# DENSO ROBOT

Vertical articulated

**V✱-D/-E/-F/-G SERIES**

Horizontal articulated

**H✱-D/-E/-G SERIES**

Cartesian coordinate

**XYC-4D SERIES**

# BEGINNER'S GUIDE

# Preface

Thank you for purchasing this high-speed, high-accuracy assembly robot.

Before operating your robot, read this manual carefully to safely get the maximum benefit from your robot in your assembling operations.

In this manual, one typical robot model is represented in illustrations and captured screen. Operations are described mainly using the teach pendant. For the mini-pendant operation, refer to the SETTING-UP MANUAL, Chapter 6 "Using the Mini-Pendant."

## Robot series covered by this manual

| | |
|---|---|
| - Vertical articulated robot | V∗-D/-E/-F/-G series |
| - Horizontal articulated robot | H∗-D/-E/-G series |
| - Cartesian coordinate robot | XYC-4D series |

## Important

To ensure operator safety, be sure to read the precautions and instructions in "SAFETY PRECAUTIONS," pages 1 through 9.

# How the documentation set is organized

The documentation set consists of the following books. If you are unfamiliar with this robot and option(s), please read all books and understand them fully before operating your robot and option(s).

**GENERAL INFORMATION ABOUT ROBOT**

Provides the packing list of the robot and outlines of the robot system, robot unit, and robot controller.

**INSTALLATION & MAINTENANCE GUIDE**

Provides instructions for installing the robot components and customizing your robot, and maintenance & inspection procedures.

**BEGINNER'S GUIDE - this book -**

Introduces you to the DENSO robot. Taking an equipment setup example, this book guides you through running your robot with the teach pendant, making a program in WINCAPSII, and running your robot automatically.

**SETTING-UP MANUAL**

Describes how to set-up or teach your robot with the teach pendant, operating panel, or mini-pendant.

**WINCAPSII GUIDE**

Provides instructions on how to use the teaching system WINCAPSII which runs on the PC connected to the robot controller for developing and managing programs.

**PROGRAMMER'S MANUAL (I), (II)**

Describes the PAC programming language, program development, and command specifications in PAC.

**CONTROLLER**
**INTERFACE MANUAL**

Describes the robot controller, interfacing with external devices, system- and user-input/output signals, and I/O circuits.

**ERROR CODE TABLES**

List error codes that will appear on the teach pendant, operating panel, or PC screen if an error occurs in the robot series or WINCAPSII. These tables provide detailed description and recovery ways.

**OPTIONS MANUAL**

Describes the specifications, installation, and use of optional devices.

# How this book is organized

This book is just one part of the documentation set. This book consists of SAFETY PRECAUTIONS, parts one through five, and appendices.

### SAFETY PRECAUTIONS

Defines safety terms, safety related symbols and provides precautions that should be observed. Be sure to read this section before operating your robot.

### Part 1 Running the Robot with the Teach Pendant

Describes how to run the robot with the teach pendant in manual mode, how to create a simple program with the teach pendant, and how to teach the robot.

### Part 2 Creating a Program on a PC in WINCAPSII

Provides instructions for setting up WINCAPSII on a PC, creating and compiling a program, and uploading the compiled program to the robot controller.

It also describes machine lock which is required for simulations to be performed in Part 3.

### Part 3 Simulating the Robot Motion on a PC according to the Program

Describes how to check the programmed operation by using the simulator on a PC.

### Part 4 Running the Robot Using Programs

Provides procedures for running your robot actually according to programs and describes palletizing which is one of the main applications on 4-axis robots. It also describes how to make use of PAC libraries which greatly improve the efficiency of task program development.

### Part 5 Features of Denso Robots

Describes the current limit function, direct teaching function, and other features of the Denso robot.

### Appendices

Appendix-1 Glossary
Appendix-2 Names of the robot controller parts
Appendix-3 Names of the teach pendant parts
Appendix-4 Menu tree on the teach pendant

# SAFETY PRECAUTIONS

Be sure to observe all of the following safety precautions.

Strict observance of these warning and caution indications are a MUST for preventing accidents, which could result in bodily injury and substantial property damage. Make sure you fully understand all definitions of these terms and related symbols given below, before you proceed to the text itself.

| | |
|---|---|
| ⚠ **WARNING** | Alerts you to those conditions, which could result in serious bodily injury or death if the instructions are not followed correctly. |
| ⚠ **CAUTION** | Alerts you to those conditions, which could result in minor bodily injury or substantial property damage if the instructions are not followed correctly. |

## Terminology and Definitions

**Maximum space:** Refers to the volume of space encompassing the maximum designed movements of all robot parts including the end-effector, workpiece and attachments. (Quoted from the RIA* Committee Draft.)

**Restricted space:** Refers to the portion of the maximum space to which a robot is restricted by limiting devices (i.e., mechanical stops). The maximum distance that the robot, end-effector, and workpiece can travel after the limiting device is actuated defines the boundaries of the restricted space of the robot. (Quoted from the RIA Committee Draft.)

**Motion space:** Refers to the portion of the restricted space to which a robot is restricted by software motion limits. The maximum distance that the robot, end-effector, and workpiece can travel after the software motion limits are set defines the boundaries of the motion space of the robot. (The "motion space" is Denso-proprietary terminology.)

**Operating space:** Refers to the portion of the restricted space (or motion space in Denso) that is actually used by the robot while performing its task program. (Quoted from the RIA Committee Draft.)

**Task program:** Refers to a set of instructions for motion and auxiliary functions that define the specific intended task of the robot system. (Quoted from the RIA Committee Draft.)

(*RIA: Robotic Industries Association)

# 1. Introduction

This section provides safety precautions to be observed during installation, teaching, inspection, adjustment, and maintenance of the robot.

The installation shall be made by qualified personal and should confirm to all national and local code.

# 2. Installation Precautions

## 2.1 Insuring the proper installation environment

### 2.1.1 For standard type

The standard type has not been designed to withstand explosions, dust-proof, nor is it splash-proof. Therefore, it should not be installed in any environment where:

(1) there are flammable gases or liquids,

(2) there are any shavings from metal processing or other conductive material flying about,

(3) there are any acidic, alkaline or other corrosive gases,

(4) there is cutting or grinding oil mist,

(5) it may likely be submerged in fluid,

(6) there is sulfuric cutting or grinding oil mist, or

(7) there are any large-sized inverters, high output/high frequency transmitters, large contactors, welders, or other sources of electrical noise.

### 2.1.2 For dust-proof, splash-proof type

The dust-proof, splash-proof type is an IP54-equivalent dust-proof and splash-proof structure, but it has not been designed to withstand explosions.

Note that the robot controller is not a dust- or splash-proof structure. Therefore, when using the robot controller in an environment exposed to mist, put it in an optional protective box.

The dust-proof, splash-proof type should not be installed in any environment where:

(1) there are any flammable gases or liquids,

(2) there are any acidic, alkaline or other corrosive gases,

(3) there are any large-sized inverters, high output/high frequency transmitters, large contactors, welders, or other sources of electrical noise,

(4) it may likely be submerged in fluid,

(5) there are any grinding or machining chips or shavings,

(6) any machining oil other than DENSO authorized oil is in use, or

Note: DENSO authorized oil: Yushiron Oil No. 4C (non-soluble)

(7) there is sulfuric cutting or grinding oil mist.

## 2.2 Service space

The robot and peripheral equipment should be installed so that sufficient service space is maintained for safe teaching, maintenance, and inspection.

## 2.3 Control devices outside the robot's restricted space

The robot controller, teach pendant, and operating panel should be installed outside the robot's restricted space and in a place where you can observe all of the robot’s movements when operating the robot controller, teach pendant, or operating panel.

## 2.4 Positioning of gauges

Pressure gauges, oil pressure gauges and other gauges should be installed in an easy-to-check location.

## 2.5 Protection of electrical wiring and hydraulic/pneumatic piping

If there is any possibility of the electrical wiring or hydraulic/pneumatic piping being damaged, protect them with a cover or similar item.

## 2.6 Positioning of emergency stop switches

Emergency stop switches should be provided in a position where they can be reached easily should it be necessary to stop the robot immediately.

(1) The emergency stop switches should be red.

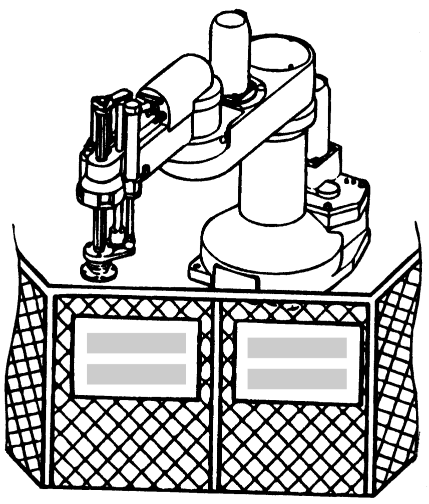
(2) Emergency stop switches should be designed so that they will not be released after pressed, automatically or mistakenly by any other person.

(3) Emergency stop switches should be separate from the power switch.

## 2.7 Positioning of operating status indicators

Operating status indicators should be positioned in such a way where workers can easily see whether the robot is on temporary halt or on an emergency or abnormal stop.

## 2.8 Setting-up the safety fence or enclosure



A safety fence or enclosure should be set up so that no one can easily enter the robot's restricted space. If it is impossible, utilize other protectors as described in Section 2.9.

(1) The fence or enclosure should be constructed so that it cannot be easily moved or removed.

(2) The fence or enclosure should be constructed so that it cannot be easily damaged or deformed through external force.

(3) Establish the exit/entrance to the fence or enclosure. Construct the fence or enclosure so that no one can easily get past it by climbing over the fence or enclosure.

(4) The fence or enclosure should be constructed to ensure that it is not possible for hands or any other parts of the body to get through it.

(5) Take any one of the following protections for the entrance/ exit of the fence or enclosure:

1) Place a door, rope or chain across the entrance/exit of the fence or enclosure, and fit it with an interlock that ensures the emergency stop device operates automatically if it is opened or removed.

2) Post a warning notice at the entrance/exit of the fence or enclosure stating "In operation--Entry forbidden" or "Work in progress--Do not operate" and ensure that workers follow these instructions at all times.

When making a test run, before setting up the fence or enclosure, place an overseer in a position outside the robot’s restricted space and one in which he/she can see all of the robot’s movements. The overseer should prevent workers from entering the robot's restricted space and be devoted solely to that task.

## 2.9 Positioning of rope or chain

If it is not possible to set up the safety fence or enclosure described in Section 2.8, hang a rope or chain around the perimeter of the robot’s restricted space to ensure that no one can enter the restricted space.

(1) Ensure the support posts cannot be moved easily.

(2) Ensure that the rope or chain’s color or material can easily be discerned from the surrounds.

(3) Post a warning notice in a position where it is easy to see stating "In operation--Entry forbidden" or "Work in progress --Do not operate" and ensure that workers follow these instructions at all times.

(4) Set the exit/entrance, and follow the instructions given in Section 2.8, (3) through (5).

## 2.10 Setting the robot's motion space

The area required for the robot to work is called the robot's operating space.

If the robot's motion space is greater than the operating space, it is recommended that you set a smaller motion space to prevent the robot from interfering or disrupting other equipment.

Refer to the "INSTALLATION & MAINTENANCE GUIDE" Chapter 4.

## 2.11 No robot modification allowed

Never modify the robot unit, robot controller, teach pendant or other devices.

## 2.12 Cleaning of tools

If your robot uses welding guns, paint spray nozzles, or other end-effectors requiring cleaning, it is recommended that the cleaning process be carried out automatically.

## 2.13 Lighting

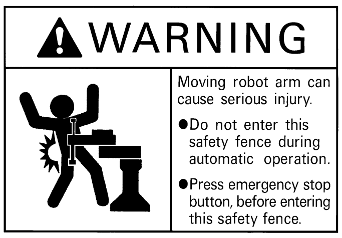
Sufficient illumination should be assured for safe robot operation.

## 2.14 Protection from objects thrown by the end-effector

If there is any risk of workers being injured in the event that the object being held by the end-effector is dropped or thrown by the end-effector, consider the size, weight, temperature and chemical nature of the object and take appropriate safeguards to ensure safety.

## 2.15 Affixing the warning label

Place the warning label packaged with the robot on the exit/entrance of the safety fence or in a position where it is easy to see.



WARNING

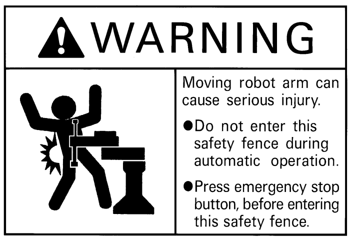
Moving robot arm can cause serious injury.

- Do not enter this safety fence during automatic operation.
- Press emergency stop button, before entering this safety fence.

## 3. Precautions while robot is running

⚠ **Warning**

Touching the robot while it is in operation can lead to serious injury. Please ensure the following conditions are maintained and that the cautions listed from Section 3.1 onwards are followed when any work is being performed.



⚠WARNING
Moving robot arm can cause serious injury.
●Do not enter this safety fence during automatic operation.
●Press emergency stop button, before entering this safety fence.

1) Do not enter the robot's restricted space when the robot is in operation or when the motor power is on.
2) As a precaution against malfunction, ensure that an emergency stop device is activated to cut the power to the robot motor upon entry into the robot's restricted space.
3) When it is necessary to enter the robot's restricted space to perform teaching or maintenance work while the robot is running, ensure that the steps described in Section 3.3 "Ensuring safety of workers performing jobs within the robot's restricted space” are taken.

### 3.1 Creation of working regulations and assuring worker adherence

When entering the robot’s restricted space to perform teaching or maintenance inspections, set “working regulations” for the following items and ensure workers adhere to them.

(1) Operating procedures required to run the robot.

(2) Robot speed when performing teaching.

(3) Signaling methods to be used when more than one worker is to perform work.

(4) Steps that must be taken by the worker in the event of a malfunction, according to the contents of the malfunction.

(5) The necessary steps for checking release and safety of the malfunction status, in order to restart the robot after robot movement has been stopped due to activation of the emergency stop device

(6) Apart from the above, any steps below necessary to prevent danger from unexpected robot movement or malfunction of the robot.

  1) Display of the control panel (See Section 3.2 on the following page)

  2) Assuring the safety of workers performing jobs within the robot's restricted space (See Section 3.3 on the following page)

  3) Maintaining worker position and stance

  Position and stance that enables the worker to confirm normal robot operation and to take immediate refuge if a malfunction occurs.

4) Implementation of measures for noise prevention
5) Signaling methods for workers of related equipment
6) Types of malfunctions and how to distinguish them

Please ensure “working regulations” are appropriate to the robot type, the place of installation and to the content of the work.

