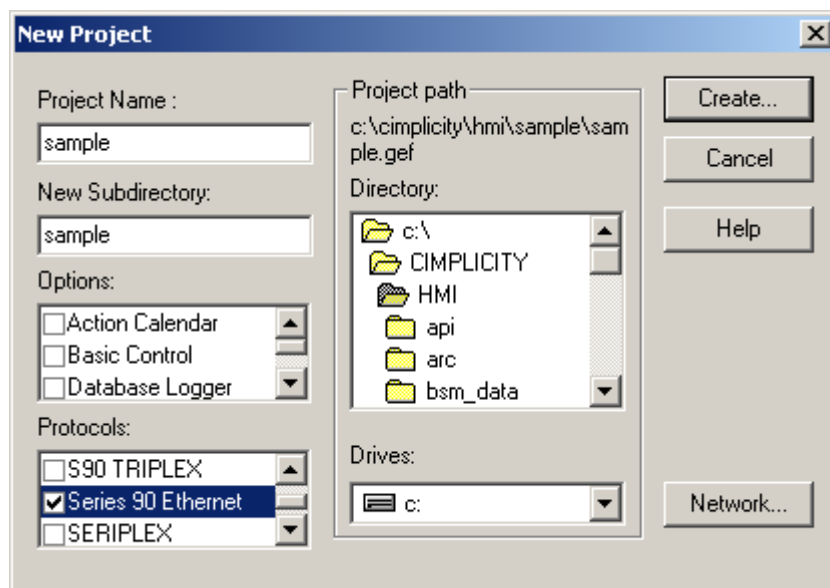
1. Create a new project.
2. Register the port, devices, and points.

First, start the CIMPLICITY HMI Workbench program.

5.1 CREATING A NEW PROJECT

Before CIMPLICITY HMI can be connected to robot controller over an Ethernet, Series 90 Ethernet must be specified as the protocol when a new project is created.

1. Open the File menu, and select New, then Project.
The New Project dialog box appears.

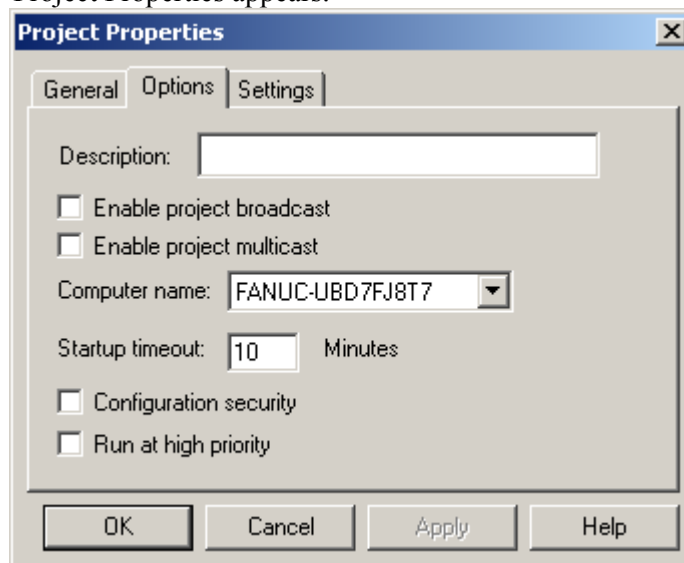


Set the following:

Project Name and New Subdirectory : Arbitray

Series 90 Ethernet in Protocols : Check this item.

2. Click Create.
Project Properties appears.



3. Click OK.

The CIMPLICITY Project Wizard dialog box appears.



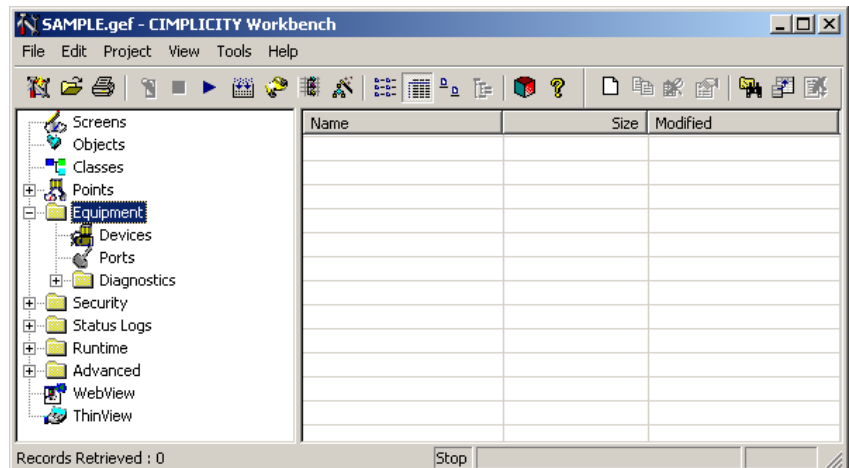
4. Click Exit.

Here, settings will be made individually without using the wizard. You can now close the dialog box.

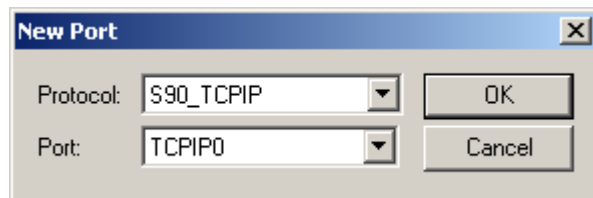
5.2 REGISTERING A PORT

Next, you will register a port.

1. Double-click the Equipment folder on the tree.
The Equipment folder opens.

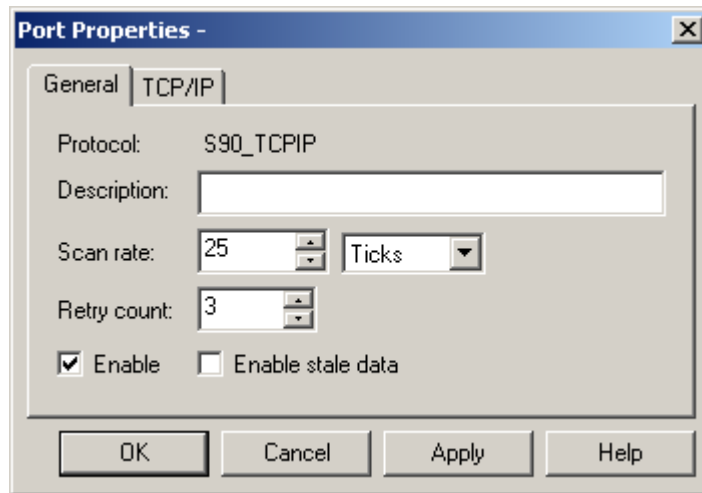


2. Double-click Ports.
The New Port dialog box appears.



Set the following:
Protocol : S90_TCPIP
Port : Arbitray (TCPIP0 in this case)

3. Click OK.
The Port Properties dialog box appears.



Set the following:

Scan Rate : 25 Ticks (250ms)

This sets the period of intervals that CIMPLICITY HMI acquires data from R-J3.

The period is set so that CIMPLICITY acquires data every 250ms.

4. Click OK.

A port for connecting robot controller is then registered under the name MASTER_TCPIP0.

5.3 REGISTERING DEVICES

Next, you will register devices.

1. Double-click Devices in the Equipment folder.
The New Device dialog box appears.

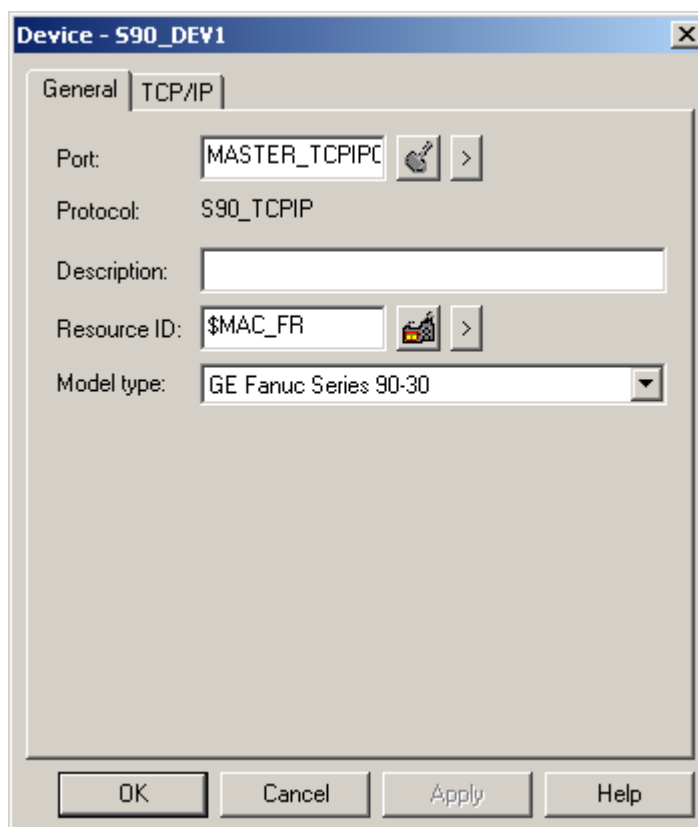


Set the following:

Device : Arbitrary (example : S90_DEV1)

Port : MASTER_TCPIP0 (the port registered in the previous step)

2. Click OK.
The Device Properties dialog box appears.



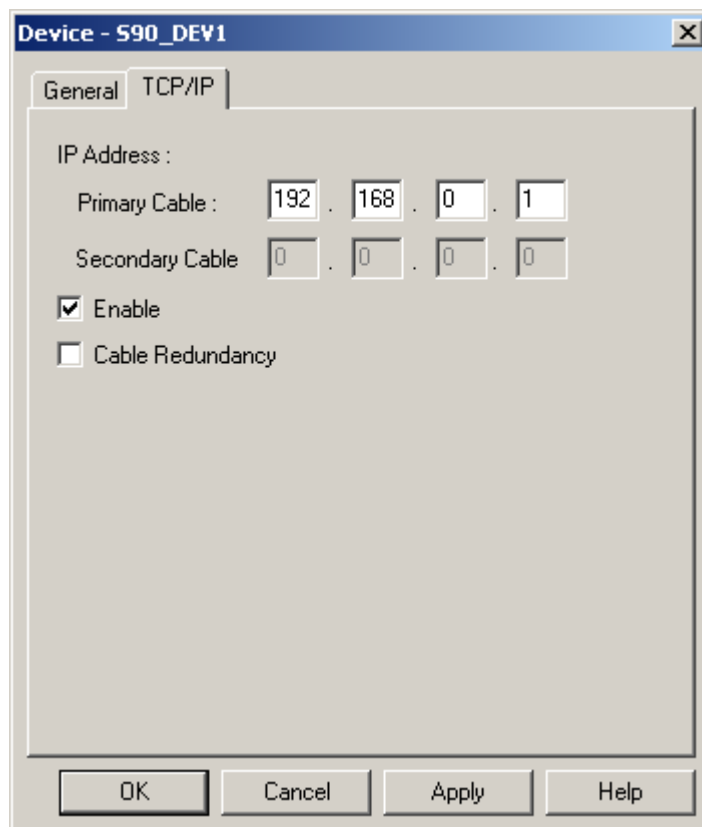
Set the following:

Model Type : GE Fanuc Series 90-30

**CAUTION**

CIMPLICITY version 4.01 can connect to robot controller control unit with the model type both GE Fanuc Series 90-30 and 90-70, but CIMPLICITY version 5.5 can connect with the model type GE-FANUC Series 90-30 only.

In the case that you update CIMPLICITY to version 5.5 or later, your project that uses GE-FANUC Series 90-70 may not work. Please change the device type from GE-Fanuc Series 90-70 to 90-30.



Select the TCP/IP tab, and set the IP address of R-J3 set on the host communication setup screen.

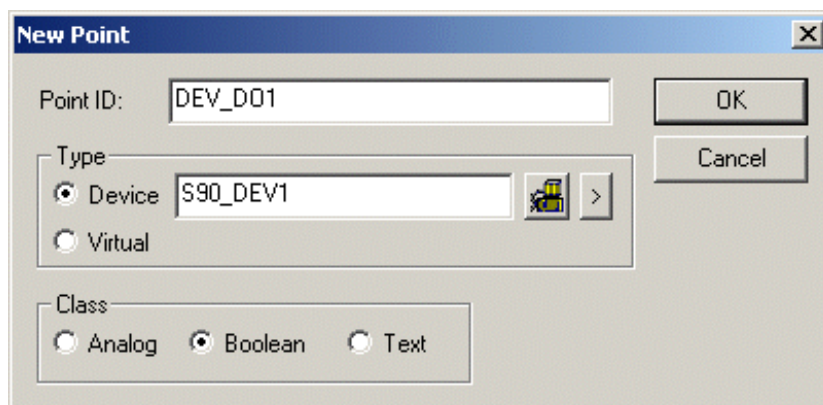
Check Enable.

Clicking OK registers S90_DEV1 as Device.

5.4 REGISTERING POINTS

Next, you will register points. (Here, you will register DO[1])

1. Double-click Points of the tree.
The New Points dialog box appears.



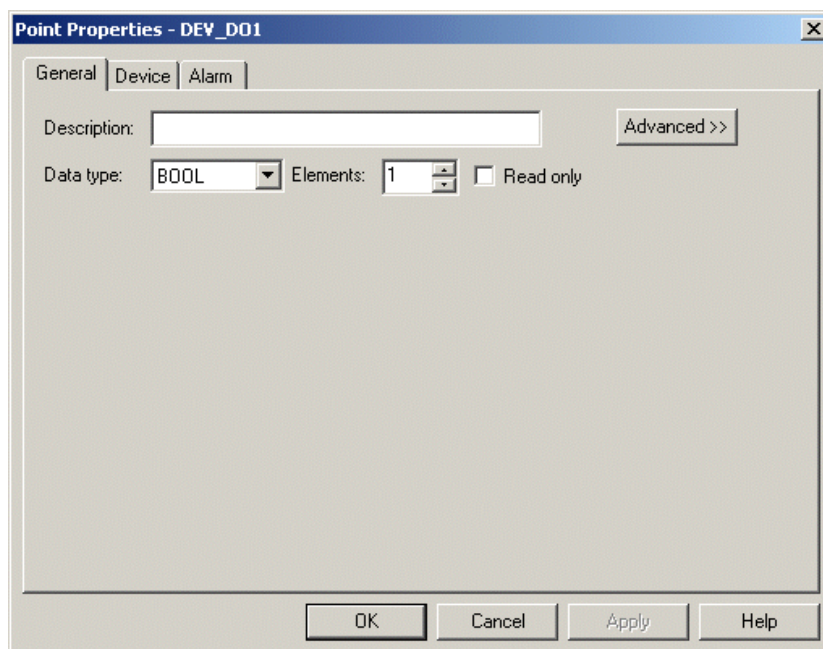
Set the following:

Point ID : Arbitrary (here, DEV_DO1 is set.)

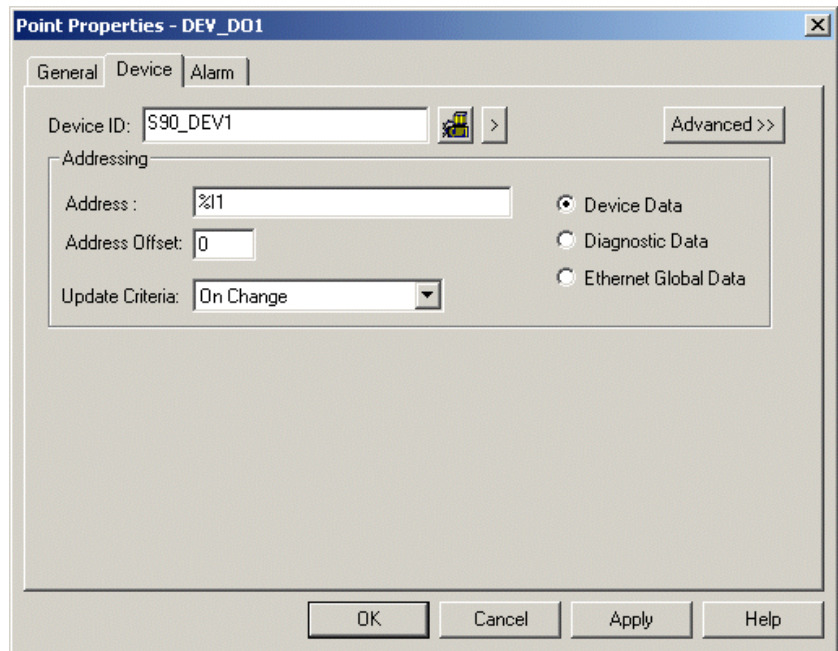
Type : Select Device and enter S90_DEV1 (this is the device that you registered previously.)

Class : Select the class according to the data to be referenced. (here, select Boolean.)

2. Click OK.
The Point Properties dialog box appears.



3. Click the Device tab.



In the Address field of Addressing, enter the address of the data you want to reference. (Here, enter %I1 for DO[1].)
For details on how to specify an address, see Chapter 6.

6

ADDRESS ASSIGNMENT TO POINTS

Many types of robot controller data are associated with addresses of the Series 90 PLC manufactured by GE Fanuc (called PLC addresses hereinafter). Therefore, CIMPLICITY can read and write robot controller data by reading and writing PLC address.

6.1 READING AND WRITING I/O SIGNALS (%I, %Q, %M, %AI, %AQ)

CIMPLICITY can read and write DI[1] of robot controller as PLC address %Q1. Robot controller I/O signals correspond to PLC addresses on a one-to-one basis as listed below.

Robot controller I/O signal		PLC address	Example
Digital Input	DI[x]	%Qx	DI[1] \leftrightarrow %Q1
Digital Output	DO[x]	%Ix	DO[1] \leftrightarrow %I1
Robot Input	RI[x]	%Q(5000+x)	RI[1] \leftrightarrow %Q5001
Robot Output	RO[x]	%I(5000+x)	RO[1] \leftrightarrow %I5001
UOP Input	UI[x]	%Q(6000+x)	UI[1] \leftrightarrow %Q6001
UOP Output	UO[x]	%I(6000+x)	UO[1] \leftrightarrow %I6001
SOP Input	SI[x]	%Q(7000+x)	SI[0] \leftrightarrow %Q7001
SOP Output	SO[x]	%I(7000+x)	SO[0] \leftrightarrow %I7001
Weld Input	WI[x]	%Q(8000+x)	WI[0] \leftrightarrow %Q8001
Weld Output	WO[x]	%I(8000+x)	WO[0] \leftrightarrow %Q8001
Wire Stick Input	WSI[x]	%Q(8400+x)	WSI[0] \leftrightarrow %Q8400
Wire Stick Output	WSO[x]	%I(8400+x)	WSO[0] \leftrightarrow %Q8401
Group Input	GI[x]	%Aqx	GI[1] \leftrightarrow %AQ1
Group Output	GO[x]	%Aix	GO[1] \leftrightarrow %AI1
Analog Input	AI[x]	%AQ(1000+x)	AI[1] \leftrightarrow %AQ1001
Analog Output	AO[x]	%AI(1000+x)	AO[1] \leftrightarrow %AI1001
PMC keep relay DO[x] (x : 10001 – 10144) Ka.b		%Ix %I((a*8)+b+10001)	DO[10001] \leftrightarrow %I10001 K2.5 \leftrightarrow %I10022
PMC internal relay DO[x] (x : 11001 – 23000) Ra.b		%M(x-11000) %M((a*8)+b+1)	DO[11001] \leftrightarrow %M1 R2.5 \leftrightarrow %M22
PMCdata table GO[x] (x : 10001 – 12000) D(a*2), D((a*2)+1)		%AI(x-6000) %AI(a+4001)	GO[10001] \leftrightarrow %AI4001 D4, D5 \leftrightarrow %AI4003



CAUTION

In the PLC address, %I and %AI are input signals, and %Q and %AQ are output signals. When robot controller data is read and written, however, the robot controller input signals correspond to %Q and %AQ, and the robot controller output signals correspond to %I and %AI. Note that, therefore, the meanings of the robot controller signals are opposite to those of the PLC signals.

**CAUTION**

When CIMPPLICITY writes data to an input signal such as DI and AI, the value changes momentarily, then it is soon restored to its actual input value. The momentary change in value, however, may cause an unpredictable errors. Therefore, avoid write into input signals such as DI and AI.

**CAUTION**

The reading from and writing to Weld Signal Input/Output and Wire Stick Input/Output functions are supported by R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC).

6.2 READING FROM AND WRITING TO REGISTERS (%R)

The standard correspondence between robot controller data and PLC addresses is as follows:

Robot controller data	PLC address	Example
Register	%Rx	R[1] ⇔ %R1



CAUTION

USE the value in the register as a 16-bit signed integer. Any fractional part of the register value is rounded off. Unless the register value is within the range from –32768 to 32767, the register value cannot be read or written correctly.

The correspondence between registers and PLC addresses is defined with system variable \$SNPX_ASG. The standard \$SNPX_ASG settings are provided as listed below. As a result, registers correspond to PLC addresses as shown above. When the \$SNPX_ASG settings have been modified, the above correspondence is not obtained. Therefore, before reading from or writing into registers through CIMPLICITY, check that \$SNPX_ASG is set as follows:

System variable	Value
\$SNPX_ASG[1].\$ADDRESS	1
\$SNPX_ASG[1].\$SIZE	10000
\$SNPX_ASG[1].\$VAR_NAME	R[1]@1.1
\$SNPX_ASG[1].\$MULTIPLY	1

\$SNPX_ASG consists of 80 arrays \$SNPX_ASG[1] to [80], each of which has four elements including \$ADDRESS, \$SIZE, \$VAR_NAME, and \$MULTIPLY. By modifying these settings, many types of data can be assigned to %R. When CIMPLICITY reads from and writes into position registers and system variables, which will be described later, \$SNPX_ASG is also used.

When assigning registers, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation
\$ADDRESS	Meaning: Start address of %R to be assigned. Range: 1 to 16384
\$SIZE	Meaning: Number of %R's to be assigned. Two %R's are required per register. Set this element according to the number of registers you want to read from or write to. (Addint @ to \$VAR_NAME changes the number of %R's per register.) Range: 1 to 16384

\$VAR_NAME	<p>Meaning: Character string indicating the data to be assigned. When assigning a register, specify, for example, R[1]. The number in brackets is a register number.</p> <p>Consecutive registers such as R[2] to R[5] can be assigned at one time. In this case, the number of the registers is four, so set 8 in \$SIZE and set R[2] in \$VAR_NAME. Here, the index 2 indicates that registers are assigned sequentially from R[2].</p> <p>The number of %R's per register can be set to 1 by adding @1.1 to the end of the character string. In this case, 16-bit data is used. Example: R[1]@1.1 R[1] indicates that registers are assigned sequentially from R[1]. @1.1 indicates that data is used as 16-bit data.</p>
\$MULTIPLY	<p>Meaning: Multiplier The register value is multiplied by the value set in \$MULTIPLY, then the multiplication result is read or written.</p> <p>When \$MULTIPLY is set to 0, it has a special meaning. %R can be read and written as 32-bit real type data.</p> <p>When \$MULTIPLY is set to non-zero value, 32-bit signed integer data is used, with its fractional part rounded off.</p> <p>Range: 0.0001 to 10000, 0</p> <p>Example: Suppose that the register value is 123.45: When \$MULTIPLY is 1, 123 is read. When \$MULTIPLY is 10, 1235 is read. When \$MULTIPLY is 0.1, 12 is read. When \$MULTIPLY is 0, 123.45(real number) is read.</p>

An example of setting \$SNPX_ASG is given below:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	2	R[1]@1.1	1
\$SNPX_ASG[2]	3	4	R[1]	100
\$SNPX_ASG[3]	7	4	R[2]	0.1
\$SNPX_ASG[4]	11	2	R[1]	0

Then, the %R-to-register correspondence is follows:

PLC address	Robot controller data that can be read and written
%R1	16-bit signed integer of R[1]
%R2	16-bit signed integer of R[2]
%R3-4	32-bit signed integer obtained by multiplying R[1] by 100
%R5-6	32-bit signed integer obtained by multiplying R[2] by 100
%R7-8	32-bit signed integer obtained by dividing R[2] by 10
%R9-10	32-bit signed integer obtained by dividing R[3] by 10
%R11-12	32-bit real number of R[1]

\$SNPX_ASG[1] indicates that two %R's, %R1 and %R2, are assigned to registers sequentially from R[1] in the format of 16-bit signed integers multiplied by one. Since \$R is 16-bit data, the number

of %R's per register is one. Therefore, %R1 corresponds to the 16-bit signed integer of R[1], and %R2 corresponds to the 16-bit signed integer of R[2].