Be sure to consult the opinions of related workers, engineers at the equipment manufacturer and that of a labor safety consultant when creating these “working regulations”.

## 3.2 Display of operation panel

To prevent anyone other than the worker from accessing the start switch or the changeover switch by accident during operation, display something to indicate it is in operation on the operating panel or teach pendant. Take any other steps as appropriate, such as locking the cover.

## 3.3 Ensuring safety of workers performing jobs within the robot's restricted space

When performing jobs within the robot’s restricted space, take any of the following steps to ensure that robot operation can be stopped immediately upon a malfunction.

(1) Ensure an overseer is placed in a position outside the robot’s restricted space and one in which he/she can see all robot movements, and that he/she is devoted solely to that task.

① An emergency stop device should be activated immediately upon a malfunction.

② Do not permit anyone other than the worker engaged for that job to enter the robot’s restricted space.

(2) Ensure a worker within the robot's restricted space carries the portable emergency stop switch so he/she can press it (the robot stop button on the teach pendant) immediately if it should be necessary to do so.

## 3.4 Inspections before commencing work such as teaching

Before starting work such as teaching, inspect the following items, carry out any repairs immediately upon detection of a malfunction and perform any other necessary measures.

(1) Check for any damage to the sheath or cover of the external wiring or to the external devices.

(2) Check that the robot is functioning normally or not (any unusual noise or vibration during operation).

(3) Check the functioning of the emergency stop device.

(4) Check there is no leakage of air or oil from any pipes.

(5) Check there are no obstructive objects in or near the robot’s restricted space.

## 3.5 Release of residual air pressure

Before disassembling or replacing pneumatic parts, first release any residual air pressure in the drive cylinder.

## 3.6 Precautions for test runs

Whenever possible, have the worker stay outside of the robot's restricted space when performing test runs.

## 3.7 Precautions for automatic operation

(1) At start-up

Before the robot is to be started up, first check the following items as well as setting the signals to be used and perform signaling practice with all related workers.

1) Check that there is no one inside the robot’s restricted space.

2) Check that the teach pendant and tools are in their designated places.

3) Check that no lamps indicating a malfunction on the robot or related equipment are lit.

(2) Check that the display lamp indicating automatic operation is lit during automatic operation.

(3) Steps to be taken when a malfunction occurs

Should a malfunction occur with the robot or related equipment and it is necessary to enter the robot's restricted space to perform emergency maintenance, stop the robot’s operation by activating the emergency stop device. Take any necessary steps such as placing a display on the starter switch to indicate work is in progress to prevent anyone from accessing the robot.

### 3.8 Precautions in repairs

(1) Do not perform repairs outside of the designated range.

(2) Under no circumstances should the interlock mechanism be removed.

(3) When opening the robot controller's cover for battery replacement or any other reasons, always turn the robot controller power off and disconnect the power cable.

(4) Use only spare tools authorized by DENSO.

## 4. Daily and periodical inspections

(1) Be sure to perform daily and periodical inspections. Before starting jobs, always check that there is no problem with the robot and related equipment. If any problems are found, take any necessary measures to correct them.

(2) When carrying out periodical inspections or any repairs, maintain records and keep them for at least 3 years.

## 5. Management of floppy disks

(1) Carefully handle and store the "Initial settings" floppy disks packaged with the robot, which store special data exclusively prepared for your robot.

(2) After finishing teaching or making any changes, always save the programs and data onto floppy disks.

Making back-ups will help you recover if data stored in the robot controller is lost due to the expired life of the back-up battery.

(3) Write the names of each of the floppy disks used for storing task programs to prevent incorrect disks from loading into the robot controller.

(4) Store the floppy disks where they will not be exposed to dust, humidity and magnetic field, which could corrupt the disks or data stored on them.

# CONTENTS

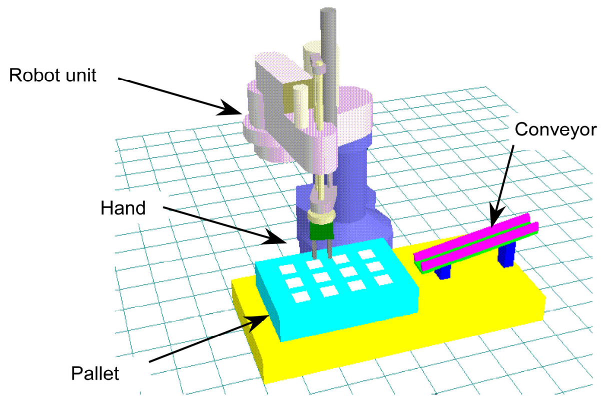
# Part 1
# Running the Robot with the Teach Pendant

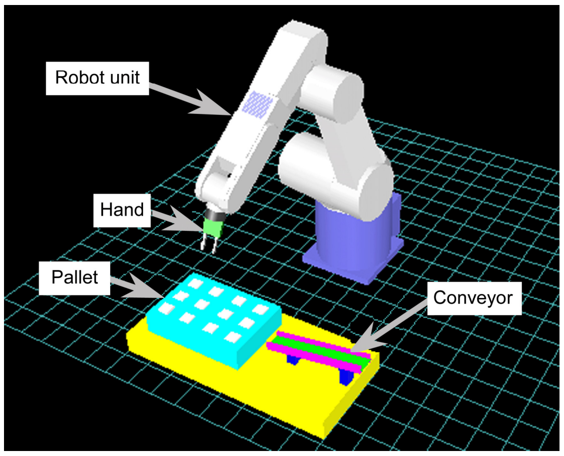
In Part 1, you will:

- Learn how to handle and operate the teach pendant.
- Practice the following with the teach pendant:
  - Performing safe and precise manual operation (in Joint, X-Y, and Tool modes)
  - Calibrating (CAL) the robot (For **-D series only)
  - Creating and editing programs
  - Performing safe teaching
  - Making a safe teach check
  - Starting up and stopping programs safely

## Equipment Setup Example

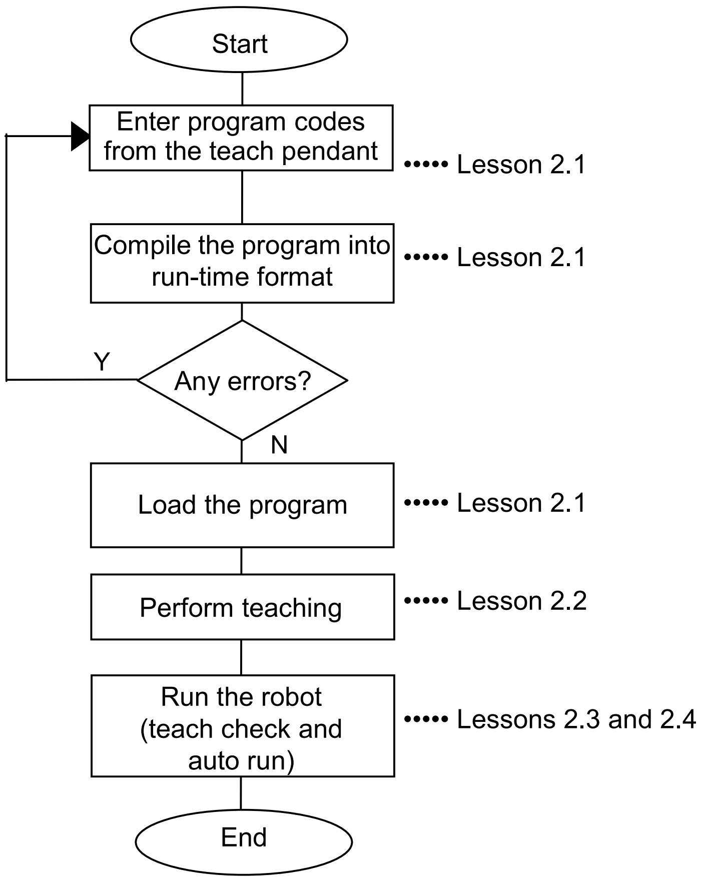
The figure below shows an example of equipment setup with the robot included as part of the production line. This robot performs palletizing operations.





## Process flow from program creation to checking of robot motion

Shown below is the process flow starting with program creation and continuing as far as checking of the robot motion.


Start
Enter program codes from the teach pendant
••••• Lesson 2.1
Compile the program into run-time format
••••• Lesson 2.1
Any errors?
Y
N
Load the program
••••• Lesson 2.1
Perform teaching
••••• Lesson 2.2
Run the robot (teach check and auto run)
••••• Lessons 2.3 and 2.4
End

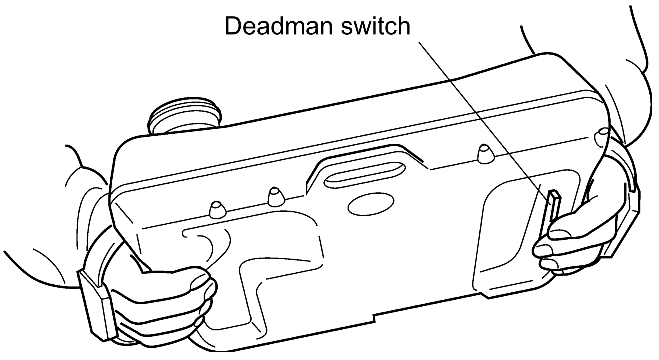
# Lesson 1 Running the Robot in Manual Mode

## 1.1 Basic teach pendant operations

**Holding the teach pendant and the deadman switch**

When operating the teach pendant, grasp it as shown below.

The teach pendant has two deadman switches, so it is possible to hold the teach pendant in the following 2 ways:



**Holding the teach pendant**

★**Tip**★ The deadman switch is provided to stop the robot automatically and safely when the operator can no longer operate the robot correctly due to unforeseen circumstances such as the operator suffering a blackout or dying while running the robot manually with the teach pendant. If a situation such as this arises, the strength with which the operator is pressing the deadman switch will either decrease or increase markedly. The deadman switch is a 3-position switch which is able to recognize and react to the following 3 operating statuses:

1) When the switch is not being pressed or is being pressed lightly
→ Switch: OFF
2) When the switch is being pressed with correct pressure
→ Switch: ON
3) When the switch is being pressed too strongly
→ Switch: OFF

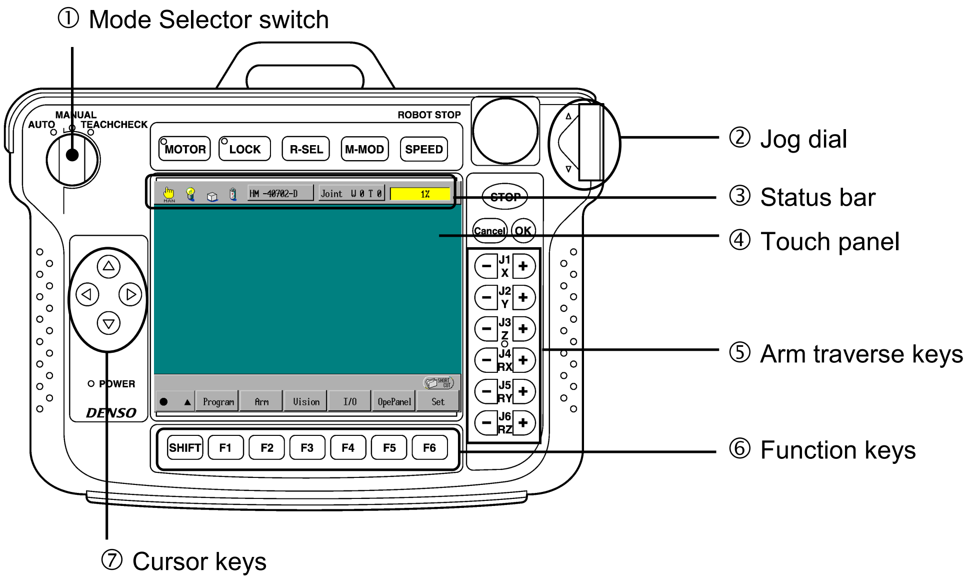
Unless the switch is ON, the robot cannot run nor is it possible to drive the robot.

In order to ensure safety, the robot is designed so that in manual mode the deadman switch should be held down, for example, when the operator presses any of the arm traverse keys.

**Tip:** A deadman switch is called "enable switch."

## Basic make-up of the teach pendant

When the controller power is turned ON, the top screen shown below appears on the teach pendant.



**Teach pendant – top screen**

① Mode selector switch
This switches operation modes between Auto, Manual and Teach check modes.

**Note:** On the RC7M teach pendant, this switch is a keylock type. Pulling out the key locks the selector switch in Manual and Teach check modes. In Auto mode, the key cannot be pulled out.

② Jog dial
This makes adjusting values easier.

③ Status bar
This always displays the current operation mode and robot status.

④ Touch panel
The LCD screen of the teach pendant is also a touch panel. By touching the buttons or data entry areas displayed on the screen, it is possible to perform operations and make selections.

**Caution:** Touch the LCD screen with your fingers only, never with the tip of a pen or any pointed object. Otherwise, the LCD will be damaged.

⑤ Arm traverse keys
These keys drive the robot arm manually in a designated direction. It is also necessary to hold down the deadman switch at the same time.

⑥ Function keys
F1 to F6 are normally displayed on the screen. This can be switched to display F7 to F12 when required by pressing the SHIFT key.

⑦ Cursor keys
These are used to move the cursor on the display screen and entry screen.

Refer to Appendix 3 for details on each section of the teach pendant.

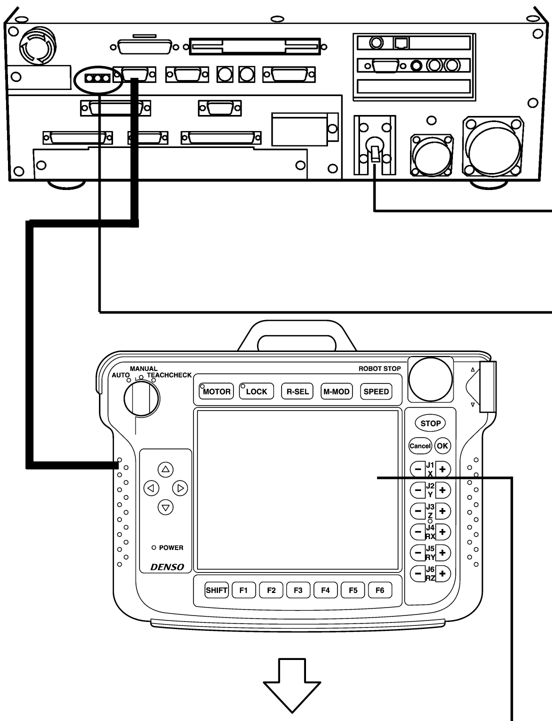
## 1.2 Running the robot manually with the teach pendant

First of all, you will practice turning the robot controller and motor ON and running the robot manually with the teach pendant.

**Step 1** **<u>Checking that it is safe to proceed</u>**

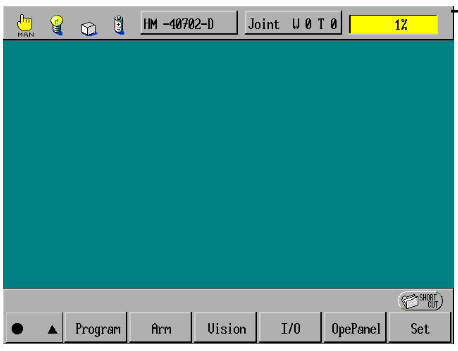
- Check that the robot is installed correctly.
- Check that there is no one within the robot's restricted space.

**Step 2** **<u>Turning the robot controller ON</u>**



① Flip the controller power switch upward.

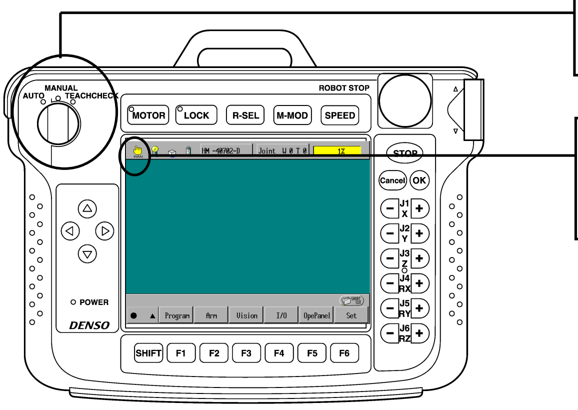
The power lamp (furthest left one of the 3 pilot lamps) will light and the remaining 2 lamps will flash momentarily.



The top screen will appear on the teach pendant soon after.

**Step 3**

## Placing the robot in Manual mode



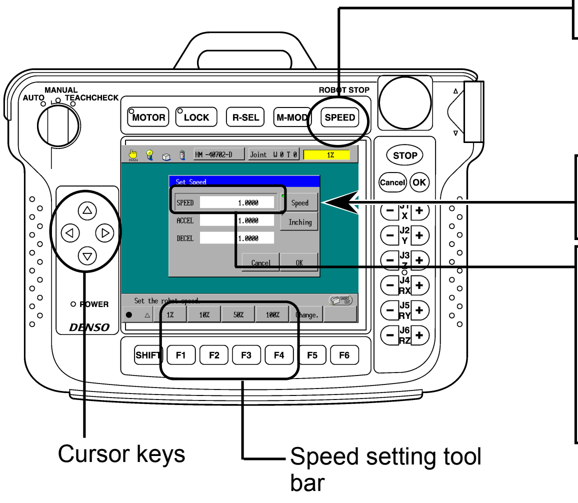
① Set the Mode Selector switch to MANUAL.

In the leftmost area of the status bar, an icon indicating Manual mode will be displayed.

**Step 4**

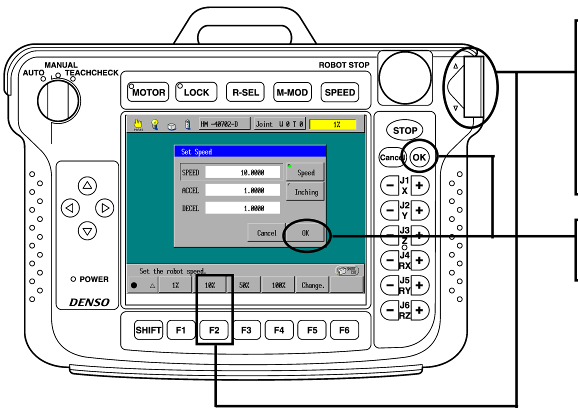
## Setting the speed and acceleration

① Press [SPEED].



The [Set Speed] window is displayed.

The SPEED box should be selected, however if either the ACCEL or DECEL box has been selected, use the UP and DOWN cursor keys to select the SPEED box.

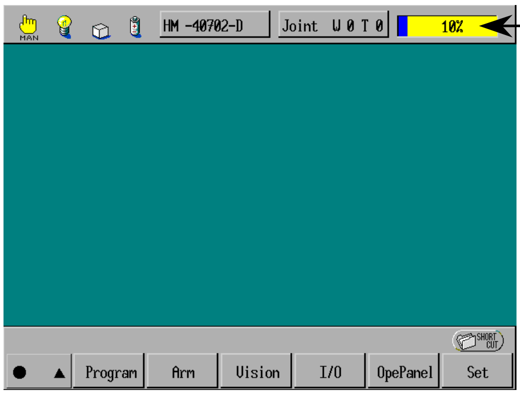


② Press [F2 10%]. (The SPEED value can also be changed with the Jog dial.)

(SPEED will be set at 10% and ACCEL and DECEL at 1%.)

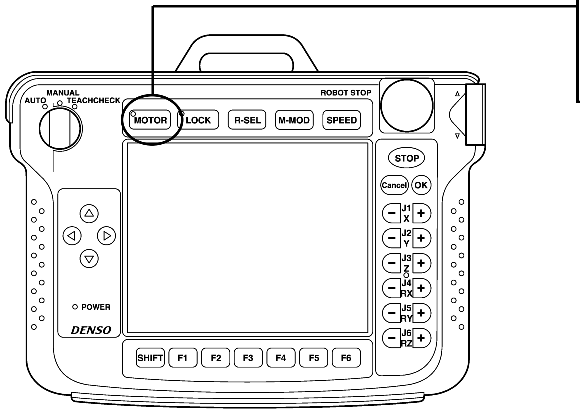
③ Press [OK].

★**Remarks**★ At the beginning, leave these settings as they are, as you will be running the robot slowly to ensure safety. The settings can be changed later on, after you have become accustomed to running the robot with the teach pendant.



The SPEED display will become 10%.

## Step 5 Turning the motor ON

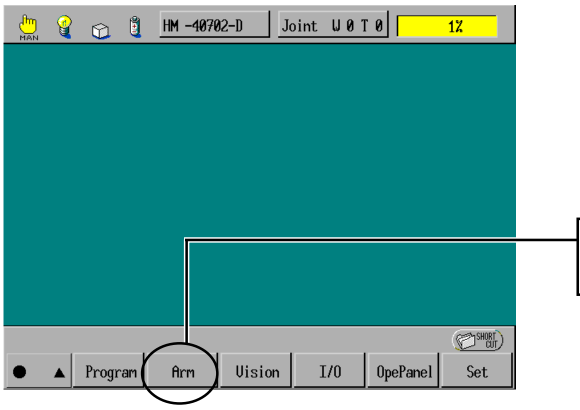


① Press [MOTOR].
The power to the motor and the [MOTOR] lamp come on.

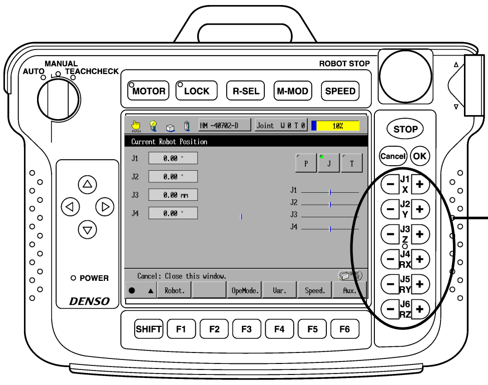
## Step 6 Moving each arm of the robot manually

**Caution**

When this operation is performed, the robot arm will move. Any workers should leave the robot's restricted space.



① Press [F2 Arm].



② While observing the robot, press the deadman switch and the arm traverse keys.

The arm corresponding with the operation of the J1 to J4 (4-axis robot) or J1 to J6 (6-axis robot) arm traverse keys will move. In the Current Robot Position window the angle of each axis will be displayed.

★**Caution**★ For the **-E/-F/-G series and VM-6083D/60B1D robots, skip Step 7 since no CAL is required. Only the **-D series requires CAL to run the robot using accurate values.

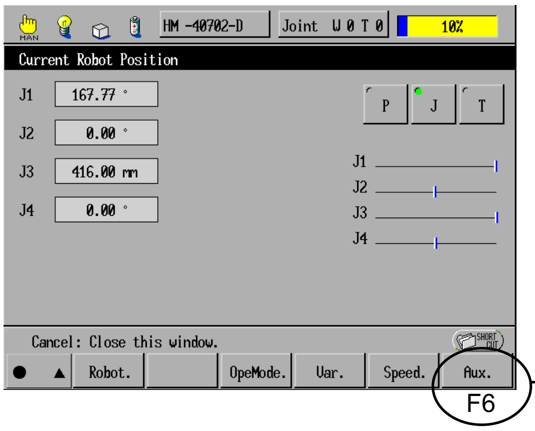
**Step 7**

## Performing CAL (calibration) (for **-D series only)

CAL stands for calibration, which actuates all robot axes to move the robot arm in small motions in order to confirm the current arm position after the controller power is turned ON.

The CAL procedure is described below.

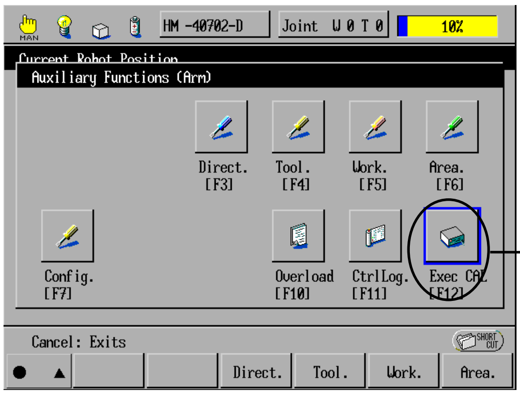
**Caution** Performing CAL will move the robot arm. Before proceeding, be sure that all workers have left the robot's restricted space and that there are no obstacles in the robot's restricted space.



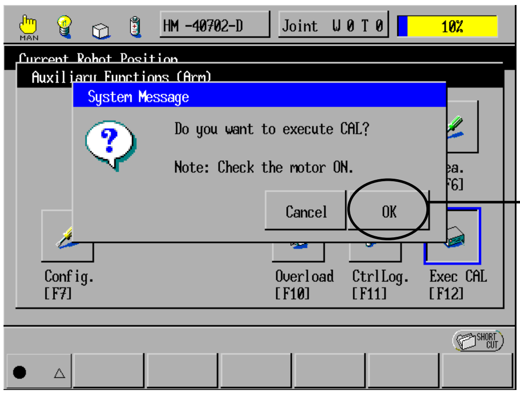
Current Robot Position

| | |
|---|---|
| J1 | 167.77 ° |
| J2 | 0.00 ° |
| J3 | 416.00 mm |
| J4 | 0.00 ° |

① Press [F6 Aux.] with the [Current Robot Position] window displayed.



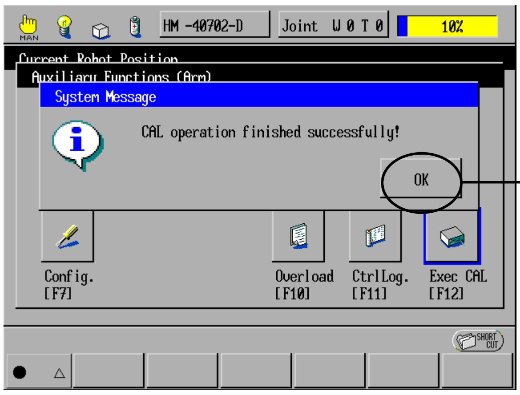
② Press [F12 Exec CAL].



Do you want to execute CAL?

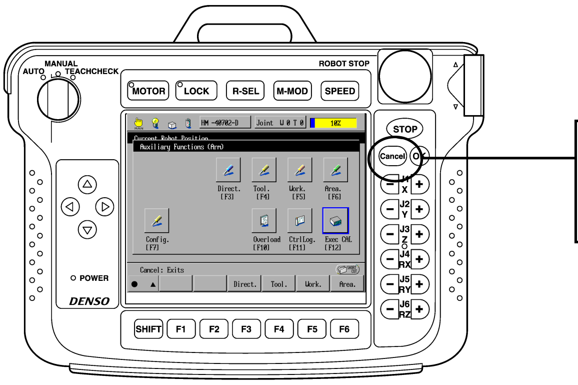
Note: Check the motor ON.

③ Check the System Message and press [OK].



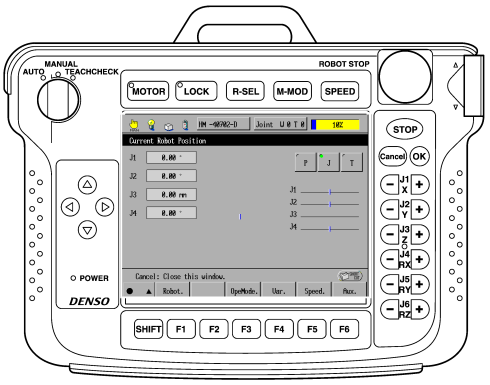
CAL operation finished successfully!

④ Check that the System Message reads "CAL operation finished successfully!" and press [OK].

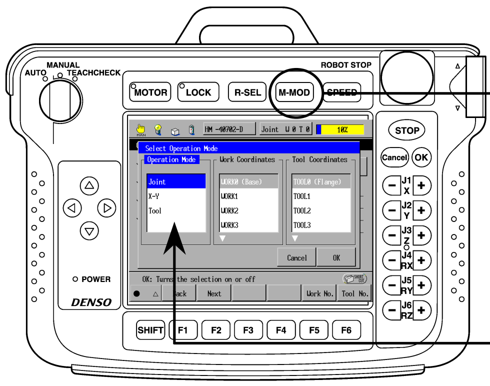


⑤ After returning to the [Auxiliary Functions (Arm)] window, press [Cancel].

The [Current Robot Position] window will be displayed again. At this point, CAL is completed and it is possible to run the robot.