\$SNPX_ASG[2] indicates that four %R's, %R3 to %R6, are assigned to registers sequentially from R[1] in the format of 32-bit signed integers multiplied by 100. Each register uses two %R's. Therefore, %R3 and %R4 are regarded as 32-bit signed integers obtained by multiplying R[1] by 100, and %R5 and %R6 are regarded as 32-bit signed integers obtained by multiplying R[2] by 100.

\$SNPX_ASG[3] indicates that four %R's, %R7 to %R10, are assigned to registers sequentially from R[2] in the format of 32-bit signed integers divided by 10. Therefore, %R7 and %R8 are regarded as 32-bit signed integers obtained by dividing R[2] by 10, and %R9 and %R10 are regarded as 32-bit signed integers obtained by dividing R[3] by 10.

\$SNPX_ASG[4] indicates that two %R's, %R11 and %R12, are assigned to registers sequentially from R[1] in real number format. Therefore, %R11 and %R12 are regarded as real numbers of R[1].

6.3 READING FROM AND WRITING TO POSITION REGISTERS (%R)

As with registers, position registers are assigned to %R using \$SNPX_ASG to allow CIMPPLICITY to read from and write to position registers. Unlike registers, position registers are not assigned as standard. So \$SNPX_ASG must always be set.

When assigning position registers, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384
\$SIZE	Meaning: Number of %R's to be assigned Fifty %R's are required per position register. (The number of %R's used per position register can be changed by specifying @ in \$VAR_NAME) Range: 1 to 16384
\$VAR_NAME	Meaning: Character string indicating the data to be assigned When assigning a position register, specify, for example, PR[1]. The number in brackets is a position register number. Consecutive position registers such as PR[2] to PR[5] can be assigned at one time. In this case, the number of the position registers is four, so set 200 in \$SIZE and set PR[2] in \$VAR_NAME. Here, the index 2 indicates that registers are assigned sequentially from PR[2]. In a multi-group system, PR[1] indicates group 1 to PR[1]. To specify data of group 2, specify the group before the index, such as PR[G2:1]. When @ is specified after the character string, it is possible to assign, for example, J1 through J6 only. This will be explained in detail later.

\$MULTIPLY	<p>Meaning: Multiplier</p> <p>Only the elements such as X, Y, Z, and J1 that have a real number are affected by \$MULTIPLY. For the affected elements, see the table given below.</p> <p>The value of each element of the position register is multiplied by the value set in \$MULTIPLY, then the multiplication result is read or written.</p> <p>When \$MULTIPLY is set to 0, it has a special meaning. %R can be read and written as 32-bit real type data.</p> <p>When \$MULTIPLY is set to a non-zero value, 32-bit signed integer data is read and written with its fractional part rounded off.</p> <p>Range: 0.0001 to 10000, 0</p> <p>Example: Suppose that a position register element value is 123.45: When \$MULTIPLY is 1, 123 is read. When \$MULTIPLY is 10, 1235 is read. When \$MULTIPLY is 0.1, 12 is read. When \$MULTIPLY is 0, 123.45 (real number) is read.</p>
------------	--

Each position registers uses 50 %R's. The contents of the 50 %R's are listed below:

%R address	Explanation		Influence by \$MULTIPLY
Cartesian coordinate data			
1-2	X	32-bit signed integer or real number (mm)	Influenced
3-4	Y	32-bit signed integer or real number (mm)	Influenced
5-6	Z	32-bit signed integer or real number (mm)	Influenced
7-8	W	32-bit signed integer or real number (deg)	Influenced
9-10	P	32-bit signed integer or real number (deg)	Influenced
11-12	R	32-bit signed integer or real number (deg)	Influenced
13-14	E1	32-bit signed integer or real number (mm, deg)	Influenced
15-16	E2	32-bit signed integer or real number (mm, deg)	Influenced
17-18	E3	32-bit signed integer or real number (mm, deg)	Influenced
19	FLIP	16-bit signed integer (1:Flip, 0:Non flip)	Not influenced
20	LEFT	16-bit signed integer (1:Left, 0:Right)	Not influenced
21	UP	16-bit signed integer (1:Up, 0:Down)	Not influenced
22	FRONT	16-bit signed integer (1:Front, 0:Back)	Not influenced
23	TURN4	16-bit signed integer (-128~127)	Not influenced
24	TURN5	16-bit signed integer (-128~127)	Not influenced
25	TURN6	16-bit signed integer (-128~127)	Not influenced
26	VALIDC	16-bit signed integer (*1)	Not influenced
Joint data			
27-28	J1	32-bit signed integer or real number (mm, deg)	Influenced
29-30	J2	32-bit signed integer or real number (mm, deg)	Influenced
31-32	J3	32-bit signed integer or real number (mm, deg)	Influenced
33-34	J4	32-bit signed integer or real number (mm, deg)	Influenced
35-36	J5	32-bit signed integer or real number (mm, deg)	Influenced
37-38	J6	32-bit signed integer or real number (mm, deg)	Influenced
39-40	J7	32-bit signed integer or real number (mm, deg)	Influenced

41-42	J8	32-bit signed integer or real number (mm, deg)	Influenced
43-44	J9	32-bit signed integer or real number (mm, deg)	Influenced
45	VALIDJ	16-bit signed integer (*2)	Not influenced
Coordinate system number			
46	UF	16-bit signed integer (0 to 15) (*3)	Not influenced
47	UT	16-bit signed integer (0 to 15) (*4)	Not influenced
48-50	Reserve	Not used	Not influenced

- *1 VALIDC indicates whether the position register has valid Cartesian coordinates. VALIDC is set to 0 in one of the following cases. In other cases, it is set to 1.
- There is data that is not taught (“*****” is indicated on the teach pendant).
 - The position register is in the joint format, and cannot be converted to the Cartesian coordinate format.
- Writing a value into VALIDC changes the format of the position register to the Cartesian coordinate format. You may write any value into VALIDC.
- *2 VALIDJ indicates whether the position register has valid values in joint format. VALIDJ is set to 0 in one of the following cases. In other cases, it is set to 1.
- There is data that is not taught (“*****” is indicated on the teach pendant).
 - The position register is in the Cartesian coordinate format, and cannot be converted to the joint format.
- Writing a value into VALIDJ changes the format of the position register to the joint format. You may write any value into VALIDJ.
- *3 UF indicates the number of the user coordinate system used.
- If UF is 0, the world coordinate system is used.
- If UF is 15, the user coordinate system currently selected is used.
- Normally, UF of the position register is set to 15 (F is indicated on the teach pendant).
- You cannot change this value from the teach pendant. Note that once you have changed this value with CIMPPLICITY, you cannot restore the original value without using CIMPPLICITY.
- *4 UT indicates the number of the tool coordinate system used.
- If UT is 0, the mechanical interface coordinate system is used.
- If UT is 15, the tool coordinate system currently selected is used.
- Normally, UT of the position register is set to 15 (F is indicated on the teach pendant).
- You cannot change this value from the teach pendant. Note that once you have changed this value with CIMPPLICITY, you cannot restore the original value without using CIMPPLICITY.

Position registers have two data formats: the Cartesian coordinate format and joint format. You can check position register contents on the position register screen on the teach pendant to find which data format is currently used for a certain position register. If X, Y, Z, W, P, and R are indicated, the Cartesian coordinate format is currently used; if J1 through J6 are indicated, the joint format is used.

When reading from a position register through CIMPPLICITY, you can read the value of a desired element at any time regardless of the current data format of the position register. Read operation does not change the data format of the position register. When the data format of the data read through CIMPPLICITY differs from the current data format of the position register, note the following:

- When a value in the position register is not within the stroke range, or when there is an element indicated as “*****”, the position data format cannot be converted, so the elements not related to the current data format are all set to 0.
For example, suppose that a position register is in the Cartesian coordinate format and that the value of X is 10000 which is beyond a stroke. In this case, reading J1 to J9 through CIMPPLICITY all read 0. However, X and other elements in the Cartesian coordinate format, UF, and UT can be read correctly.
- When the data format of the data read through CIMPPLICITY differs from the current data format of the position register, data format conversion is performed at the time of communication. This processing requires several milliseconds to several tens of milliseconds, which adversely affects the communication response time. When many position registers are read, data conversion is performed several times, so this may affect the communication response time significantly.

When an untaught position register is indicated on the position register screen on the teach pendant, each element is indicated as “*****”. When an element indicated in such a way is read through CIMPPLICITY, 0 is read. To determine whether the element is actually set to 0 or is an untaught element, check the values of VALIDC and VALIDJ. If there is any untaught element, VALIDC or VALIDJ is set to 0.

When a value is written into an element of a position register through CIMPPLICITY, and that element is in the Cartesian coordinate format, the data format of the position register becomes the Cartesian coordinate format. When a value is written into an element that is in the joint format, the data format of the position register becomes the joint format. When a coordinate system number is written, the data format of the position register is left unchanged.

An example of setting \$SNPX_ASG is given below:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	50	PR[1]	1

Then, the points corresponded to the elements of position register are as follows:

Point Name	Address	Data Type	Point Name	Address	Data Type
PR1_X	%R1	REAL	PR1_J1	%R27	REAL
PR1_Y	%R3	REAL	PR1_J2	%R29	REAL
PR1_Z	%R5	REAL	PR1_J3	%R31	REAL
PR1_W	%R7	REAL	PR1_J4	%R33	REAL
PR1_P	%R9	REAL	PR1_J5	%R35	REAL
PR1_R	%R11	REAL	PR1_J6	%R37	REAL
PR1_E1	%R13	REAL	PR1_J7	%R39	REAL
PR1_E2	%R15	REAL	PR1_J8	%R41	REAL
PR1_E3	%R17	REAL	PR1_J9	%R43	REAL
PR1_FLIP	%R19	INT			
PR1_LEFT	%R20	INT	PR1_UF	%R46	INT
PR1_UP	%R21	INT	PR1_UT	%R47	INT
PR1_FRONT	%R22	INT			
PR1_TURN4	%R23	INT	PR1_VALIDC	%R26	INT
PR1_TURN5	%R24	INT	PR1_VALIDJ	%R45	INT
PR1_TURN6	%R25	INT			

Specifying @ in \$VAR_NAME

Adding @ to \$VAR_NAME was described in the section on registers. Here, @ will be explained in detail.

Suppose that you want to read or write just X, Y, and Z of PR[1] through PR[3]. Normally, \$SNPX_ASG is set as follows:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	150	PR[1]	1

When the above setting is made, the %R-to-position-register correspondence is as follows:

PLC address	Robot controller data that can be read and written
%R1-2	32-bit signed integer of X of PR[1]
%R3-4	32-bit signed integer of Y of PR[1]
%R5-6	32-bit signed integer of Z of PR[1]
%R51-52	32-bit signed integer of X of PR[2]
%R53-54	32-bit signed integer of Y of PR[2]
%R55-56	32-bit signed integer of Z of PR[2]
%R101-102	32-bit signed integer of X of PR[3]
%R103-104	32-bit signed integer of Y of PR[3]
%R105-106	32-bit signed integer of Z of PR[3]

Eighteen %R's are actually read from and written to. With the above setting, however, 150 %R's are occupied. Also, CIMPLICITY attempts to read partial data not actually used such as %R7 to %R50. This is because reading a large amount of data at one time makes communications more efficient than reading small amounts of data several times. When the position register is in the Cartesian coordinate format, however, data format conversion must be performed for the part from %R27 to %R45 which is joint data. This deteriorates the communication response time.

For efficient communication, set \$SNPX_ASG as follows:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	18	PR[1]@1.6	1

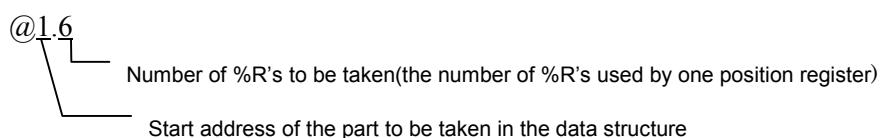
When the above setting is made, the %R-to-position-register correspondence is as follows:

PLC address	Robot controller data that can be read and written
%R1-2	32-bit signed integer of X of PR[1]
%R3-4	32-bit signed integer of Y of PR[1]
%R5-6	32-bit signed integer of Z of PR[1]
%R7-8	32-bit signed integer of X of PR[2]
%R9-10	32-bit signed integer of Y of PR[2]
%R11-12	32-bit signed integer of Z of PR[2]
%R13-14	32-bit signed integer of X of PR[3]
%R15-16	32-bit signed integer of Y of PR[3]
%R17-18	32-bit signed integer of Z of PR[3]

The modification to the settings include addition of @1.6 after PR[1] of \$VAR_NAME and change of the value in \$SIZE from 150 to 18. As a result of these modifications, the number of %R's used by one position register, which is normally 50, has decreased to 6. That is, the value "6" at the end of "@1.6" specifies the number of %R's used by one position register. The value "1" in "@1.6" specifies the

location of these six %R's in the data structure of the position register. Here, these six %R's are located at addresses 1 through 6 in the data structure.