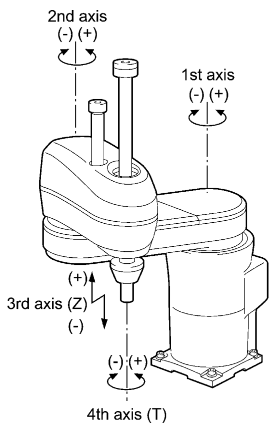
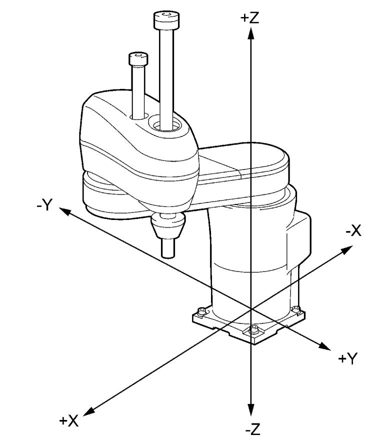
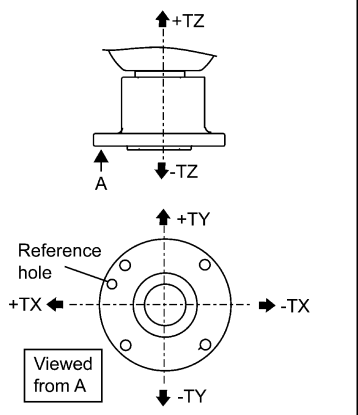
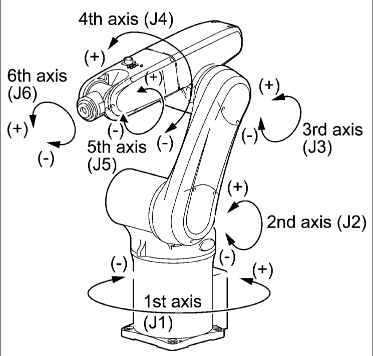
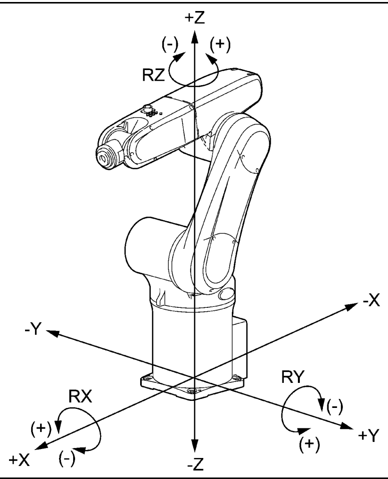
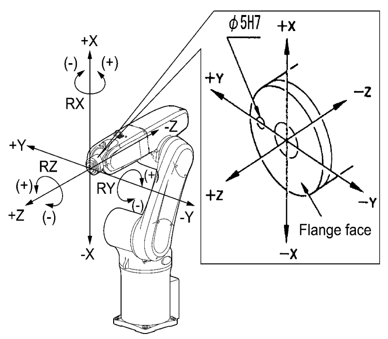
**Step 8** **Selecting Manual mode and running the robot manually**



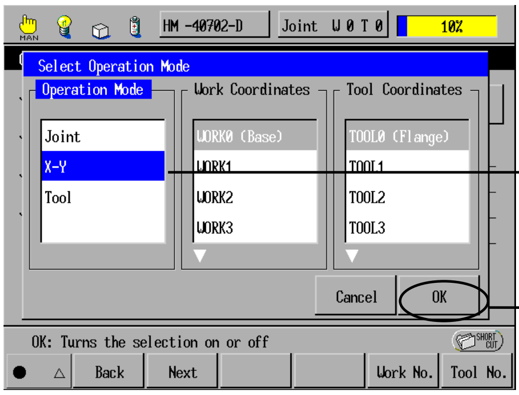
① Press [M-MOD].

The [Operation Mode] window is displayed.

**★Point★** In Manual mode you may select any of these three modes: Joint mode, X-Y mode and Tool mode.

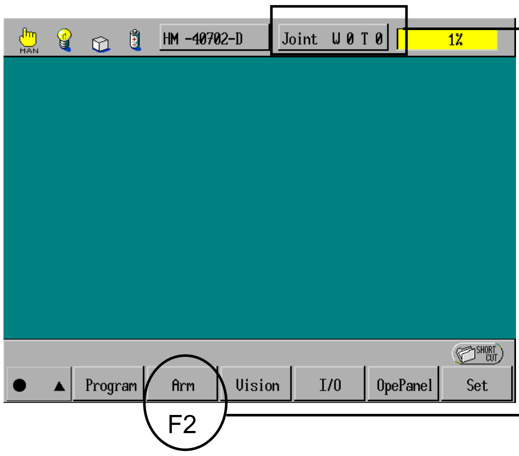
| | <Joint mode > | <X-Y mode> | <Tool mode> |
|---|---|---|---|
| Action | Drives each of the joints independently. | Drives the robot flange linearly in base coordinates. | Drives the robot flange linearly along the work coordinates of the 4th axis. |
| 4-axis robot |  2nd axis (-) (+) 1st axis (-) (+) 3rd axis (Z) (+) (-) (-) (+) 4th axis (T) | The 4th axis will keep the orientation that was held immediately before the robot was driven.  | The 4th axis will hold its orientation.  |
| 6-axis robot |  4th axis (J4) (+) 6th axis (J6) (+) (-) 5th axis (J5) (+) (-) 3rd axis (J3) (+) (-) 2nd axis (J2) (-) (+) 1st axis (J1) |  |  +X (-) (+) RX +Y RZ (+) +Z (-) RY -X -Z -Y ϕ5H7 +X +Y -Z +Z -Y Flange face -X |

In this lesson, you will practice running the robot in X-Y mode.



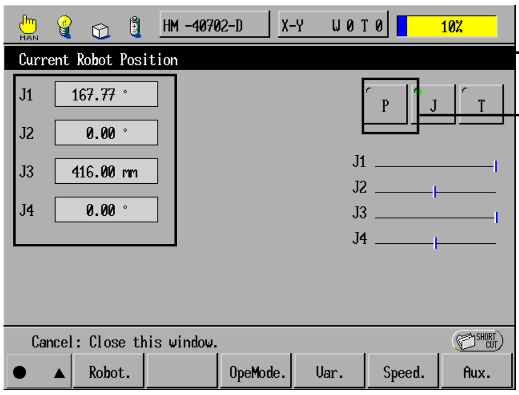
② In the [Select Operation Mode] window, select "X-Y" (use the UP and DOWN cursor keys or the Jog dial).

③ Press [OK].
The top screen will appear where you press [F2 Arm].



X-Y appears on the status bar.

④ Press [F2 Arm].

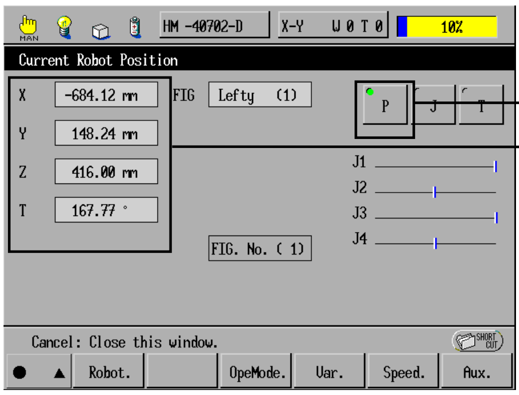


The Current Robot Position window appears.

⑤ Press the P (position variable) button to show the current robot position. You may press the shift key and [F7 Show P] in the menu bar, instead of the P button.

(This is necessary to run the robot in X-Y mode.)

⇩


HM -40702-D | X-Y | U 0 T 0 | 10%

Current Robot Position

X -684.12 mm
Y 148.24 mm
Z 416.00 mm
T 167.77 °

FIG Lefty (1)

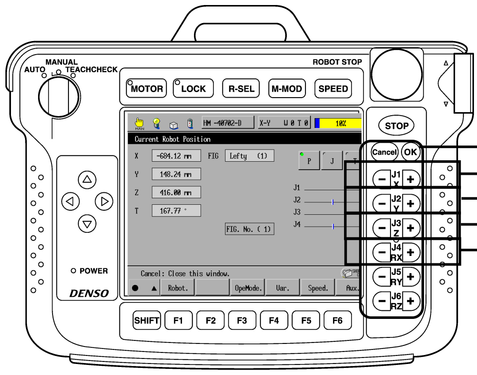
FIG. No. ( 1)

Cancel: Close this window.

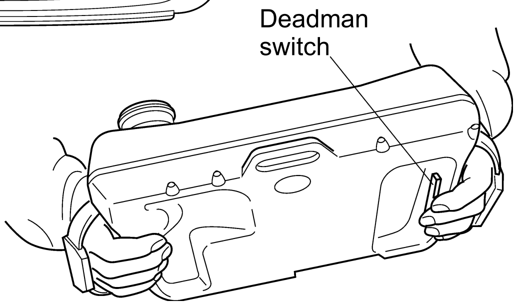
The P lamp comes on and the screen changes to one where the current robot position is expressed in position variables.

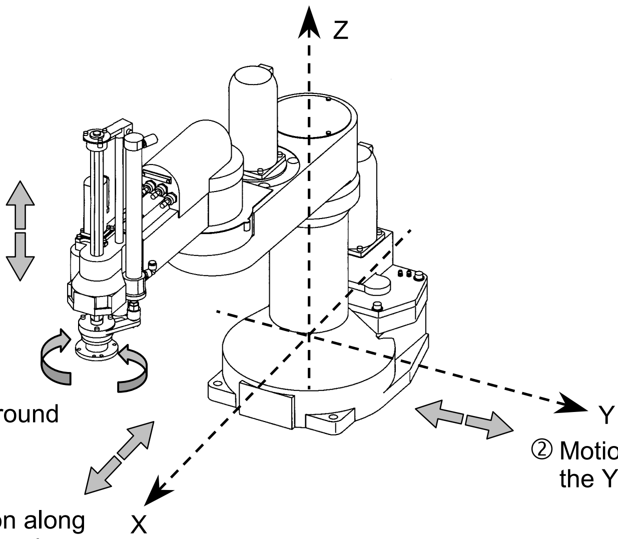
⇩

⑥ Run the robot by pressing the arm traverse keys with the deadman switch held down.



Arm traverse keys
① Motion in X direction
② Motion in Y direction
③ Motion in Z direction
④ Rotation around T-axis





③ Motion along the Z-axis.

④ Rotation around T-axis

② Motion along the Y-axis

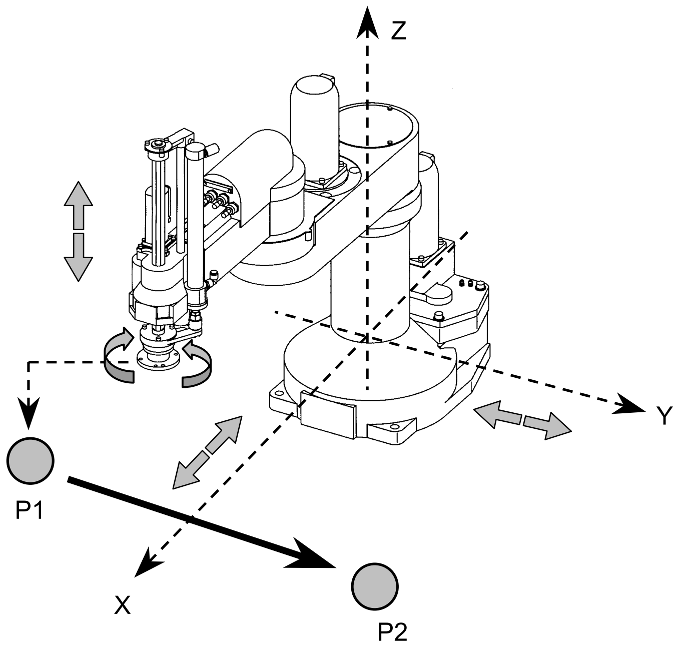
① Motion along the X-axis

# Lesson 2  Running the Robot Using a Simple Program

In order to run the robot in a designated way, it is necessary to create a program and teach the positions you want the robot arm to move to.

In this lesson, you will practice moving the robot arm from P1 to P2 as shown below. This will be described in the following order.

2.1  Creating a simple program from the teach pendant
2.2  Teaching (teaching of P1 and P2)
2.3  Teach check
2.4  Running the robot with automatic operation



## 2.1 Creating a simple program from the teach pendant

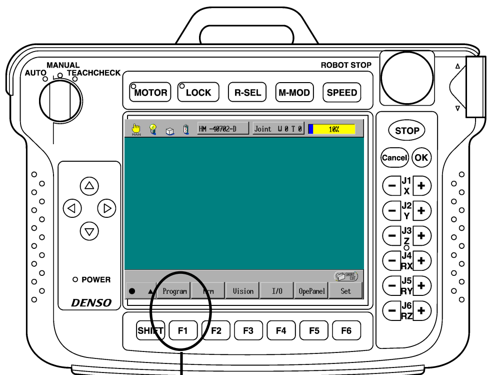
First, you will enter codes of a simple program using the teach pendant.

Creating and editing of programs should be done in the Manual mode. If you take the following procedure immediately after performing the previous lesson, the robot is now in the Manual mode, so proceed as is. If not, you need to place the robot in Manual mode before proceeding.

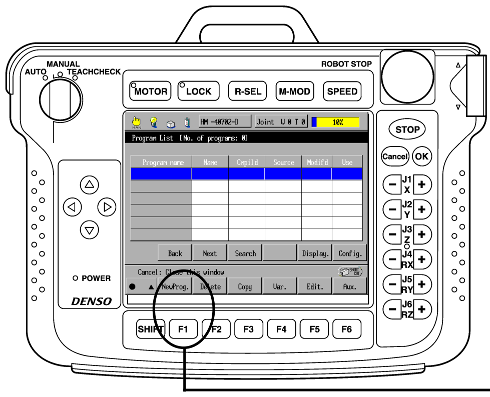
Refer to the PROGRAMMER'S MANUAL for a detailed description on writing programs.

### Step 1 Opening a program edit window

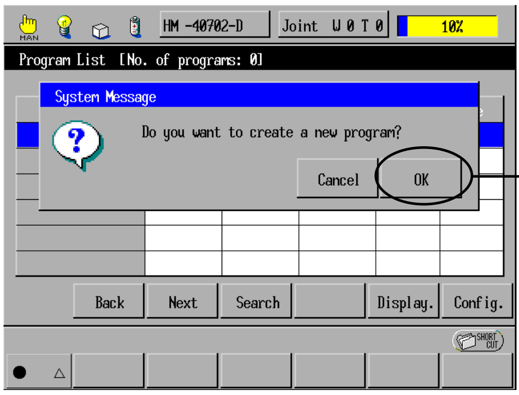
To create a new program, it is necessary to open the window for editing programs on the teach pendant screen.



① Press [F1 Program] on the top screen.



② Press [F1 NewProg.].



HM -40702-D Joint W 0 T 0 10%

Program List [No. of programs: 0]

System Message

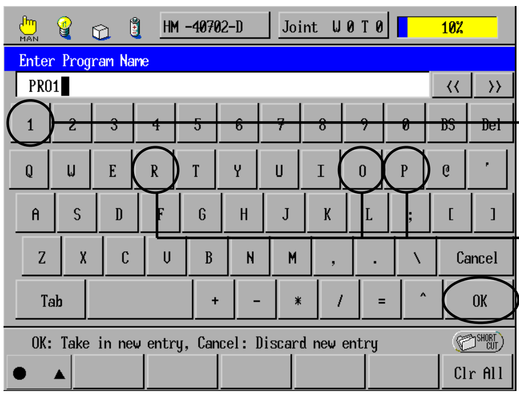
Do you want to create a new program?

Cancel OK

Back Next Search Display. Config.

③ Press [OK].

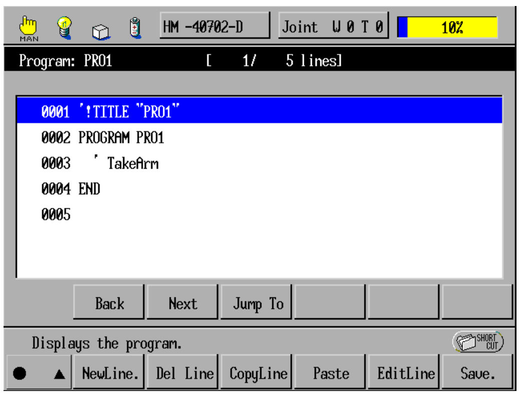
Next, type the file name of the program (here we will use PRO1) to be created.



④ Type PRO1 using the letter and numeric buttons.

⑤ After typing PRO1 correctly, press [OK].

This ends the preparation for program editing.



```
'!TITLE "PRO1"
PROGRAM PRO1
  ' TakeArm
END

```

The preset program codes are displayed.

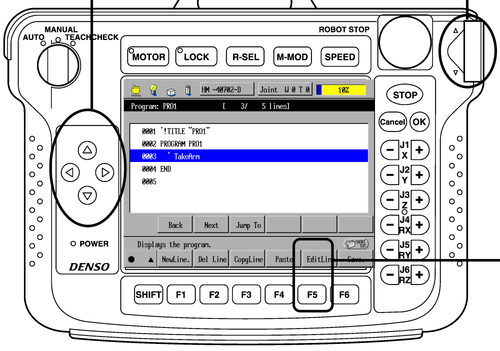
**Step 2**

## Entering program codes

In this step, you will create a program to move from P1 to P2. Enter the program codes listed in the table below.

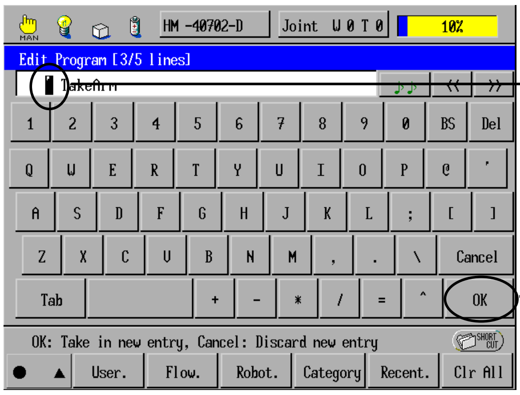
**Coding List for "PRO1"**

```
PROGRAM  PRO1
     TAKEARM          'Acquires the arm semaphore
     SPEED 100        'Specifies internal speed
     MOVE L, P1       'Moves to specified coordinates for P1
     MOVE L, P2       'Moves to specified coordinates for P2
     GIVEARM          'Releases the arm semaphore
END
```



① In the "Program: PRO1" window, move the cursor to the 3rd line using the cursor keys or jog dial.

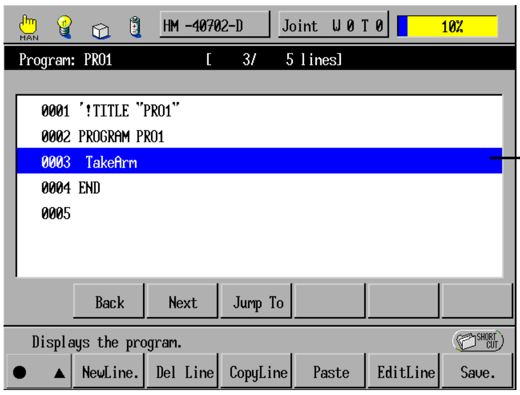
② Press [F5 EditLine].



```
HM -40702-D   Joint  W 0 T 0   10%
Edit Program [3/5 lines]
TakeArm
OK: Take in new entry, Cancel: Discard new entry
User.  Flow.  Robot.  Category  Recent.  Clr All
```

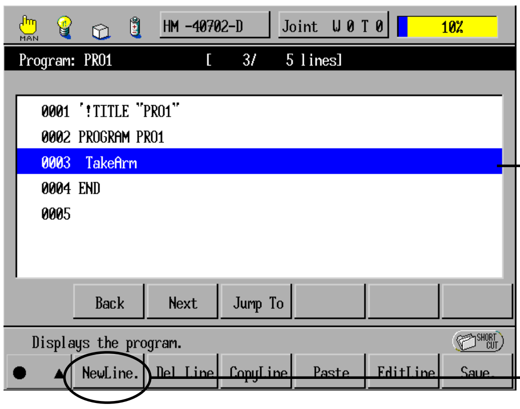
③ Delete the apostrophe (') from the head of the line using the cursor keys and [Del].

④ Press [OK].



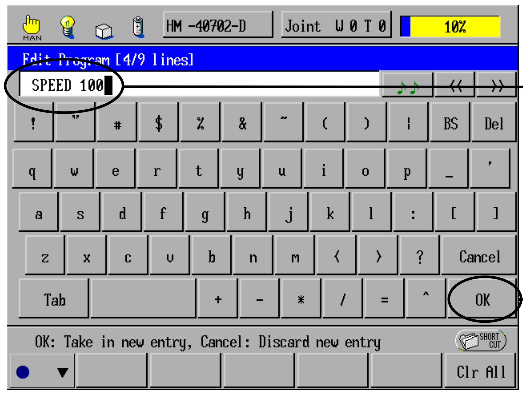
```
HM -40702-D   Joint  W 0 T 0   10%
Program: PRO1          [   3/    5 lines]
0001 '!TITLE "PRO1"
0002 PROGRAM PRO1
0003  TakeArm
0004 END
0005
Displays the program.
```

The screen shows the program edit window [Program: PRO1] again where the 3rd line has been modified.



```
HM -40702-D   Joint  W 0 T 0   10%
Program: PRO1            [   3/    5 lines]
0001 '!TITLE "PRO1"
0002 PROGRAM PRO1
0003  TakeArm
0004 END
0005
Back  Next  Jump To
Displays the program.
NewLine.  Del Line  CopyLine  Paste  EditLine  Save.
```

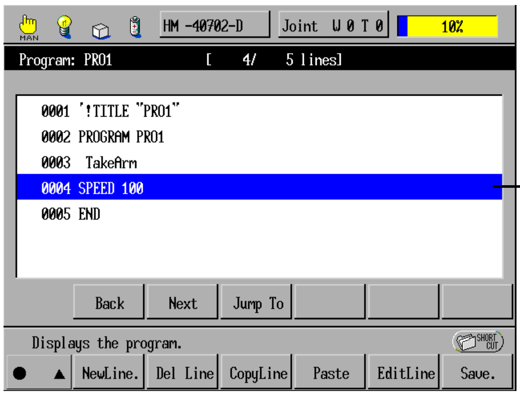
⑤ Move the cursor to the 3rd line and press [F1 NewLine.].



```
HM -40702-D   Joint  W 0 T 0   10%
Edit Program [4/9 lines]
SPEED 100
OK: Take in new entry, Cancel: Discard new entry
Clr All
```

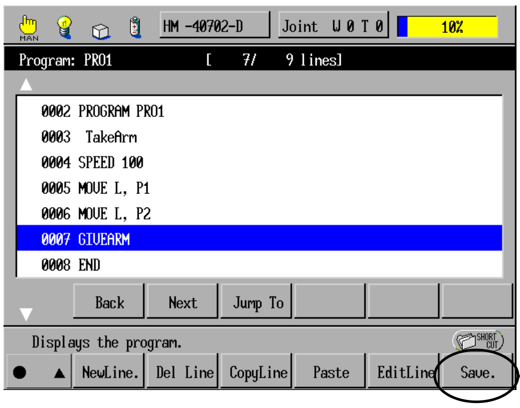
⑥ Enter "SPEED 100" from the keyboard. This is displayed in this window.

⑦ Press [OK].



```
HM -40702-D   Joint  W 0 T 0   10%
Program: PRO1            [   4/    5 lines]
0001 '!TITLE "PRO1"
0002 PROGRAM PRO1
0003  TakeArm
0004 SPEED 100
0005 END
Back  Next  Jump To
Displays the program.
NewLine.  Del Line  CopyLine  Paste  EditLine  Save.
```

The program edit window "Program: PRO1" is displayed and "SPEED 100" is displayed in the 4th line.

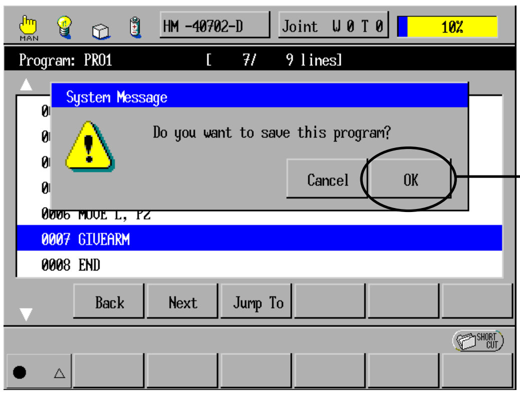


```
Program: PRO1              [   7/    9 lines]
0002 PROGRAM PRO1
0003  TakeArm
0004 SPEED 100
0005 MOVE L, P1
0006 MOVE L, P2
0007 GIVEARM
0008 END
```

Displays the program.

⑧ Enter all of the program codes given on p.16 in the same way used to enter "SPEED 100".

⑨ After completing entry of all codes, press [F6 Save.].



Program: PRO1 [ 7/ 9 lines]

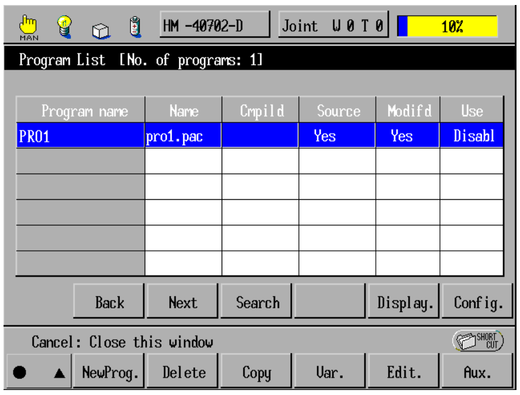
System Message

Do you want to save this program?

```
0006 MOVE L, P2
0007 GIVEARM
0008 END
```

⑩ Press [OK] to save the newly entered program.

The display will return to the Program List window.



Program List [No. of programs: 1]

| Program name | Name | Cmpild | Source | Modifd | Use |
|---|---|---|---|---|---|
| PRO1 | pro1.pac | | Yes | Yes | Disabl |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Cancel: Close this window

**★Caution★**

(1) If you do not want to save the changes made, press [Cancel] instead of [OK] and the display will return to the program edit screen without the changes being saved.

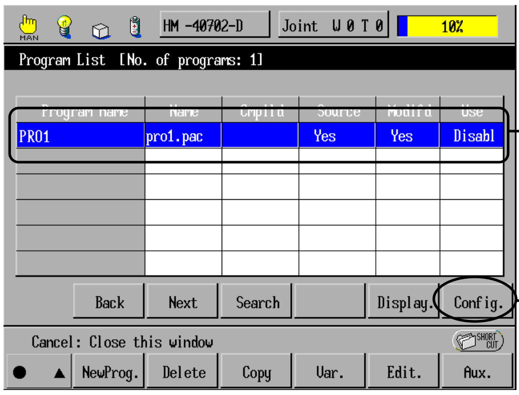
(2) To create a new program, return to Step 1.

**Step 3**

## Compiling the program into run-time format

After editing a program, you need to compile it; that is, transform the edited program into run-time format which is executable by the robot controller.

During compiling, syntax errors will be detected if contained in the edited program. You need to correct all syntax errors since programs containing them cannot be loaded or executed.



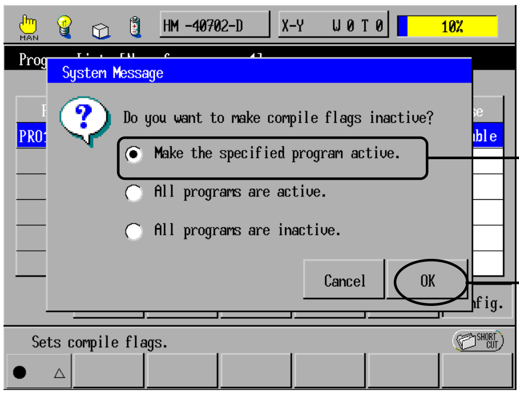
HM -40702-D　Joint　W 0 T 0　10%

Program List [No. of programs: 1]

| Program name | Name | Cmpl'd | Source | Modif'd | Use |
|---|---|---|---|---|---|
| PRO1 | pro1.pac | | Yes | Yes | Disabl |

① Select "PRO1" in the Program List window.
(You may select it by using the cursor keys or jog dial, or by touching the screen directly.)

② Press [F12 Config.].



HM -40702-D　X-Y　W 0 T 0　10%

System Message

Do you want to make compile flags inactive?

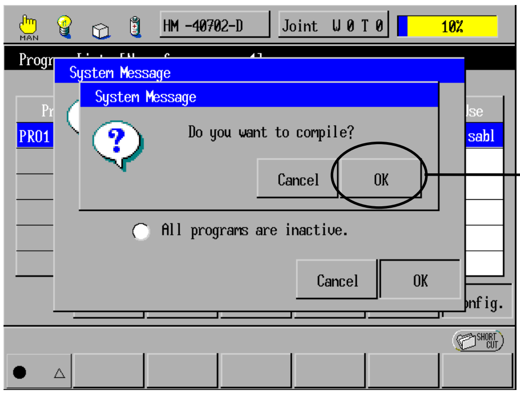
[✓ Make the specified program active.]
All programs are active.
All programs are inactive.

Sets compile flags.

③ Select "Make the specified program active".

④ Press [OK].

★**Caution**★ It is possible to perform the same operation with [Config.] which is located at the bottom right of the Program List window.



HM -40702-D Joint W 0 T 0 10%

System Message

System Message

Do you want to compile?

Cancel OK

All programs are inactive.