As explained above, by specifying @ in \$VAR_NAME, a certain part can be taken from the data structure of the position register that uses 50 %R's and can be assigned repeatedly.



Specification of @ can be used not only for the position register but also for all data that uses \$SNPX_ASG. In the standard setting, @1.1 used for register assignment functions in a similar manner.

Another example is given. To assign J1 through J6 of PR[3] to PR[10], set the following:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	96	PR[3]@27.12	1

6.4 READING AND WRITING THE CURRENT POSITION (%R)

The current position of the robot can be read through CIMPPLICITY. In the same manner of the position register, set \$SNPX_ASG for %R assignment.

For the assignment of the current position, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384
\$SIZE	Meaning: Number of %R's to be assigned For the current position, 50 %R's are used. (The number of %R's used can be changed by specifying @ in \$VAR_NAME) Range: 1 to 16384
\$VAR_NAME	Meaning: Character string indicating the data to be assigned When assigning the current position, specify, for example, POS[0]. The number in brackets is the user coordinate system number. When 0 is specified as the user coordinate system number, the current position in the world coordinate system can be read. This is equivalent to selecting WORLD on the current position screen on the teach pendant. When 15 is specified as the user coordinate system number, the current position in the currently selected user coordinate system can be read. This is equivalent to selecting USER on the current position screen on the teach pendant. When a number from 1 to 9 is specified as the user coordinate system number, the current position in the specified user coordinate system can be read regardless of the user coordinate system currently selected. The data structure is the same as that of the position register. The current joint position is not affected by the user coordinates. Therefore, the current joint position can be read even when any user coordinates are specifies. In a multi-group system, POS[0] indicates the current position of the robot of group 1. To specify the robot of group 2, specify the group before the user coordinate system number, such as POS[G2:0]. When @ is specified after the character string, only J1 through J6 are assigned. This will be explained in detail later.
\$MULTIPLY	Meaning: Multiplier Only the elements such as X, Y, Z, and J1 that have a real number are affected by \$MULTIPLY. For the affected elements, see the table given below: The value of each element of the current position is multiplied by the value set in \$MULTIPLY, then the multiplication result is read or written. When \$MULTIPLY is set to 0, it has a special meaning. %R can be read and written as 32-bit real type data.

	<p>When \$MULTIPLY is set to a non-zero value, 32-bit signed integer data is read and written with the fractional part rounded off.</p> <p>Range: 0.0001 to 10000, 0</p> <p>Example: Suppose that a position register element value is 123.45:</p> <p>When \$MULTIPLY is 1, 123 is read.</p> <p>When \$MULTIPLY is 10, 1235 is read.</p> <p>When \$MULTIPLY is 0.1, 12 is read.</p> <p>When \$MULTIPLY is 0, 123.45 (real number) is read.</p>
--	--

NOTE

The current position is read only. Even when a write operation is performed, nothing occurs.

The data format is the same as that of the position register. Fifty %R's are used.

When UT is read, 15 is always read.

When UF is read, the user coordinate system number specified in \$VAR_NAME is read.

Unlike the position register, consecutive assignment is not permitted for the current position. Even when settings are made as shown below, for example, the current position with user coordinate system number 1 cannot be assigned to %R51 to %R100. For POS[0] and POS[1], set \$SNPX_ASG respectively.

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	100	POS[0]	1

6.5 READING ALARM HISTORY (%R)

Robot controller alarm history can be read through CIMPPLICITY. Set \$SNPX_ASG to perform %R assignment in the same manner as for registers.

When assigning alarm history, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384
\$SIZE	Meaning: Number of %R's to be assigned For each alarm, 100 %R's are used. (The number of %R's used for each alarm can be changed by specifying @ in \$VAR_NAME) Range: 1 to 16384
\$VAR_NAME	Meaning: Character string indicating the data to be assigned When assigning alarm history, specify, for example, ALM[1]. The number in brackets corresponds to the line number on the alarm screen. The line number of the latest alarm is set to 1. When ALM[1] is specified, the alarm displayed on the alarm issuance screen on the teach pendant is read. When E is added before the alarm number such as ALM[E1], the alarm displayed on the alarm history screen on the teach pendant is read. When the password option is ordered, specifying P before the alarm number such as ALM[P1] allows reading of the operation history displayed on the password history screen on the teach pendant.
\$MULTIPLY	Meaning: Multiplier There is no data in alarm history that required the multiplier. \$MULTIPLY for alarm history specifies the data format of character string data. Whenever communicating with CIMPPLICITY, set this element to 1. Range: 1

NOTE

The alarm history is read only. Even when a write operation is performed, nothing occurs.

Each alarm use 100 %R's. The contents of the 100 %R's are listed below:

%R address	Explanation
1	Alarm ID 16-bit signed integer For SRVO-001, the alarm ID indicating the servo, 11, is read. The alarm ID number are listed in the alarm code table in the robot controller Operator's Manual.
2	Alarm number 16-bit signed integer For SRVO-001, 1 in 001 is read.

3	Cause code: alarm ID	16-bit signed integer	Normally, when an alarm is issued, the alarm is indicated in the first line of the teach pendant. Some alarms are indicated in the upper two lines of the teach pendant. In this case, the indication on the second line is the cause code. At this address, the alarm ID of the cause code can be read. For an alarm with no cause code, 0 is read.																								
4	Cause code: alarm number	16-bit signed integer	The alarm number of the cause code can be read. For an alarm with no cause code, 0 is read.																								
5	Alarm severity	16-bit signed integer	A numeric value indicating the alarm severity can be read. <table> <tr> <td>NONE</td><td>128</td><td>SERVO</td><td>54</td></tr> <tr> <td>WARN</td><td>0</td><td>ABORT.L</td><td>11</td></tr> <tr> <td>PAUSE.L</td><td>2</td><td>ABORT.G</td><td>45</td></tr> <tr> <td>PAUSE.G</td><td>34</td><td>SERVO2</td><td>58</td></tr> <tr> <td>STOP.L</td><td>6</td><td>SYSTEM</td><td>122</td></tr> <tr> <td>STOP.G</td><td>38</td><td></td><td></td></tr> </table>	NONE	128	SERVO	54	WARN	0	ABORT.L	11	PAUSE.L	2	ABORT.G	45	PAUSE.G	34	SERVO2	58	STOP.L	6	SYSTEM	122	STOP.G	38		
NONE	128	SERVO	54																								
WARN	0	ABORT.L	11																								
PAUSE.L	2	ABORT.G	45																								
PAUSE.G	34	SERVO2	58																								
STOP.L	6	SYSTEM	122																								
STOP.G	38																										
6	Date of occurrence (calendar year)	16-bit signed integer																									
7	Date of occurrence (month)	16-bit signed integer																									
8	Date of occurrence (day)	16-bit signed integer																									
9	Time of occurrence (hours in 24-hour clock)	16-bit signed integer																									
10	Time of occurrence (minutes)	16-bit signed integer																									
11	Time of occurrence (seconds)	16-bit signed integer																									
12-51	Alarm message	Up to 80 characters	An alarm message can be read. The characters displayed on the first line on the teach pendant including the characters of the alarm code such as "SRVO-001" can be read directly.																								
52-91	Cause code message	Up to 80 characters	The alarm message of the cause code can be read.																								
92-100	Alarm severity characters	Up to 18 characters	A character string indicating alarm severity such as WARN can be read.																								

For RESET, 0 is read for both the alarm ID and alarm number. For the alarm message, RESET is read.

When there are only two alarm lines on the alarm occurrence screen, the elements of ALM[3] and later are all set to 0.

In elements containing a character string such as an alarm message, the remaining part after the character string are set to 0. Katakana characters are indicated in shifted JIS character code.

Alarm history has such a large data structure that each alarm uses 100 %R's, and most of the data structure is for character strings such as an alarm message. If you don't need these strings, @ can be specified for efficient communication.

For example, set \$SNPX_ASG as follow:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	12	ALM[E1]@1.4	1

Then, the %R-to-alarm-history correspondence is as follows:

PLC address	Robot controller data that can be read and written
%R1	Alarm ID of alarm 1
%R2	Alarm number of alarm 1
%R3	Alarm ID of cause code of alarm 1
%R4	Alarm number of cause code of alarm 1
%R5	Alarm ID of alarm 2
%R6	Alarm number of alarm 2
%R7	Alarm ID of cause code of alarm 2
%R8	Alarm number of cause code of alarm 2
%R9	Alarm ID of alarm 2
%R10	Alarm number of alarm 3
%R11	Alarm ID of cause code of alarm 3
%R12	Alarm number of cause code of alarm 3

6.6 READING THE PROGRAM EXECUTION STATUS (%R)

The robot controller program execution status can be read through CIMPPLICITY. In the same manner as registers, set \$SNPX_ASG for %R assignment.

For assignment of the program execution status, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384
\$SIZE	Meaning: Number of %R's to be assigned For one task, 18 %R's are used. (The number of R's used by one task can be changed by specifying @ in \$VAR_NAME) Range: 1 to 16384
\$VAR_NAME	Meaning: Character string indicating the data to be assigned When assigning the program execution status, specify, for example, PRG[1]. The number in brackets is the task number. Unless multitasking is performed, PRG[1] is always set. When two tasks are executed simultaneously in a multitasking system, PRG[1] and PRG[2] are used to read the execution statuses of these tasks. Between the two tasks, the task assigned to PRG[1] depends on the program activating timing and communication timing. Once the execution status of a task has been read as PRG[1], it can be read as PRG[1] continuously until program execution terminates.
\$MULTIPLY	Meaning: Multiplier The program execution status has no data that requires the multiplier. \$MULTIPLY specifies the data format of character string data. Whenever communicating with CIMPPLICITY, set this element to 1. Range: 1

NOTE

The program execution status is read only. Even when a write operation is performed, nothing occurs.

Each task uses 18 %R's. The contents of the 18 %R's are listed below:

%R address	Explanation
1-8	Program name Up to 16 characters Name of a program being executed When a subprogram is called, the name of the subprogram being executed
9	Line number 16-bit signed integer Line number of a program being executed When a subprogram is called, the line number of the subprogram being executed

10	Execution status 16-bit signed integer End 0 Pause 1 Running 2
11-18	Calling program name Up to 16 characters Name of a program started first When no subprogram is called, the same name as the program name is set.

When the program has already terminated, all elements are set to 0.

You can read just a part of elements by specifying @.

For example, set \$SNPX_ASG as follows:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	4	PRG[1]@9.2	1

Then, the %R-to-program-execution-status correspondence is as follows:

PLC address	Robot controller data that can be read and written
%R1	Line number of task 1
%R2	Execution status of task 1
%R3	Line number of task 2
%R4	Execution status of task 2

6.7 READING FROM AND WRITING INTO SYSTEM VARIABLES (%R)

Robot controller system variables can be read from and written into through CIMPPLICITY. Set \$SNPX_ASG to perform %R assignment in the same manner as for registers.

When assigning system variables, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384
\$SIZE	Meaning: Number of %R's to be assigned The number of %R's required for a system variable depends on the data type of the system variable. For details, see the table given below. (The number of %R's used for one system variable can be changed by specifying @ in \$VAR_NAME.) Range: 1 to 16384
\$VAR_NAME	Meaning: Character string indicating the data to be assigned When assigning a system variable, set a system variable name such as \$WAITTMOUT. When a system variable with an array such as UALARM_SEV[1] is specified, array elements are assigned successively in the same manner as for registers. When assigning a KAREL variable, set character string indicating a KAREL program name and a KAREL variable name such as \$[KAREL-program-name]KAREL-variable-name.
\$MULTIPLY	Meaning: Multiplier The meaning of this element varies depending on the data type of the specified system variable. For details, see the table given below.



CAUTION

The reading from and writing to KAREL variables functions are supported by R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC).

When a system variable is read from or written to, the data type used with CIMPPLICITY reads from or writes into the system variable and the meaning of \$MULTIPLY vary depending on the data type of the system variable. In addition, CIMPPLICITY cannot sometimes read from or write into a system variable depending on the data type of the system variable.