Cancel OK

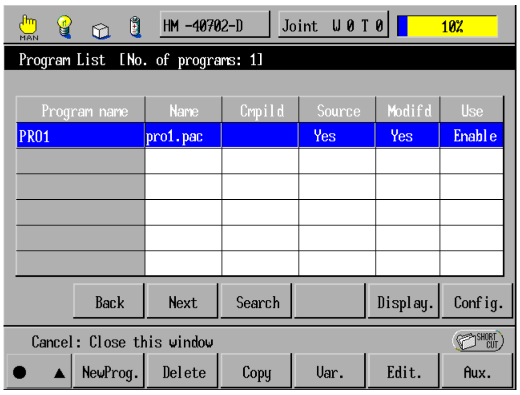
⑤ Press [OK].
Compiling will start.

When compiling is complete, the screen will return to the [Program List] window.

**★Caution★**

(1) If you press [Cancel] instead of [OK] at this point, the screen will return to the [Program List] window without performing the compiling operation.

(2) There is one other way with which you may compile programs into run-time format.
Press [F6 Aux.] in the [Program List] window to call up the [Auxiliary Functions (Programs)] window. In the window, press [F12 Compile]. With this method, you may continue on to load programs after compiling.



HM -40702-D Joint W 0 T 0 10%

Program List [No. of programs: 1]

| Program name | Name | Cmpild | Source | Modifd | Use |
|---|---|---|---|---|---|
| PRO1 | pro1.pac | | Yes | Yes | Enable |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Cancel: Close this window

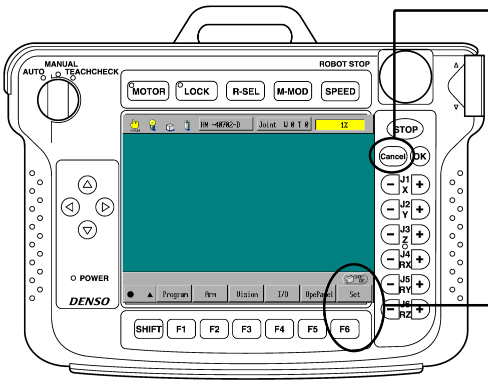
F6

**Step 4**

## Loading the program

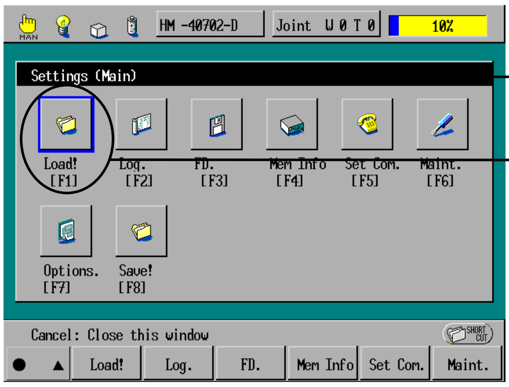
You need to load the compiled program so that the robot controller can execute it.

Even if compiled programs are transferred from the PC connected to the robot controller, they cannot execute. They need to be loaded to the memory area where the program can be executed.



① Display the top screen.
(If any other screen is displayed, press [Cancel] as many times as necessary until the top screen appears.)

② Press [F6 Set] on the top screen.


HM -40702-D Joint U 0 T 0 10%

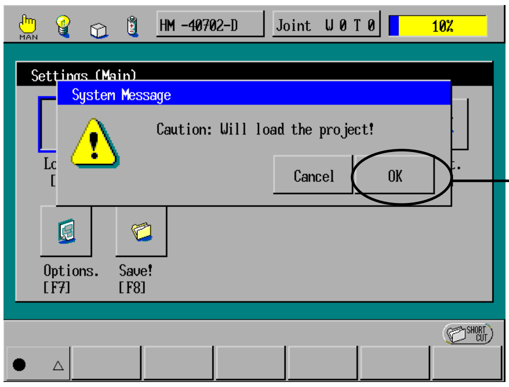
Settings (Main)

| | | | | | |
|---|---|---|---|---|---|
| Load! [F1] | Log. [F2] | FD. [F3] | Mem Info [F4] | Set Com. [F5] | Maint. [F6] |
| Options. [F7] | Save! [F8] | | | | |

Cancel: Close this window

The [Settings (Main)] window appears.

③ Press [F1 Load].

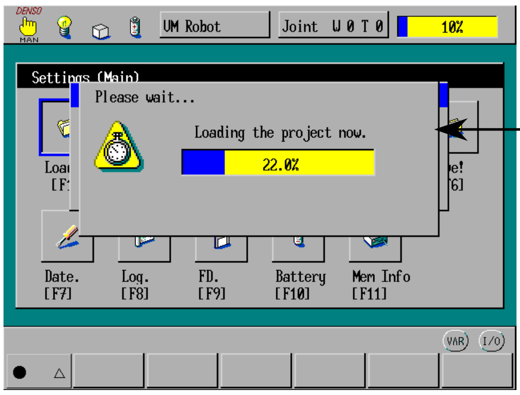

HM -40702-D Joint U 0 T 0 10%

Settings (Main)

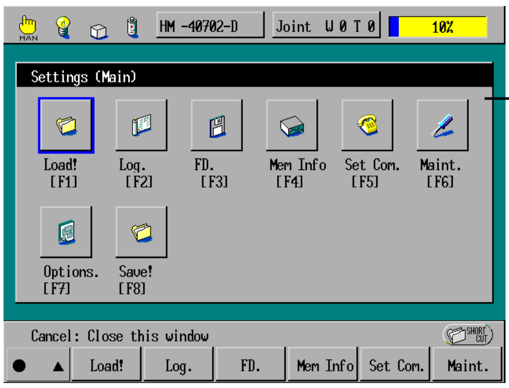
System Message

Caution: Will load the project!

④ Press [OK].



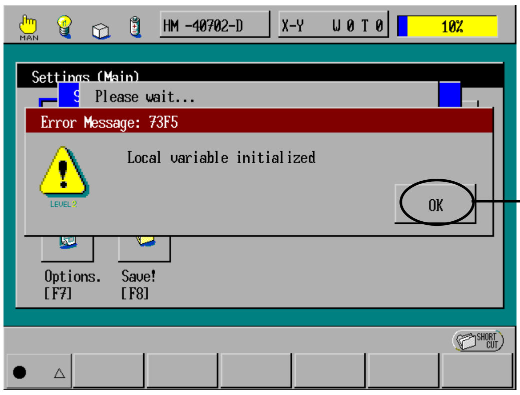
The message "Please wait… Loading the project now." is displayed.



Upon completion of loading, the screen returns to the [Setting (Main)] window.

**★Caution★** If you load a project using local variables different from those used in the previous project, the error message "Local variable initialized" is displayed.

Press [OK] to continue.



⑤ Press [OK].

Now, the program is ready to execute.
Press [Cancel] to return to the top screen.

This completes the creation of the program to run the robot.

## 2.2 Teaching

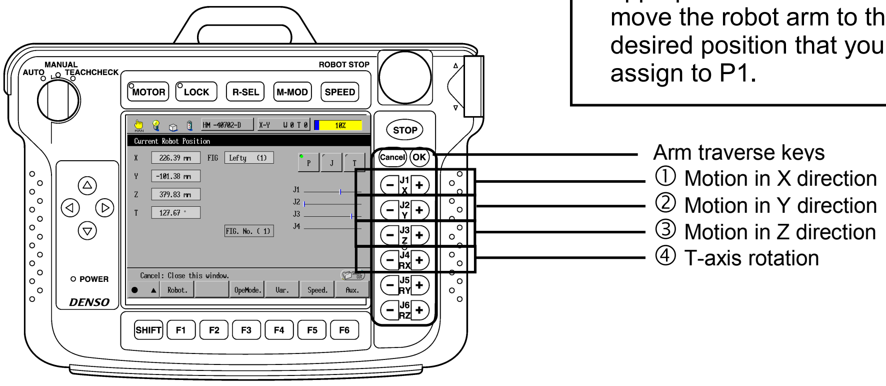
Teaching refers to a method of programming in which you guide a robot through its motions using the teach pendant. In teaching, the robot is taught its motion.

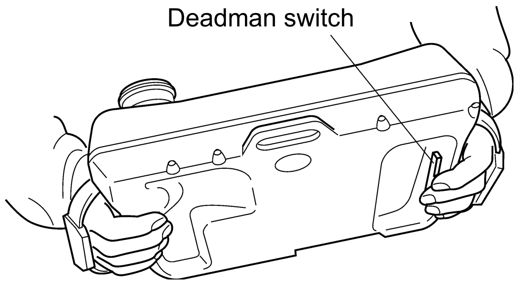
In programming, you may specify positions as constants. However, in order to make the robot accurately learn the relative positional relationship between itself and objective point, you need to move the robot actually on site. Consequently, you write positions as variables in programming and assign actual values to those variables by on-site teaching.

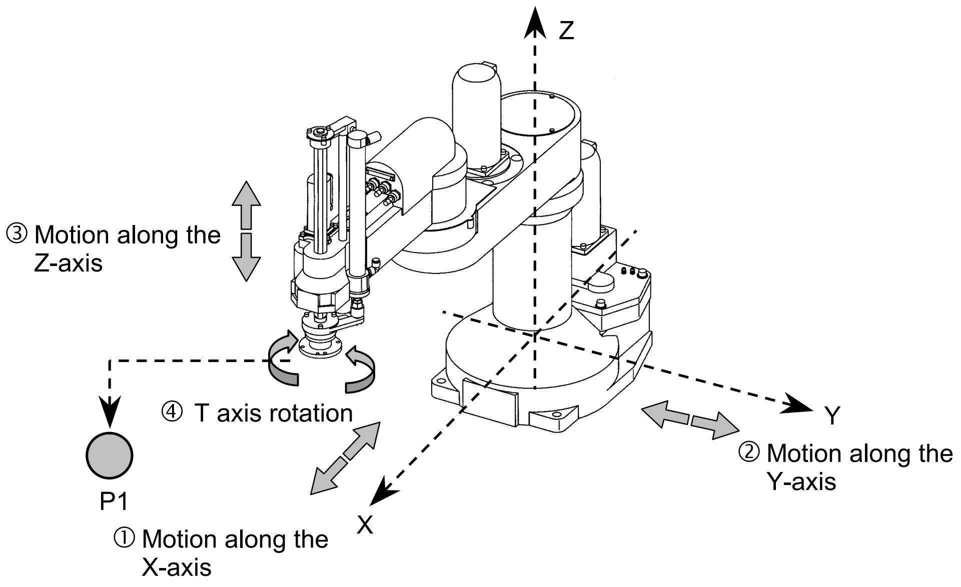
The program created in Lesson 2.1 contains two position variables P1 and P2. This section gives you how to teach the robot values for P1 and P2.

**Step 1** **Teaching the robot position P1**

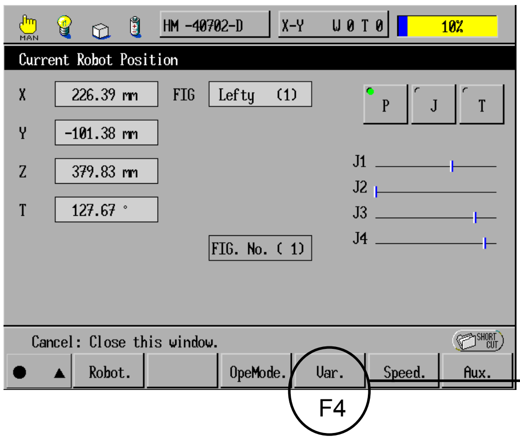
① While holding down the deadman switch, press the appropriate arm traverse keys to move the robot arm to the desired position that you want to assign to P1.



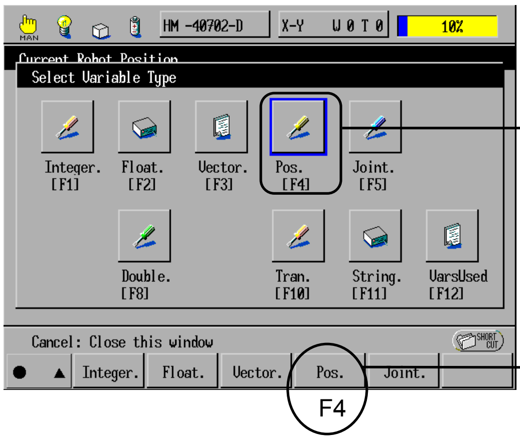



Z
③ Motion along the Z-axis
④ T axis rotation
P1
① Motion along the X-axis
X
Y
② Motion along the Y-axis

## Step 2 Assigning the taught value to [Variable P1]



① Press [F4 Var.].

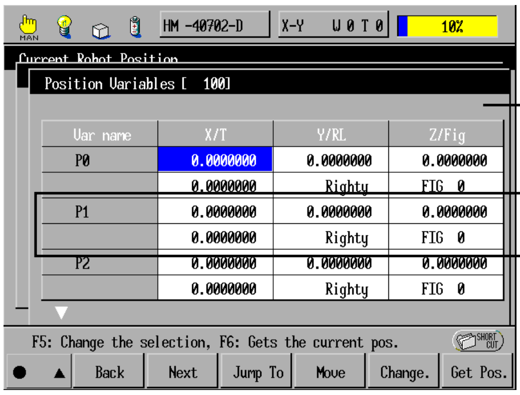


② Select the variable type in the [Select Variable Type] window.
At this point, press [F4 Pos.] to assign a value to a position variable.
(It is also possible to touch [Pos.] in the window.)

**★Tip★**

A variable refers to a program identifier for a storage location which can contain any number or characters and which may vary during the program. The following types of variables are supported:

| | |
|---|---|
| I. (Integer): | Integer variable (range: -2147483648 to +2147483647) |
| F. (Float): | Floating-point variable<br>(range: -3.402823E+383.402823E+38) |
| D. (Double): | Double-precision variable<br>(range: -1.7976931348623157D+308 to 1.7976931348623157D+308) |
| V. (Vector): | Vector variable (X, Y, Z) |
| P. (Pos): | Position variable (X, Y, Z, T, FIG) |
| J. (Joint): | Joint variable (J1, J2, J3, J4) |
| T. (Trans): | Homogeneous transform matrix variable |
| S. (String): | Character string variable (which can contain a character string of up to 247 characters) |



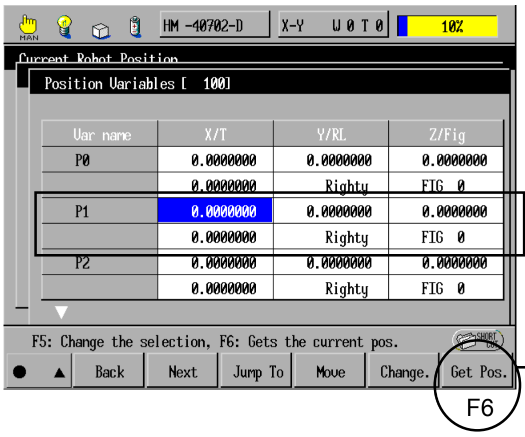
| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P0 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P1 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

F5: Change the selection, F6: Gets the current pos.