For the data types of system variables that can be read from and written into through CIMPPLICITY, the number of %R's used for each system variable, the meaning of \$MULTIPLY, and the data type used for read/write by CIMPPLICITY, see the table given below.

Data type of system variable	Number of %R's required per system variable	Meaning of \$MULTIPLY and data type used for read/write
INTEGER 32-bit signed integer	2	The value in the system variable, multiplied by the value in \$MULTIPLY, is read/written. The fractional part is rounded off. When \$MULTIPLY is 0, it has the same meaning as when it is 1. Read/write data as 32-bit signed integer.
SHORT 16-bit signed integer	2	
BYTE 8-bit signed integer	2	
REAL 32-bit real number	2	The value in the system variable, multiplied by the value in \$MULTIPLY, is read/written. When \$MULTIPLY is 0, it has a special meaning, and data can be read/written a 32-bit real type data. When \$MULTIPLY is a non-zero value, data is treated as 32-bit signed integer data with its fractional part rounded off.
BOOLEAN TRUE/FALSE	2	Read/write data as a 32-bit signed integer. If TRUE is set, 1 is read. If FALSE is set, 0 is read. Writing 0 sets FALSE. Writing a non-zero value sets TRUE. \$MULTIPLY is not used.
POSITION Position data	50	This data type has the same data structure as the position register. Each element can be read/written. \$MULTIPLY has also the same meaning as for the position register.
STRING Character string	40	The data type of a character string is specified. Whenever communicating with CIMPLICITY, set 1.

For INTEGER, SHORT, and BYTE, data can be read and written through CIMPLICITY similarly, so there is no need to distinguish these data types. When an integer value is indicated for a system variable on the system variable screen, the system variable is of the INTEGER, SHORT, or BYTE type.

When decimal positions are indicated for a system variable on the system variable screen, the system variable is of the REAL type.

When TRUE or FALSE is indicated for a system variable on the system variable screen, the system variable is of the BOOLEAN type.

When a character string is indicated for a system variable on the system variable screen, the system variable is of the STRING type.

When POSITION is indicated for a system variable on the system variable screen, the system variable is of the POSITION type.

**CAUTION**

Even when a system variable is indicated as a read only variable on the system variable screen, CIMPLICITY can write into this system variable. If an inappropriate value is written into a system variable, the system may be affected significantly. Therefore, before writing a value into a system variable, carefully consider the meaning of the system variable.

You can specify @ to read only specific elements.

For example, to assign X, Y, and Z of user coordinate systems 1 and 2 of group 1, set \$SNPX_ASG as follows:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	12	\$MNUFRAME[1,1]@1.6	1

Then, the %R-to-system-variable correspondence is as follows:

PLC address	Robot controller data that can be read and written
%R1-2	X of user coordinate system 1 of group 1 (X of \$MNUFRAME[1,1])
%R3-4	Y of user coordinate system 1 of group 1 (Y of \$MNUFRAME[1,1])
%R5-6	Z of user coordinate system 1 of group 1 (Z of \$MNUFRAME[1,1])
%R7-8	X of user coordinate system 2 of group 1 (X of \$MNUFRAME[1,2])
%R9-10	Y of user coordinate system 2 of group 1 (Y of \$MNUFRAME[1,2])
%R11-12	Z of user coordinate system 2 of group 1 (Z of \$MNUFRAME[1,2])

6.8 READING AND WRITING THE COMMENT OF REGISTERS, POSITION REGISTERS, AND I/O (%R)

The comment of robot controller registers, position registers, and I/O can be read from and written to through CIMPLICITY. In the same manner as registers, set \$SNPX_ASG for %R assignment.

For assignment of the comment, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation																																				
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384																																				
\$SIZE	Meaning: Number of %R's to be assigned For one comment, 40 %R's are used. (The number of %R's used by one comment can be changed by specifying @ in \$VAR_NAME) Range: 1 to 16384																																				
\$VAR_NAME	<p>Meaning: Character string indicating the data to be assigned When assigning the comment of register, specify, for example, R[C1]. The number in brackets is the register number. When assigning the comment of others, specify as listed below:</p> <table> <tr><td>Register</td><td>R[C1]</td></tr> <tr><td>Position Register</td><td>PR[C1]</td></tr> <tr><td>DI</td><td>DI[C1]</td></tr> <tr><td>DO</td><td>DO[C1]</td></tr> <tr><td>RI</td><td>RI[C1]</td></tr> <tr><td>RO</td><td>RO[C1]</td></tr> <tr><td>UI</td><td>UI[C1]</td></tr> <tr><td>UO</td><td>UO[C1]</td></tr> <tr><td>SI</td><td>SI[C1]</td></tr> <tr><td>SO</td><td>SO[C1]</td></tr> <tr><td>WI</td><td>WI[C1]</td></tr> <tr><td>WO</td><td>WO[C1]</td></tr> <tr><td>WSI</td><td>WSI[C1]</td></tr> <tr><td>WSO</td><td>WSO[C1]</td></tr> <tr><td>GI</td><td>GI[C1]</td></tr> <tr><td>GO</td><td>GO[C1]</td></tr> <tr><td>AI</td><td>AI[C1]</td></tr> <tr><td>AO</td><td>AO[C1]</td></tr> </table> <p>Comments of consecutive registers such as R[2] to R[5] can be assigned at one time. In this case, the number of the register is four and the number of %R's required per register is forty, so set 160 in \$SIZE and set R[C2] in \$VAR_NAME. Here, the index 2 indicates that registers are assigned sequentially from R[2].</p> <p>The number of %R's per comment can be set to 5 by adding @1.5 to the end of the character string. In this case, the top 10 characters in the comment are read/written. (One %R includes 2 characters.)</p> <p>Example: R[C1]@1.5 R[C1] indicates that registers that comment is read/written are assigned sequentially from R[1].</p>	Register	R[C1]	Position Register	PR[C1]	DI	DI[C1]	DO	DO[C1]	RI	RI[C1]	RO	RO[C1]	UI	UI[C1]	UO	UO[C1]	SI	SI[C1]	SO	SO[C1]	WI	WI[C1]	WO	WO[C1]	WSI	WSI[C1]	WSO	WSO[C1]	GI	GI[C1]	GO	GO[C1]	AI	AI[C1]	AO	AO[C1]
Register	R[C1]																																				
Position Register	PR[C1]																																				
DI	DI[C1]																																				
DO	DO[C1]																																				
RI	RI[C1]																																				
RO	RO[C1]																																				
UI	UI[C1]																																				
UO	UO[C1]																																				
SI	SI[C1]																																				
SO	SO[C1]																																				
WI	WI[C1]																																				
WO	WO[C1]																																				
WSI	WSI[C1]																																				
WSO	WSO[C1]																																				
GI	GI[C1]																																				
GO	GO[C1]																																				
AI	AI[C1]																																				
AO	AO[C1]																																				

	@1.1 indicates that the top 10 characters in the comment are read/written.
\$MULTIPLY	Meaning: \$MULTIPLY specifies the data format of character string data. Whenever communicating with CIMPLICITY, set this element to 1.

**CAUTION**

This function is supported by R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC).

You can specify @ to read only specific elements.

For example, set \$SNPX_ASG as follows:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	50	R[C1]@1.10	1

Then, the %R-to-comment-of-register correspondence is as follows:

PLC address	Robot controller data that can be read and written
%R1-10	Top 20 characters of comment of R[1]
%R11-20	Top 20 characters of comment of R[2]
%R21-30	Top 20 characters of comment of R[3]
%R31-40	Top 20 characters of comment of R[4]
%R41-50	Top 20 characters of comment of R[5]

6.9 READING AND WRITING THE VALUE AND SIM STATUS OF I/O (%R)

I/O values can be read and written assigning to %I, %Q, %AI, %AQ. And I/O values can be also assigned to %R like registers. And I/O SIM status can be read and written through SIMPLICITY.

Set \$SNPX_ASG to perform %R assignment in the same manner as for registers.

When assigning the value or SIM status of I/O, set the elements of \$SNPX_ASG as follows:

\$SNPX_ASG element	Explanation																																																					
\$ADDRESS	Meaning: Start address of %R to be assigned Range: 1 to 16384																																																					
\$SIZE	Meaning: Number of %R's to be assigned The number of %R's required for I/O depends on the value of \$MULTIPLY. When \$MULTIPLY is 1, one %R is required per one I/O. When \$MULTIPLY is 0, one %R is required per sixteen I/O. (In the case GI/O and AI/O, one %R is always required per one I/O undepending on the value of \$MULTIPLY.) Range: 1 to 16384																																																					
\$VAR_NAME	<div>Meaning: Character string indicating the data to be assigned When assigning I/O, specify, for example, DI[1]. The number in brackets is an I/O number.</div> <table><tr><td></td><td>Value</td><td>SIM status</td></tr><tr><td>DI</td><td>DI[1]</td><td>DI[S1]</td></tr><tr><td>DO</td><td>DO[1]</td><td>DO[S1]</td></tr><tr><td>RI</td><td>RI[1]</td><td>RI[S1]</td></tr><tr><td>RO</td><td>RO[1]</td><td>RO[S1]</td></tr><tr><td>UI</td><td>UI[1]</td><td></td></tr><tr><td>UO</td><td>UO[1]</td><td></td></tr><tr><td>SI</td><td>SI[1]</td><td></td></tr><tr><td>SO</td><td>SO[1]</td><td></td></tr><tr><td>WI</td><td>WI[1]</td><td>WI[S1]</td></tr><tr><td>WO</td><td>WO[1]</td><td>WO[S1]</td></tr><tr><td>WSI</td><td>WSI[1]</td><td>WSI[S1]</td></tr><tr><td>WSO</td><td>WSO[1]</td><td>WSO[S1]</td></tr><tr><td>GI</td><td>GI[1]</td><td>GI[S1]</td></tr><tr><td>GO</td><td>GO[1]</td><td>GO[S1]</td></tr><tr><td>AI</td><td>AI[1]</td><td>AI[S1]</td></tr><tr><td>AO</td><td>AO[1]</td><td>AO[S1]</td></tr></table> <div>Example: Assigning the values of 32 I/O's, DI[11] to DI[42] When \$MULTIPLY is 1, set 32 to \$SIZE, and set DI[11] to \$VAR_NAME. Here, the index 11 indicates that I/O's are assigned sequentially from DI[11]. When \$MULTIPLY is 0, set 2 to \$SIZE because one %R is assigned to 16 I/O's, and set DI[11] to \$VAR_NAME.</div>				Value	SIM status	DI	DI[1]	DI[S1]	DO	DO[1]	DO[S1]	RI	RI[1]	RI[S1]	RO	RO[1]	RO[S1]	UI	UI[1]		UO	UO[1]		SI	SI[1]		SO	SO[1]		WI	WI[1]	WI[S1]	WO	WO[1]	WO[S1]	WSI	WSI[1]	WSI[S1]	WSO	WSO[1]	WSO[S1]	GI	GI[1]	GI[S1]	GO	GO[1]	GO[S1]	AI	AI[1]	AI[S1]	AO	AO[1]	AO[S1]
	Value	SIM status																																																				
DI	DI[1]	DI[S1]																																																				
DO	DO[1]	DO[S1]																																																				
RI	RI[1]	RI[S1]																																																				
RO	RO[1]	RO[S1]																																																				
UI	UI[1]																																																					
UO	UO[1]																																																					
SI	SI[1]																																																					
SO	SO[1]																																																					
WI	WI[1]	WI[S1]																																																				
WO	WO[1]	WO[S1]																																																				
WSI	WSI[1]	WSI[S1]																																																				
WSO	WSO[1]	WSO[S1]																																																				
GI	GI[1]	GI[S1]																																																				
GO	GO[1]	GO[S1]																																																				
AI	AI[1]	AI[S1]																																																				
AO	AO[1]	AO[S1]																																																				
\$MULTIPLY	Meaning: \$MULTIPLY specifies if I/O is assigned as a bit value or not. One I/O is assigned to one %R 1 Sixteen I/O's are assigned to one %R 0																																																					

**CAUTION**

This function is supported by R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC).

When \$MULTIPLY is 1, the value or SIM status of one I/O is assigned to one %R.

When the value is ON, the value of the corresponded %R is 1. When OFF, the value is 0.

When \$MULTIPLY is 0, the values or SIM statuses of sixteen I/O's are assigned one %R as bit values.

When the value is ON, the value of the corresponded bit of %R is 1.

When OFF, the value is 0.

The I/O with less index number is assigned to the less significant bit of %R.

Example: DI[1] to DI[16] are assigned to %R

When DI[1] is ON and others are OFF, the value of %R1 is 1.

When DI[16] is ON and others are OFF, the value of %R1 is 32768.

(In the case 16-bit signed integer)

6.10 SETTING \$SNPX_ASG FROM CIMPLICITY (%G)

When you want to read or write data other than I/O signal data, you must set \$SNPX_ASG. Usually, this is set on the system variable screen. \$SNPX_ASG can also be set through communication from CIMPLICITY.

When a character string is written to PLC address %G, robot controller performs processing regarding the written character string as a command. Robot controller performs command processing similarly at any address number if it is within the %G area. The following commands can be executed:

CLRASG Erasing \$SNPX_ASG

Format: CLRASG

Function: Initializes all \$SNPX_ASG settings.

Execute this command once before executing a SETASG command sequence.

SETASG Setting \$SNPX_ASG

Format: SETASG (\$ADDRESS) (\$SIZE) (\$VAR_NAME) [(\$MULTIPLY)]

Function: Sets a specified value in \$SNPX_ASG.

Robot controller automatically selects an unused \$SNPX_ASG area for setting. So there is no need to specify the number of \$SNPX_ASG.

\$MULTIPLY may be omitted. By default, \$MULTIPLY is set to 1.

Before executing the SETASG command, execute CLRASG once.

SETVAR Setting a system variable

Format: SETVAR (system-variable-name) (value)

Function: Set a value in a specified system variable.

The data types of system variables that can be set are INTEGER, SHORT, BYTE, REAL, BOOLEAN, and STRING.

For the BOOLEAN type, specify 1 for TRUE, or 0 for FALSE.

When specifying a character string that includes blanks, enclose the character string with double quotation marks ("").

CLRALM Erasing alarm histories

Format: CLRALM

Function: Initializes all alarm histories.

Example: When setting \$SNPX_ASG as follows:

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	2	R[1]@1.1	1
\$SNPX_ASG[2]	3	4	R[1]	100
\$SNPX_ASG[3]	7	4	R[2]	0.1
\$SNPX_ASG[4]	11	2	R[1]	0

Write the following character strings in %G sequentially:

CLRASG
SETASG 1 2 R[1]@1.1 1
SETASG 3 4 R[1] 100
SETASG 7 4 R[2] 0.1
SETASG 11 2 R[1] 0

Some commands can be written at once marking off them by CR(0x0A) or LF(0x0D).

In this case, adding CR or LF at the end of the last command makes the execution a little faster.

**CAUTION**

CLRALM command and the function writing some commands at once are supported by R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC).

6.11 NOTES AND TIPS ON USAGE

6.11.1 \$SNPX_ASG Is Set, but Data That Should Have Been Assigned Cannot Be Read/Written.

When a %R area not assigned with \$SNPX_ASG is read from, 0 is always read. When a non-zero value is set at the %R address in question, the %R address can be considered to have been assigned to other data.

When \$SNPX_ASG is set as follows, for example, %R101 through %R150 are assigned both by \$SNPX_ASG[1] and \$SNPX_ASG[2].

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	1000	R[1]@1.1	1
\$SNPX_ASG[2]	101	50	PR[1]	100

In this case, \$SNPX_ASG with the smaller number takes precedence.

In the above example, \$R101 to %R150 are assigned to R[101] to R[150]. So, PR[1] cannot be read from nor written to.

Next, suppose that 0 is set at the %R address in question. Also in this case, it is considered that other data is assigned to the %R area as explained above, and that the data is 0 by accident. Check for a duplicate assignment.

If a duplicate assignment is not made, the assignment setting itself may have a problem. Check the \$SNPX_ASG setting. The following are error-prone:

- The correct formats for \$VAR_NAME are listed below. If the setting does not follow any of these formats, assignment is not performed.

"R[number]"	
"PR[number]"	When specifying a group, specify a colon (:) after a group number.
"PR[G number:number]"	
"POS[number]"	When specifying a group, specify a colon (:) after a group number.
"POS[G number:number]"	
"ALM[number]"	Specify a number immediately after E and P. A colon (:) is not added.
"ALM[E number]"	
"ALM[P number]"	
"PRG[number]"	
System-variable-name	A system variable name begins with a dollar sign (\$).
\$(KAREL-program-name)KAREL-variable-name	
"DI[number]" "DI[S number]" "DI[C number]"	
"DO[number]" "DO[S number]" "DO[C number]"	
"RI[number]" "RI[S number]" "RI[C number]"	
"RO[number]" "RO[S number]" "RO[C number]"	

"UI[number]" "UI[S number]" "UI[C number]" "UO[number]" "UO[S number]" "UO[C number]" "SI[number]" "SI[S number]" "SI[C number]" "SO[number]" "SO[S number]" "SO[C number]" "WI[number]" "WI[S number]" "WI[C number]" "WO[number]" "WO[S number]" "WO[C number]" "WSI[number]" "WSI[S number]" "WSI[C number]" "WSO[number]" "WSO[S number]" "WSO[C number]" "GI[number]" "GI[S number]" "GI[C number]" "GO[number]" "GO[S number]" "GO[C number]" "AI[number]" "AI[S number]" "AI[C number]" "AO[number]" "AO[S number]" "AO[C number]"	
--	--

- When a blank is included in \$VAR_NAME, assignment is not performed. Also, when using @, be sure to specify @ immediately after a variable name without inserting a blank.
- When successive assignment is performed, the number of %R's used for one element varies depending on the type of assigned data. Confirm the explanation of each data.
- When specifying @, follow the format "@number.number". A period (.) is used between the numbers. Be sure to specify @ immediately after the variable name.

6.11.2 Improving Communication Efficiency

- Read/Write of I/O signals is faster than read/write of %R's assigned with \$SNPX_ASG. The speed of read/write of %R may be slower during program execution. On the other hand, read/write of I/O signals is not affected by program execution. When emphasis is placed on the communication speed, it is recommended that the use of %R be avoided if possible.
- Data conversion of a position register between the Cartesian coordinate and joint formats requires a significant amount of time. In position register assignment, @ can be used to assign only those elements that are actually required. This can improve communication efficiency. In addition, when data is read through CIMPLICITY, using the same data format as that of the actual position register eliminates the necessity of data conversion, thus improving communication efficiency.
- CIMPLICITY sometimes automatically reads from an address that actually is not used. In such a case, to prevent the slowing of communication speed, do not assign unnecessary data whenever possible. In addition, assign %R's so that they are as close as possible to compact the %R area for read/write. This can improve communication efficiency.

6.11.3 Version Of Communication Function With CIMPLICITY

The value of \$SNPX_PARAM.\$VERSION specifies the version of communication function with CIMPLICITY. When \$SNPX_PARAM.\$VERSION does not exist, the version is 1.

The functions supported in version 2 or later are listed below:

- Reading and writing the comment of registers, position registers, and I/O's
- Reading and writing the WDI/O and WSI/O
- Reading and writing the i/O SIM status
- Writing the plural commands at once
- CLRALM command
- Assigning I/O value to %R
- Reading and writing the KAREL variables
- Multi Connection and multiplex \$SNPX_ASG

6.11.4 Multi Connection And Multiplex \$SNPX_ASG

When the version is 2 or later, one robot controller can accept some TCP/IP connections with CIMPPLICITY at once. (Only one connection can be accepted communicating by RS-232-C.)

Normally, the multi connection is disenabled. To enable the multi connection, set the number of CIMPPLICITY to \$SNPX_PARAM.\$NUM_CIMP. (Setting too large value may make the robot controller stop because of the lack of memory area.)

When the version is 2 or later, the multiplex \$SNPX_ASG is supported to accept that some CIMPPLICITY set the different values to \$SNPX_ASG.

When the version is 1, CLRASG command erases the setting of \$SNPX_ASG. When the version is 2 or later, CLRASG command creates the local \$SNPX_ASG for one CIMPPLICITY that writes this command. After that, this local \$SNPX_ASG is used for the communication with this CIMPPLICITY instead of the system variable \$SNPX_ASG. This local \$SNPX_ASG is deleted when this connection is disconnected.

After CLRASG command is executed, the results of SETASG command cannot be confirmed on the system variable screen. When CLRASG command is not executed, the system variable \$SNPX_ASG is used for SETASG command or reading/writing %R's.

To disenable the multiplex \$SNPX_ASG, set 0 to \$SNPX_PARAM.\$NUM_CIMP. (Default value is 0.)

When the number of connection is over this value, the connection that has not communicated for the longest time is disconnected.

7

APPENDIX

7.1 SAMPLE PROJECT OF “HMI FOR ROBOT”

7.1.1 Sample Project Overview

CIMPLICITY HMI for Robot Package includes the sample project that communicates with 3 robots.

In the sample project, many points for robot controller data are registered, and there are some screens to monitor the robot information, such as, for example, program execution status, robot motion, the values of position registers, and alarm history.

You can easily create your own project modifying this sample project.

This sample project is consists of 6 screens.

- **START.CIM**
Start page. Open this first, make Full Screen, and click the button to go TOP.CIM.
- **TOP.CIM**
You can see the motion of 3 robots as 3D animation with ROBOGUIDE.
- **UOP1.CIM, UOP2.CIM, UOP3.CIM**
You can see the statuses such as robot signals, robot program execution status, and current alarm.
- **POS1.CIM, POS2.CIM, POS3.CIM**
You can see the current position of the robot as 3D animation.
- **ALARM1.CIM, ALARM2.CIM, ALARM3.CIM**
You can see the alarm history.
And you can also get detail informations about alarm in history as HTML document.
- **REG1.CIM, REG2.CIM, REG3.CIM**
You can see and change the values and comment of position registers.

7.1.2 Registered Devices

Three devices named SNPE1, SNPE2, and SNPE3 for each robot.

Here, the Model type of the device must be “GE FANUC Series 90-30”.

7.1.3 Registered Points

In this sample project, \$SNPX_ASG in each robot controller must be as follow:

(These parameters are automatically set in the script. This will be explained in detail later.)

	\$ADDRESS	\$SIZE	\$VAR_NAME	\$MULTIPLY
\$SNPX_ASG[1]	1	400	R[1]	1
\$SNPX_ASG[2]	401	5000	PR[1]	0
\$SNPX_ASG[3]	5401	1000	ALM[1]	1
\$SNPX_ASG[4]	6401	1000	ALM[E1]	1
\$SNPX_ASG[5]	7401	50	POS[0]	0
\$SNPX_ASG[6]	7451	18	PRG[1]	1

\$SNPX_ASG[7]	7501	100	R[C1]@1.10	1
\$SNPX_ASG[8]	7601	100	PR[C1]@1.10	1
\$SNPX_ASG[9]	7701	2	\$mcr.\$genoverride	1

And many pointst for robot controller data are registered for SNPE1 as follow:

("_2" and "_3" are added to the end of name of points for SNPE2 and SNPE3, such as DI1_2.)