The [Position Variables] window appears.

③ Select the [P1] box using the cursor keys or jog dial.

The [Position Variables] window shows five types of data for each variable name. If you select and highlight any one of them, for example, any in the [Var name P1] box, then it means that the [Var name P1] is selected.

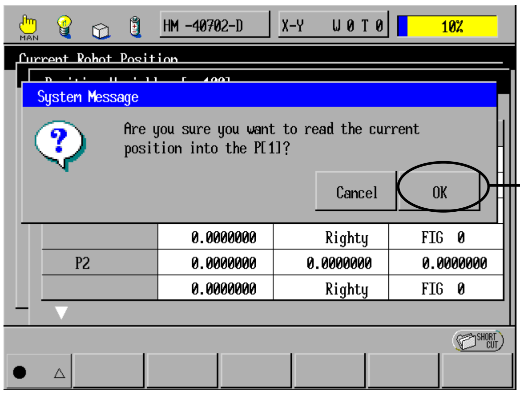


| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P0 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P1 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

F5: Change the selection, F6: Gets the current pos.

④ Check that the [Var name P1] is selected.

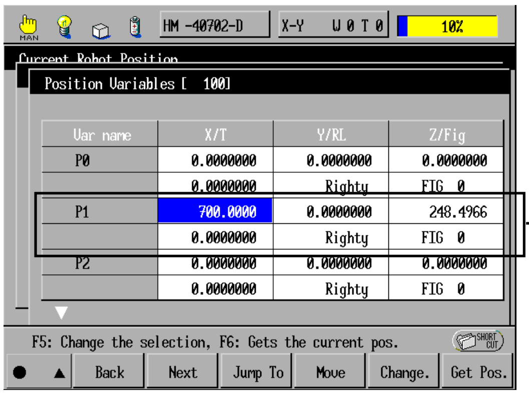
⑤ Press [F6 Get Pos.].



Are you sure you want to read the current position into the P[1]?

| | | | |
|---|---|---|---|
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

⑥ Check the system message and if all is correct, press [OK].


HM -40702-D X-Y W 0 T 0 10%

Current Robot Position

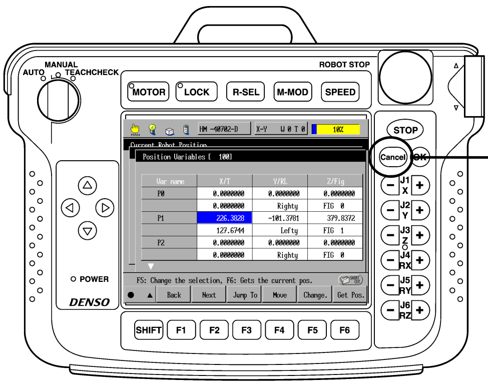
Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P0 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P1 | 700.0000 | 0.0000000 | 248.4966 |
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

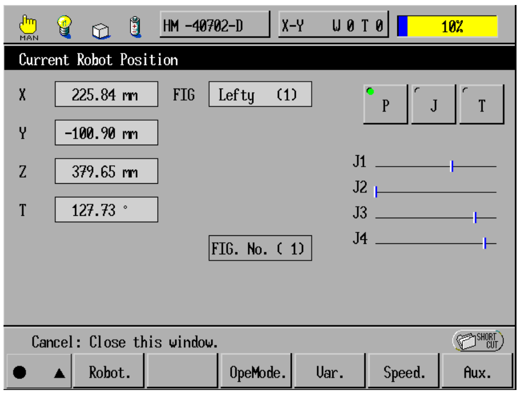
F5: Change the selection, F6: Gets the current pos.

The current position will be read into variable P1.

## Step 3 Teaching robot position P2 and assigning it to [Var name P2]



① Press [Cancel] twice to return to the [Current Robot Position] window.


HM -40702-D X-Y W 0 T 0 10%

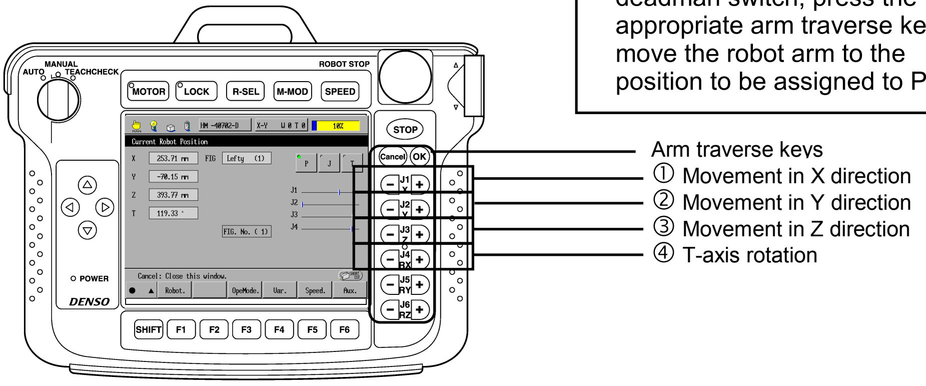
Current Robot Position

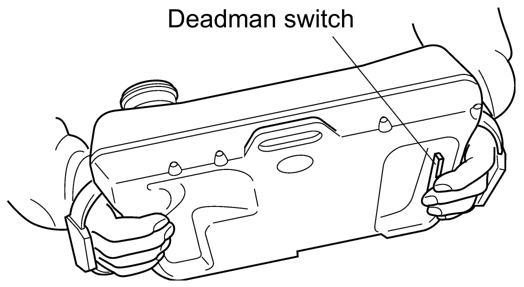
| | | | |
|---|---|---|---|
| X | 225.84 mm | FIG | Lefty (1) |
| Y | -100.90 mm | | |
| Z | 379.65 mm | | |
| T | 127.73 ° | | |

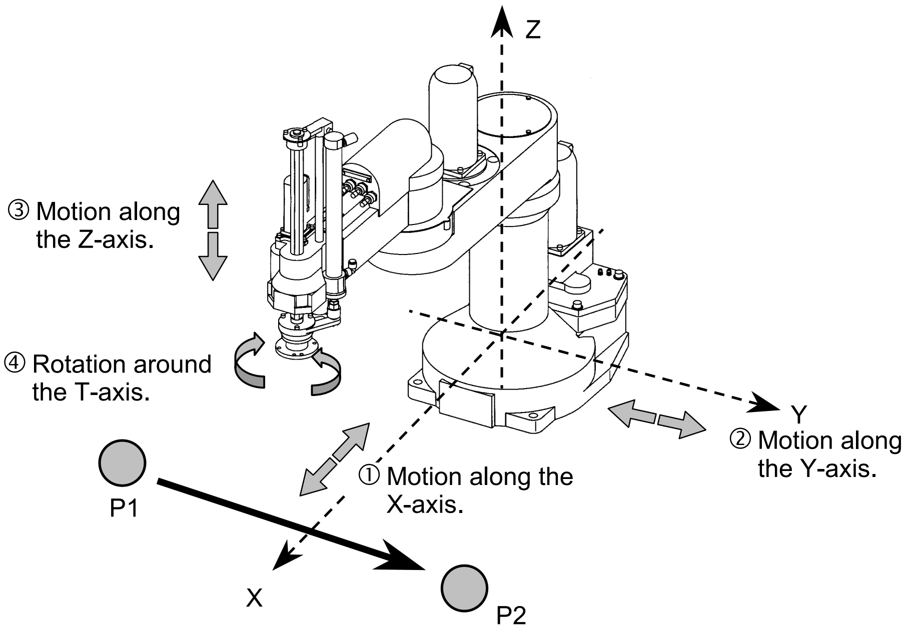
FIG. No. ( 1)

Cancel: Close this window.

② While holding down the deadman switch, press the appropriate arm traverse keys to move the robot arm to the position to be assigned to P2.






Z
③ Motion along the Z-axis.
④ Rotation around the T-axis.
Y
② Motion along the Y-axis.
① Motion along the X-axis.
P1
X
P2

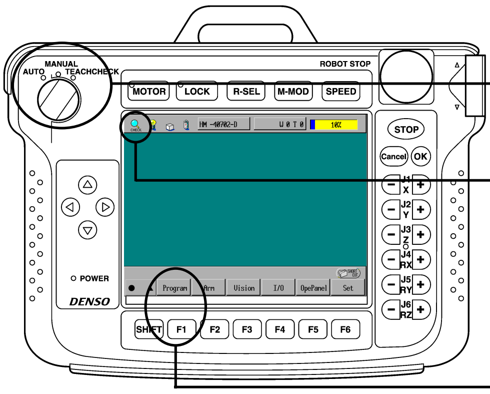
③ Assign the value taught for P2 to [Var name P2] in the same way as in Step 2, "Assigning the taught value to [Variable P1]."

This completes the teaching of P1 and P2.

## 2.3 Teach check

"Teach check" refers to checking the teaching results by running the program manually. You may take the teach check procedure in Teach check mode.

**Step 1** <u>**Placing the robot in Teach check mode**</u>

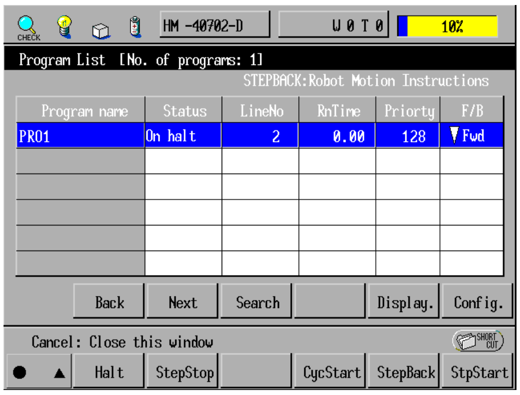


① Set the mode selector switch to the TEACHCHECK position.

In the leftmost area of the status bar, an icon indicating TEACHCHECK mode is displayed.

② Press [F1 Program] in the top screen.

The Program List window appears.


CHECK HM -40702-D U 0 T 0 10%

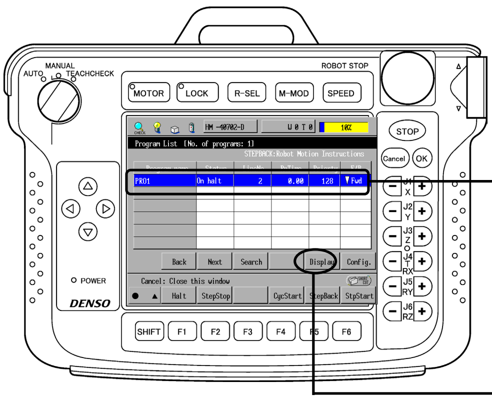
Program List [No. of programs: 1]

STEPBACK:Robot Motion Instructions

| Program name | Status | LineNo | RnTime | Priorty | F/B |
|---|---|---|---|---|---|
| PRO1 | On halt | 2 | 0.00 | 128 | Fwd |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Cancel: Close this window

## Step 2 Step check

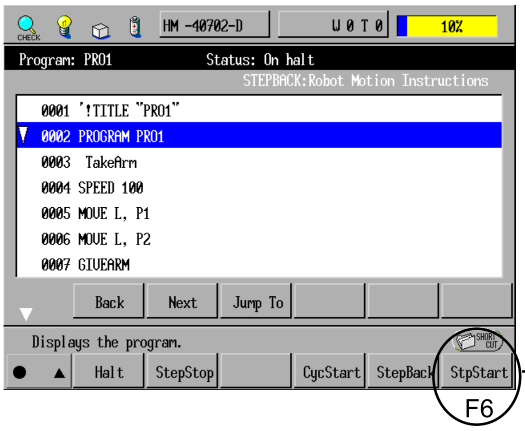


① Select "PRO1" in the Program List window.

(Selection can be made using the cursor keys or jog dial, or by touching the screen directly.)

② Press [Display.] to display the PRO1 program codes.

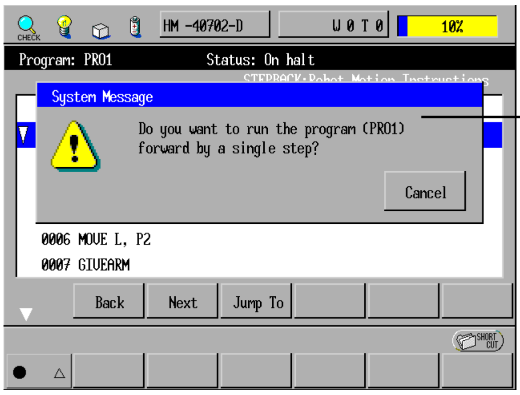
The PRO1 coding list appears in the program edit window "Program: PRO1".



```
HM -40702-D          U 0 T 0   10%
Program: PRO1              Status: On halt
                     STEPBACK:Robot Motion Instructions
0001 '!TITLE "PRO1"
0002 PROGRAM PRO1
0003  TakeArm
0004 SPEED 100
0005 MOVE L, P1
0006 MOVE L, P2
0007 GIVEARM
```

③ Press [F6 StpStart].

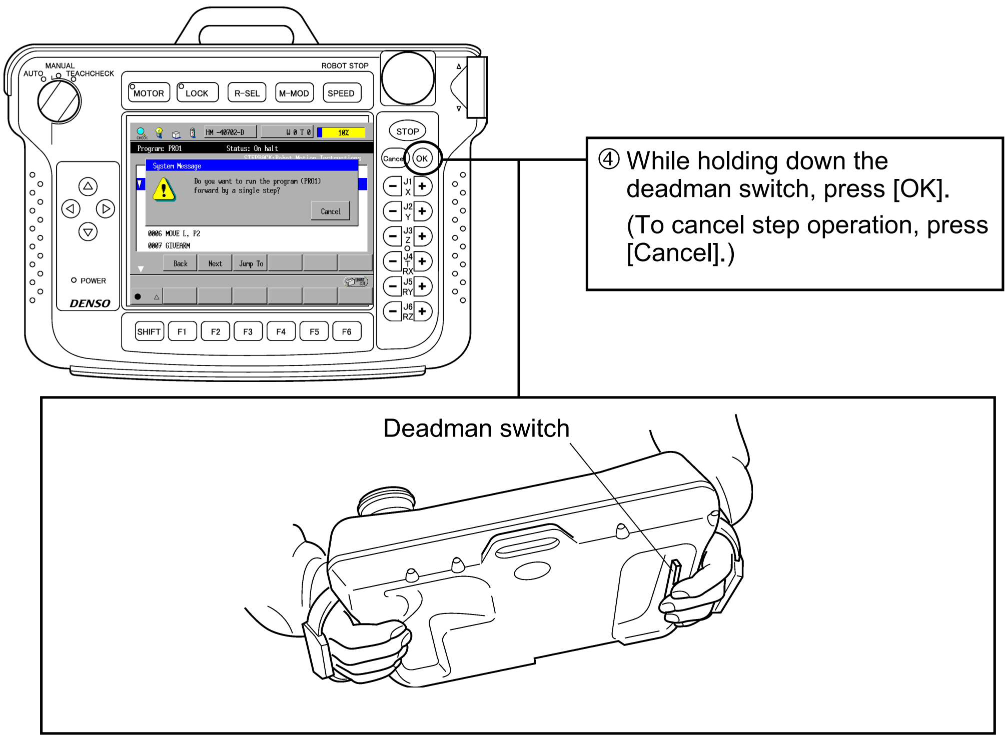
(This is also possible with the right cursor.)



```
HM -40702-D        U 0 T 0   10%
Program: PRO1          Status: On halt
System Message
Do you want to run the program (PRO1)
forward by a single step?
                              Cancel
0006 MOVE L, P2
0007 GIVEARM
```

This system message is displayed.