- I/O Signal
 - DI1 to DI20 (Degital Input DI[1] to DI[20]) : As BOOL
 - DO1 to DO20 (Degital Output DO[1] to DO[20]) : As BOOL
 - UI1 to UI18 (UOP Input UI[1] to UI[18]) : As BOOL
 - UO1 to UO20 (UOP Output UO[1] to UO[20]) : As BOOL
 - SI0 to SI15 (SOP Input SI[0] to SI[15]) : As BOOL
 - SO0 to SO15 (SOP Output SO[0] to SO[15]) : As BOOL
 - GI1 to GI5 (Group Input GI[1] to GI[5]) : As UINT
 - GO1 to GO5 (Group Output GO[1] to GO[5]) : As UINT
- Register
 - R1 to R10 (R[1] to R[10]) : As DINT
 - RCOMMENT1 to RCOMMENT10 (Comments of R[1] to R[10]) : As STRING_20
- Position Register
 - PR1_X, Y, Z, W, P, R to PR10_X, Y, Z, W, P, R (X, Y, Z, W, P, R of PR[1] to PR[10]) : As REAL
 - PR1_FLIP to PR10_FLIP (FLIP of PR[1] to PR[10]) : As INT
 - PR1_LEFT to PR10_LEFT (LEFT of PR[1] to PR[10]) : As INT
 - PR1_UP to PR10_UP (UP of PR[1] to PR[10]) : As INT
 - PR1_FRONT to PR10_FRONT (FRONT of PR[1] to PR[10]) : As INT
 - PR1_T1, T2, T3 to PR10_T1, T2, T3 (TRUN4, 5, 6 of PR[1] to PR[10]) : As INT
 - PR1_J1, J2, J3, J4, J5, J6 to PR10_J1, J2, J3, J4, J5, J6 (J1 to J6 of PR[1] to PR[10]) : As REAL
 - PRCOMMENT1 to PRCOMMENT10 (Comments of PR[1] to PR[10]) : As STRING_20
- Current Position
 - CUR_POS_X, Y, Z, W, P, R (X, Y, Z, W, P, R of Current Position) : As REAL
 - CUR_POS_FLIP (FLIP of Current Position) : As INT
 - CUR_POS_LEFT (LEFT of Current Position) : As INT
 - CUR_POS_UP (UP of Current Position) : As INT
 - CUR_POS_FRONT (FRONT of Current Position) : As INT
 - CUR_POS_T1, T2, T3 (TRUN4, 5, 6 of Current Position) : As INT
 - CUR_POS_J1, J2, J3, J4, J5, J6 (J1 to J6 of Current Position) : As REAL
 - CUR_POS_VALIDC (VALIDC of Current Position) : As INT

- CUR_POS_VALIDJ (VALIDJ of Current Position) : As INT
- Current Alarm
 - ALM1_ID (Alarm ID of Current Alarm) : As INT
 - ALM1_NO (Alarm Number of Current Alarm) : As INT
 - ALM1_MSG (Alarm Message of Current Alarm) : As STRING_80
- Alarm History
 - ALM_ID_REAL1 to ALM_ID_REAL10 (Alarm ID of Alarms) : As INT
 - ALME_NO_REAL1 to ALME_NO_REAL10 (Alarm Number of Alarms) : As INT
 - ALME_MSG_REAL1 to ALME_MSG_REAL10 (Alarm Message of Alarms) : As STRING_80
 - ALM_SEV_REAL1 to ALM_SEV_REAL10 (Alarm Severity of Alarms) : As INT
 - ALME_DATE_REAL1[0] to ALME_DATE_REAL10[0] (Year of Occurrence of Alarms) : As INT
 - ALME_DATE_REAL1[1] to ALME_DATE_REAL10[1] (Month of Occurrence of Alarms) : As INT
 - ALME_DATE_REAL1[2] to ALME_DATE_REAL10[2] (Day of Occurrence of Alarms) : As INT
 - ALME_DATE_REAL1[3] to ALME_DATE_REAL10[3] (Hour of Occurrence of Alarms) : As INT
 - ALME_DATE_REAL1[4] to ALME_DATE_REAL10[4] (Minute of Occurrence of Alarms) : As INT
 - ALME_DATE_REAL1[5] to ALME_DATE_REAL10[5] (Second of Occurrence of Alarms) : As INT
- Program Execution Status
 - PRG_NAME (Program Name) : As STRING_20
 - PRG_LINE (Line Number) : As UINT
 - PRG_STATUS (Execution Status) : As UINT
 - PRG_ROOT (Calling Program Name) : As STRING_20
 - OVERRIDE (Override) : As DINT
- Command String
 - COMMAND_1 (Command String for Setting \$SNPX_ASG) : As STRING_80

**CAUTION**

%R8001 to %R8100 and %R8101 to %R8200 are assigned to the comments of R[1] to R[10] and PR[1] to PR[10], and the points, RCOMMENT1 to RCOMMENT10 and PRCOMMENT1 to PRCOMMENT10, refer to these %R's. These settings are supported by R-J3iB system software 7D80, 45 or later, and 7D81, 09 or later, and R-30iA(R-J3iC).

7.1.4 \$SNPX_ASG Settings (em_init.bcl)

In the VB script em_init.bcl, the values of \$SNPX_ASG are set for assignment to %R's as follows:

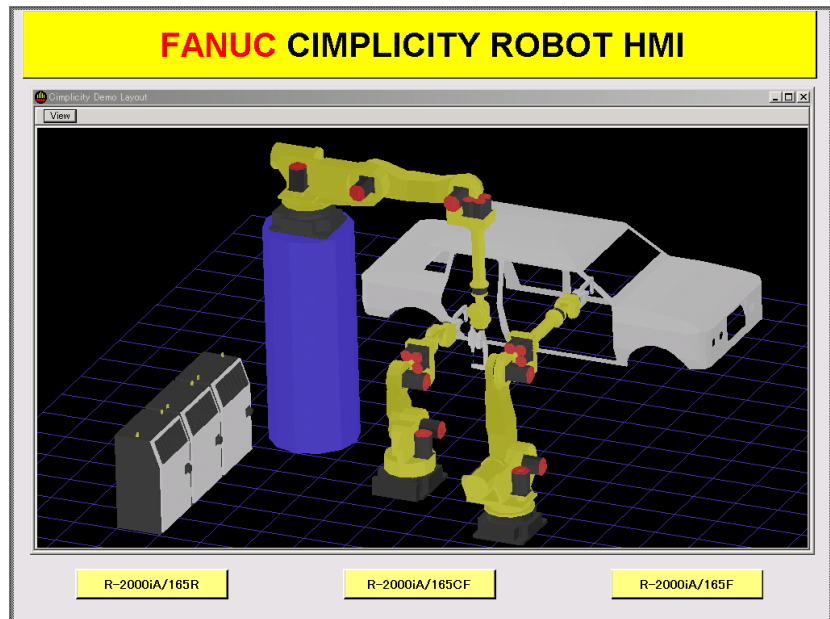
```
intRobotSum = 3
For i = 1 To intRobotSum
    PointSet "COMMAND_" & CStr(i),"CLRASG"
    PointSet "COMMAND_" & CStr(i),"SETASG      1      400
R[1]  1.0"
    PointSet "COMMAND_" & CStr(i),"SETASG      401     5000
PR[1]  0"
    PointSet "COMMAND_" & CStr(i),"SETASG     5401     1000
ALM[1]  1.0"
    PointSet "COMMAND_" & CStr(i),"SETASG     6401     1000
ALM[E1]  1.0"
    PointSet "COMMAND_" & CStr(i),"SETASG     7401        50
POS[0]  0"
    PointSet "COMMAND_" & CStr(i),"SETASG     7451        18
PRG[1]  1.0"
    PointSet "COMMAND_" & CStr(i),"SETASG     7501       100
R[C1]@1.10  1.0"
    PointSet "COMMAND_" & CStr(i),"SETASG     7601       100
PR[C1]@1.10  1.0"
    PointSet "COMMAND_" & CStr(i),"SETASG     7701         2
$mcr.$genoverride  1.0"
Next i
```

Here, three points named COMMAND_1, COMMAND_2, and COMMAND_3 are assigned to %G1 of each robot. Setting command strings to these points (executing PointSet in VB Scrip), these commands are written to PLC address %G, and are executed in R-J3.

7.1.5 Running ROBOGUIDE To Monitor The Robot Motion (TOP.CIM)

On the screen TOP.CIM, 3D animation of three robots is displayed by ROBOGUIDE.

ROBOGUIDE is called, and is given the robot motion infomations in the VB Script defined in TOP.CIM.



In OnScreenOpen(), ROBOGUIDE is called, is displayed, and open the robot system layout.

```
If robgraph Is Nothing Then  
    Set robgraph = CreateObject("RoboGuidecim.Core")  
End If
```

First, ROBOGUIDE Object "robgraph" is created.

```
robgraph.show "cimplicity_demo"
```

ROBOGUIDE opens the environment
C:/ROBOGUIDE/cimplicity_demo.

```
robgraph.caption="Cimplicity Demo Layout"
```

The title of ROBOGUIDE windows is set Cimplicity Demo Layout.

```
robgraph.move pleft, ptop, pwidth, pheigh
```

The size of ROBOGUIDE window is changet to (pwidth, pheight).
And the ROBOGUIDE window is moved to (pleft, ptop).
(The values of pwidth, pheight, pleft, and ptop are already given in other lines.)

```
Set objLayout = robgraph.layouts.Item("layout1")  
objLayout.Selected = True
```

ROBOGUIDE opens the robot system layout Layout1.

```
Set objRobot1 = objLayout.PlacedObjects.Item("R-2000i_165F")  
Set objRobot2 = objLayout.PlacedObjects.Item("R-2000i_165CF")  
Set objRobot3 = objLayout.PlacedObjects.Item("R-2000i_165R")
```

The robot objects objRobot1, objRobot2, and objRobot3 is assigned to the three robots, R-2000I_165F, R-2000I_165CF, and R-2000_165R, placed in Layout1.

```
Set objView = robgraph.screens.currentview
objView.X = -26
objView.Y = -17
objView.Z = 77
objView.RX = 1220
objView.RY = 86
objView.RZ = -3250
objView.ScaleV = 18
objView.VGDistance = 250
```

The view point of ROBOGUIDE is changed to the specified parameters.

To get the parameters for the property X, Y, Z, RX, RY, RZ, ScaleV, and VGDistance, click the View button on ROBOGUIDE window. The text file C:/ROBOGUIDE/view.txt is created. In this text file shows the view point parameters.

robgraph.topmost = True

ROBOGUIDE window is always displayed over CIMPLICITY screen.

In OnTimer1(), ROBOGUIDE is given the current robot positions. Here, OnTimer1() is automatically called every 250ms, so the robots displayed in the ROBOGUIDE window move like the real robot motions.

For i = 0 To 5

```
    joint1(i) = PointGet ("CUR_POS_J" & Cstr(i+1))
    joint2(i) = PointGet ("CUR_POS_J" & Cstr(i+1) & "_2")
    joint3(i) = PointGet ("CUR_POS_J" & Cstr(i+1) & "_3")
```

Next i

Joint1(), joint2(), and joint3() is set the current position of three robots.

Here, the points CUR_POS_J1 to CUR_POS_J6 are assigned to the current position of R-2000i/165F in the joint format. CUR_POS_J1_2 to CUR_POS_J6_2 and CUR_POS_J1_3 to CUR_POS_J6_3 are also assigned to the current positions of R-2000i/165CF and R-2000i/165R in the joint format.

```
If Not(objRobot1 Is Nothing) Then
    objRobot1.SetJoint joint1()
```

```
Else
```

```
    Set objRobot1=
    objLayout.PlacedObjects.Item("R-2000i_165F")
End If
```

ROBOGUIDE is given the values in joint1().

As a result, the robot R-2000I_165F in ROBOGUIDE is moved to the given position.

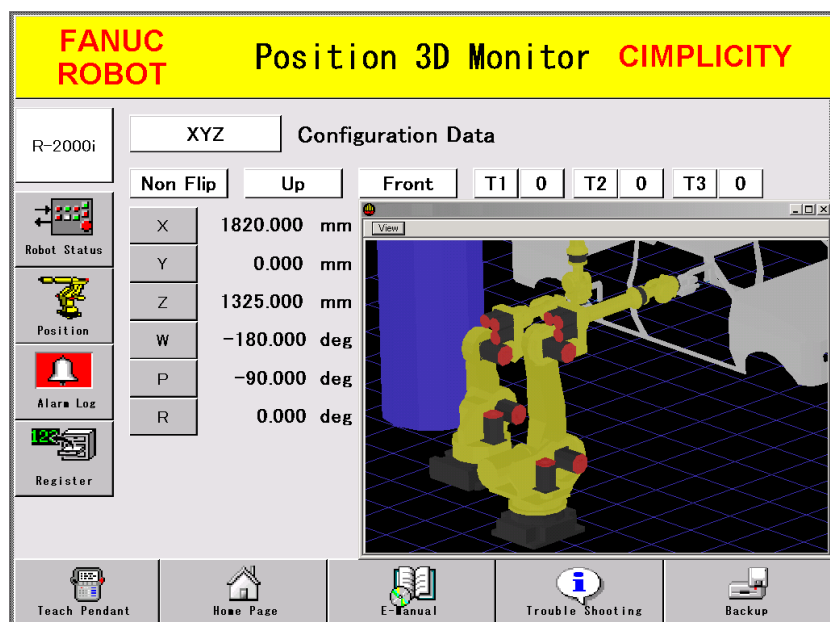
Other two robots are also moved in the same manner with R-2000I_165F.

When TOP.CIM is closed, OnScreenClose1() is called.

robgraph.hide

ROBOGUIDE window is hidden in OnScreenClose1().

On the screen POS1.CIM, POS2.CIM and POS3.CIM, ROBOGUIDE is used in the same manner of TOP.CIM.



7.1.6 Getting The Alarm Information (ALARM1.CIM)

On the screen ALARM1.CIM, the history of alarms occurred in robot controller is displayed.

FANUC ROBOT Current Alarm Logs CIMPLICITY		
R-2000i	Date / Time	Alarm Message
	26 Nov, 11:58:50	FCT-005 \$0B460005-\$00210004 ->
Robot Status	26 Nov, 11:58:46	R E S E T
Position	26 Nov, 11:58:44	SRVO-027 Robot not mastered(Group:1)
Alarm Log	26 Nov, 11:58:44	SRVO-058 FSSB 1 init error
Register	26 Nov, 11:58:44	VARS-006 Unknown Variable Name
	26 Nov, 11:58:44	SYST-046 Control Reliable/GE Mark config mismatch
	26 Nov, 11:58:42	SYST-026 System normal power up
	26 Nov, 11:56:12	FCT-005 \$0B460005-\$00210004 ->
	26 Nov, 11:56:8	R E S E T
	26 Nov, 11:56:8	SRVO-027 Robot not mastered(Group:1)
Teach Pendant Home Page E-Manual Trouble Shooting Backup		

Here, CIMPLICITY can get the information of date and time of occurrence as the values of year, month, day, hour, minute, and second. And these informations are displayed as character string in the format such as "Nov 26, 11:58:50".

The raw data CIMPLICITY can get, in this case, year, month, day, hour, minute, and second, are converted into the character string in OnTimer().

For i=1 To 10

alm_month = Val(PointGet ("ALME_DATE_REAL" & Cstr(i) & "[1]"))

if alm_month = 1 Then alm_date1 = "Jan"

if alm_month = 2 Then alm_date1 = "Feb"

if alm_month = 3 Then alm_date1 = "Mar"

if alm_month = 4 Then alm_date1 = "Apr"

if alm_month = 5 Then alm_date1 = "May"

if alm_month = 6 Then alm_date1 = "Jun"

if alm_month = 7 Then alm_date1 = "Jul"

if alm_month = 8 Then alm_date1 = "Aug"

if alm_month = 9 Then alm_date1 = "Sep"

if alm_month = 10 Then alm_date1 = "Oct"

if alm_month = 11 Then alm_date1 = "Nov"

if alm_month = 12 Then alm_date1 = "Dec"

alm_date2 = PointGet ("ALME_DATE_REAL" & Cstr(i) & "[2]")

alm_date3 = PointGet ("ALME_DATE_REAL" & Cstr(i) & "[3]")

```

    alm_date4 = PointGet ("ALME_DATE_REAL" &
Cstr(i) & "[4]")
    alm_date5 = PointGet ("ALME_DATE_REAL" &
Cstr(i) & "[5]")
    alm_date = alm_date1 + " " + alm_date2 + ", " +
alm_date3 + ":" + alm_date4 + ":" + alm_date5
    PointSet ("ALME_DATE[" & Cstr(i-1) &
"]"),alm_dateNext i

```

Finally, created date/time strings are set to the point ALME_DATE[1 to 10], and these points are displayed on the screen.

And, clicking the alarm message, the detail informations about its alarm is displayed as HTML document.

The diagnostic contents are installed into the local hard drive, and alarm informations are included.

The file path instead of, for example, SRVO-001, is "C:/Diagnostics/alarms/SRVO.htm#SRVO-001".



When the alarm message is clicked, this file path instead of the alarm is created in and given to Web Browser in DispHelp().

```

err_type_no = PointGet("ALME_ID[" & Cstr(alm_no) &
"]")
err_type = ""

```

```

if err_type_no = 1 then err_type = "SRIO"
if err_type_no = 2 then err_type = "FIIE"
if err_type_no = 3 then err_type = "PROG"
if err_type_no = 4 then err_type = "COND"
if err_type_no = 5 then err_type = "ELOG"
if err_type_no = 6 then err_type = "MCTL"
if err_type_no = 7 then err_type = "MEMO"
if err_type_no = 8 then err_type = "GUID"
if err_type_no = 9 then err_type = "TPIF"
if err_type_no = 10 then err_type = "FLPY"
if err_type_no = 11 then err_type = "SRVO"
if err_type_no = 12 then err_type = "INTP"
if err_type_no = 13 then err_type = "PRIO"

```


if err_type_no = 15 then err_type = "MOTN"
if err_type_no = 16 then err_type = "VARS"
if err_type_no = 17 then err_type = "ROUT"
if err_type_no = 18 then err_type = "WNDW"
if err_type_no = 19 then err_type = "JOG"
if err_type_no = 20 then err_type = "APPL"
if err_type_no = 21 then err_type = "LANG"
if err_type_no = 23 then err_type = "SPOT"
if err_type_no = 24 then err_type = "SYST"
if err_type_no = 25 then err_type = "SCIO"
if err_type_no = 26 then err_type = "PALT"
if err_type_no = 28 then err_type = "PMON"
if err_type_no = 29 then err_type = "TOOL"
if err_type_no = 31 then err_type = "PWD"
if err_type_no = 32 then err_type = "VISN"
if err_type_no = 33 then err_type = "DICT"
if err_type_no = 34 then err_type = "KCIL"
if err_type_no = 36 then err_type = "TKSP"
if err_type_no = 37 then err_type = "COPT"
if err_type_no = 38 then err_type = "APSH"
if err_type_no = 42 then err_type = "CMND"
if err_type_no = 43 then err_type = "RPM"
if err_type_no = 44 then err_type = "LNTK"
if err_type_no = 45 then err_type = "WEAV"
if err_type_no = 46 then err_type = "TCPP"
if err_type_no = 47 then err_type = "TAST"
if err_type_no = 48 then err_type = "MUPS"
if err_type_no = 49 then err_type = "MIGE"
if err_type_no = 50 then err_type = "LSR"
if err_type_no = 51 then err_type = "SEAL"
if err_type_no = 53 then err_type = "ARC"
if err_type_no = 54 then err_type = "TRAK"
if err_type_no = 55 then err_type = "CMCC"
if err_type_no = 56 then err_type = "SP"
if err_type_no = 57 then err_type = "MACR"
if err_type_no = 58 then err_type = "SENS"
if err_type_no = 59 then err_type = "COMP"
if err_type_no = 60 then err_type = "THSR"
if err_type_no = 61 then err_type = "QMGR"
if err_type_no = 64 then err_type = "DJOG"
if err_type_no = 66 then err_type = "HRTL"
if err_type_no = 67 then err_type = "HOST"
if err_type_no = 68 then err_type = "MENT"
if err_type_no = 69 then err_type = "SSPC"
if err_type_no = 70 then err_type = "FIG"
if err_type_no = 71 then err_type = "HIGH"
if err_type_no = 72 then err_type = "DX"
if err_type_no = 73 then err_type = "CNTR"
if err_type_no = 75 then err_type = "PFMS"
if err_type_no = 76 then err_type = "DNET"
if err_type_no = 82 then err_type = "CD"

```

if err_type_no = 84 then err_type = "DMDR"
if err_type_no = 85 then err_type = "FRSY"
if err_type_no = 87 then err_type = "FLEX"
if err_type_no = 88 then err_type = "IB-S"
if err_type_no = 89 then err_type = "RTCP"
if err_type_no = 90 then err_type = "TG"
if err_type_no = 92 then err_type = "PROF"
if err_type_no = 93 then err_type = "RPC"
if err_type_no = 95 then err_type = "TSDT"
if err_type_no = 99 then err_type = "ELSE"

```

```

err_no = PointGet("ALME_NO[" & Cstr(alm_no) & "]")
err_no_str = PointGet("ALME_MSG[" & Cstr(alm_no) &
"]")
err_num = right$("000" & Cstr(err_no), 3)

```

```

If (err_no_str <> "") And (err_no_str <> "R E S E T") And
(err_type <> "") Then
    fname = "C:/Diagnostics/alarms/" & err_type &
".htm#" & err_type & "-" & err_num
    Set IE = CreateObject("InternetExplorer.Application")
    IE.Navigate fname
    IE.Visible = True
    Set IE = Nothing
End If

```

When the alarm is not RESET, the variable fname is set the file path created from Alarm ID and Alarm Number, and is given to the Internet Explorer object.

If you hope to put the HTML contents on the Web Server, create the URL string from Alarm ID and Alarm Number such as, for example, "http://192.168.0.1/Diagnostics/alarms/SRVO.htm#SRVO-001"

INDEX

<Symbol>

\$\$SNPX_ASG Is Set, but Data That Should Have Been Assigned Cannot Be Read/Written	50
\$\$SNPX_ASG Settings (em_init.bcl)	57

<A>

ADDRESS ASSIGNMENT TO POINTS	21
APPENDIX	53

<C>

CABLING AND CONNECTION	7
CAUTIONS (BE SURE TO READ THE FOLLOWING:)6	
CHECKING NETWORK-RELATED SETTINGS	11
CREATING A NEW PROJECT	13

<E>

ENVIRONMENT	2
-------------------	---

<G>

Getting The Alarm Information (ALARM1.CIM).....	61
---	----

</>

Improving Communication Efficiency	51
--	----

<M>

Multi Connection And Multiplex \$\$SNPX_ASG	52
---	----

<N>

NETWORK	5
NETWORK-RELATED SETTINGS	9
NOTES AND TIPS ON USAGE	50

<O>

OPERATOR SAFETY	s-4
Operator Safety	s-7

<P>

Precautions for Mechanism	s-12
Precautions for Mechanisms	s-13
Precautions in Operation	s-13
Precautions in Programming	s-12
Precautions in Programming	s-13
Precautions in Programming	s-14
PREFACE	1

<R>

READING ALARM HISTORY (%R)	36
----------------------------------	----

READING AND WRITING I/O SIGNALS

(%I, %Q, %M, %AI, %AQ)	22
------------------------------	----

READING AND WRITING THE COMMENT OF

REGISTERS, POSITION REGISTERS, AND I/O (%R)44	
---	--

READING AND WRITING THE CURRENT POSITION

(%R)	34
------------	----

READING AND WRITING THE VALUE AND SIM

STATUS OF I/O (%R)	46
--------------------------	----

READING FROM AND WRITING INTO SYSTEM

VARIABLES (%R)	41
----------------------	----

READING FROM AND WRITING TO POSITION

REGISTERS (%R)	27
----------------------	----

READING FROM AND WRITING TO REGISTERS

(%R)	24
------------	----

READING THE PROGRAM EXECUTION STATUS

(%R)	39
------------	----

Registered Devices	54
--------------------------	----

Registered Points	54
-------------------------	----

REGISTERING A PORT	15
--------------------------	----

REGISTERING DEVICES	17
---------------------------	----

REGISTERING POINTS	19
--------------------------	----

REQUIRED SOFTWARE	3
-------------------------	---

RESTRICTION ON USE WITH OTHER OPTIONAL

FUNCTIONS	4
-----------------	---

Running ROBOGUIDE To Monitor The Robot Motion

(TOP.CIM)	57
-----------------	----

<S>

Safety During Maintenance	s-11
---------------------------------	------

SAFETY IN MAINTENANCE	s-14
-----------------------------	------

SAFETY OF THE END EFFECTOR	s-14
----------------------------------	------

SAFETY OF THE ROBOT MECHANISM	s-13
-------------------------------------	------

Safety of the Teach Pendant Operator	s-8
--	-----

SAFETY OF THE TOOLS AND PERIPHERAL

DEVICES	s-12
---------------	------

SAFETY PRECAUTIONS	s-3
--------------------------	-----

SAMPLE PROJECT OF “HMI FOR ROBOT”	54
---	----

Sample Project Overview	54
-------------------------------	----

SETTING \$\$SNPX_ASG FROM CIMPPLICITY (%G)....	48
--	----

Setting Full-Duplex Mode (on the Hub Side)	10
--	----

Setting Full-Duplex Mode (on the Robot Side)	10
--	----

Setting the Host Names, Internet (IP) Addresses, and
Subnet Mask..... 9

SETTING UP CIMPPLICITY HMI..... 12

SETTINGS ON THE ROBOT SIDE 8

<V>

Version Of Communication Function With
CIMPPLICITY 51

<W>

WARNING LABEL..... s-15

Revision Record

FANUC Robot series (R-J3/R-J3iB/R-30iA CONTROLLER) CIMPPLICITY HMI for Robots
OPERATOR'S MANUAL (B-82604EN)

01	Feb., 2007								
Edition	Date	Contents	Edition	Date	Contents	Edition	Date	Contents	Contents