★**Caution**★ During teach check, always keep one hand free and ready to press the STOP key.



④ While holding down the deadman switch, press [OK].

(To cancel step operation, press [Cancel].)

In Teach check mode, keep both the deadman switch and OK key depressed until the execution is completed. If either of them is released, the robot comes to a halt instantly.

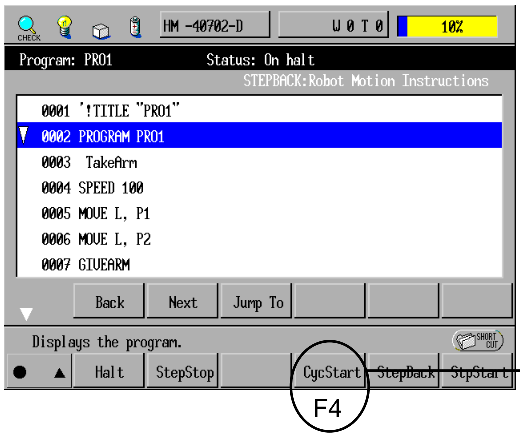
⇩

Perform the procedure above repeatedly to execute all codes in PRO1, checking that each motion is safe.

**Step 2**

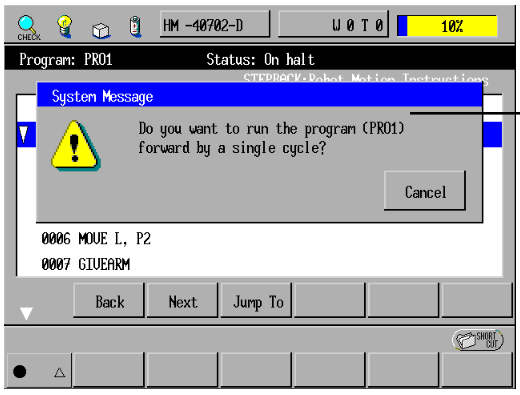
## Cycle check

Next, check the program you have just checked with Step check, this time with Cycle check. The Cycle check executes the selected program from the current program line to the end as a single cycle.



```
HM -40702-D    W 0 T 0   10%
Program: PR01          Status: On halt
                STEPBACK:Robot Motion Instructions
0001 '!TITLE "PR01"
0002 PROGRAM PR01
0003  TakeArm
0004 SPEED 100
0005 MOVE L, P1
0006 MOVE L, P2
0007 GIVEARM
Displays the program.
```

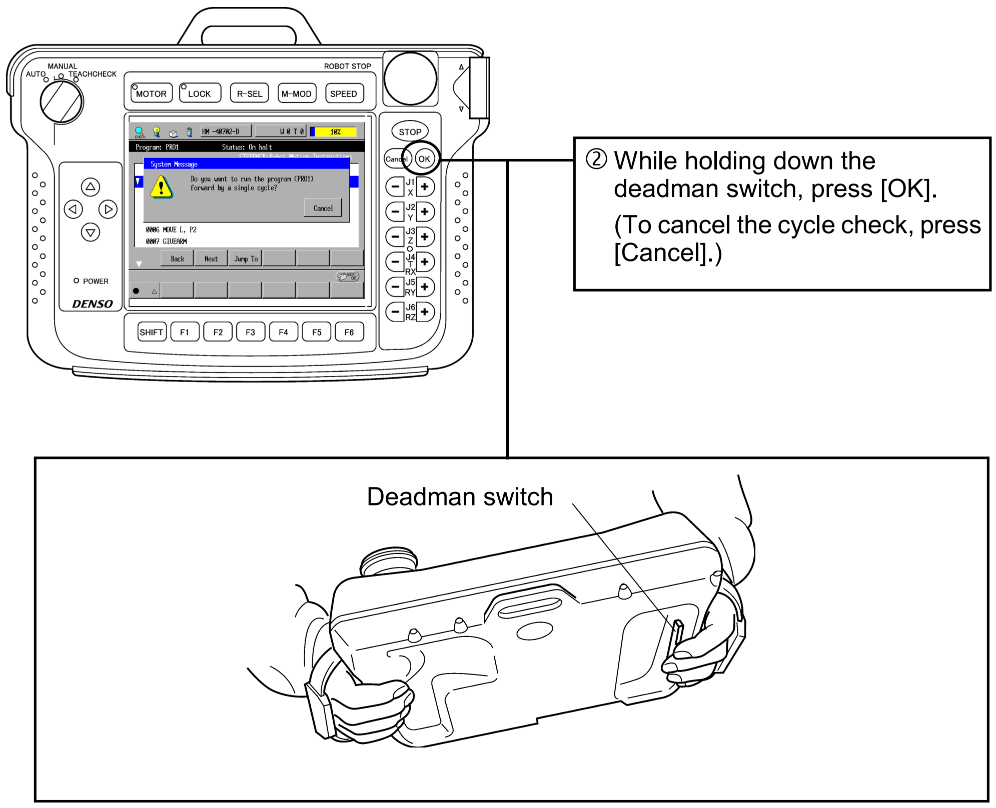
① Press [F4 CycStart].



```
HM -40702-D    W 0 T 0   10%
Program: PR01          Status: On halt
System Message
Do you want to run the program (PR01)
forward by a single cycle?
0006 MOVE L, P2
0007 GIVEARM
```

This system message appears.

**★Caution★** During teach check, always keep one hand free and ready to press the STOP key.



Program: PRO1 Status: On halt

System Message

Do you want to run the program (PRO1) forward by a single cycle?

0006 MOVE L, P2
0007 GIVEARM

② While holding down the deadman switch, press [OK].
(To cancel the cycle check, press [Cancel].)

In Teach check mode, keep both the deadman switch and OK key depressed until the execution is completed. If either of them is released, the robot comes to a halt instantly.

⇩

As the program starts to execute cycle check so that the robot runs, the highlighted section on the coding list window will proceed in order.

When the program has been executed through to the end, it will stop.

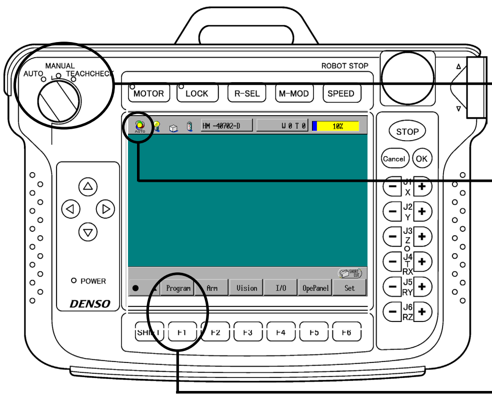
## 2.4 Running the robot in Auto mode

After the teach check, now you will run the program in Auto mode according to the program PRO1 that you edited in the last section.

**Caution:** For programs that will be executed for the first time in Auto mode, set the reduced ratio of the programmed speed at 10% or less. In Auto mode, the robot may run at full speed, while in Manual mode or Teach check mode the robot speed is automatically reduced to 10% of the full speed.

**Step 1**

**Placing the robot in Auto mode**



① Set the mode selector switch to AUTO.

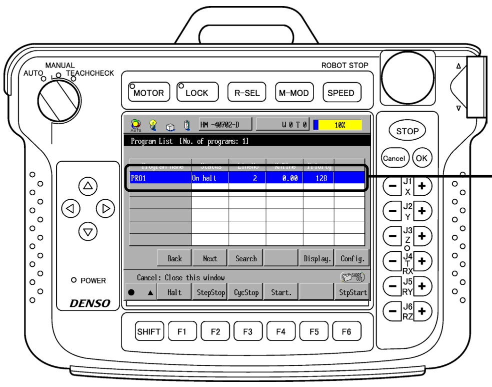
In the leftmost area of the status bar, an icon indicating Auto mode will be displayed.

② Press [F1 Program].

**Step 2**

**Selecting the program to be executed**

In the [Program List] window, select the program to be run in Auto mode.

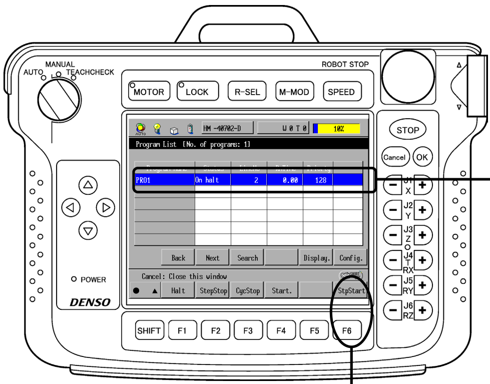


① Select "PRO1".

(Selection can be made using the cursor keys or jog dial, or by touching the screen directly.)

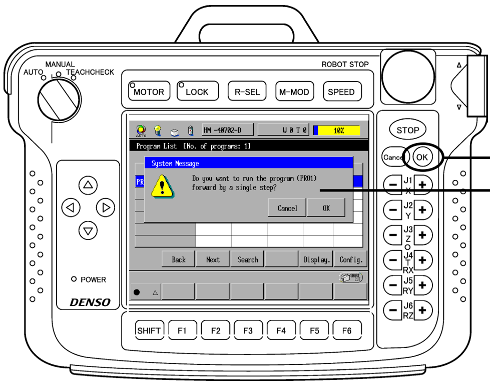
**Step 3** **Single-step run**

**★Note★** If you want to display the program during a single-step run, press [F11 Display] beforehand.



① Check that the program to be started up is selected.

② Press [F6 StpStart].
(This is also possible with the right cursor.)



This system message appears.

③ Press [OK].
(To cancel a single-step run, press [Cancel].)

**★Caution★** During program running, always keep one hand free and ready to press the STOP key.

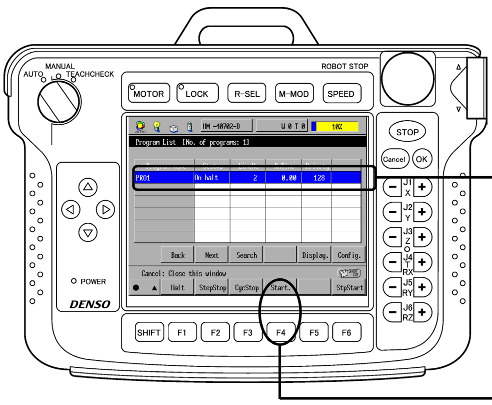
The PRO1 program will start a single-step run in Auto mode.

Perform the procedure above repeatedly through to the end of the program, checking that each motion is safe.

**Step 4**

## Single-cycle run

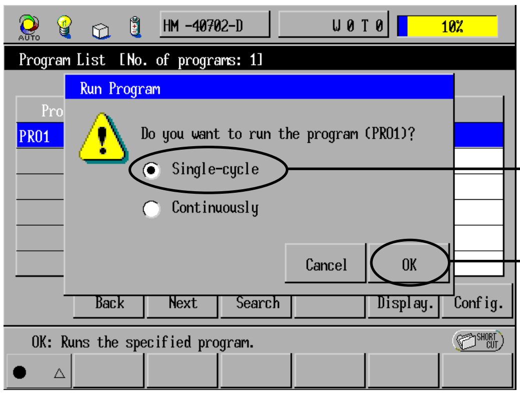
After running a single-step run, start a single-cycle run.



① Check that the program to be started is selected.

② Press [F4 Start.].

**★Caution★** During program running, always keep one hand free and ready to press the STOP key.



Program List [No. of programs: 1]

Run Program

PRO1

Do you want to run the program (PRO1)?

[✓ Single-cycle] Continuously

Cancel [✓ OK]

OK: Runs the specified program.

③ Select [Single-cycle] and press [OK].
Program PRO1 is executed.

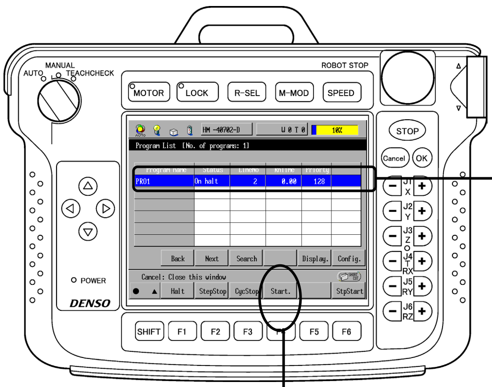
Once the program has been run to the end, it will stop.

**★Caution★** The elapsed time on display refers to the length of time from the start to end of the program including temporary stop time caused by Step stop or Halt.

**Step 5**

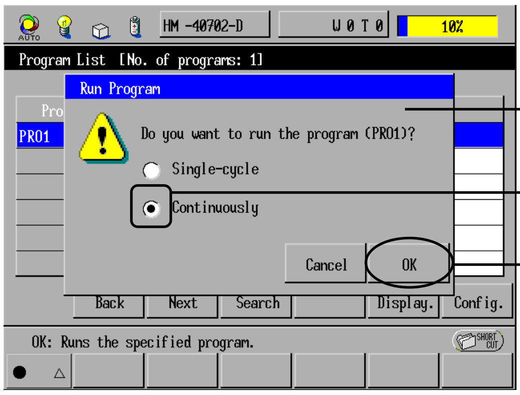
## Continuous run

Start a continuous run of the program.



① Check that the program to be started is selected.

② Press [F4 Start.].



Do you want to run the program (PRO1)?

Single-cycle [✓ Continuously]

Cancel OK

OK: Runs the specified program.

The selection screen for [Single-cycle] and [Continuously] is displayed.

③ Select [Continuously].

④ Press [OK].
Program PRO1 will be executed continuously.
(You may stop continuous run by Halt (Stop) or Step stop.)

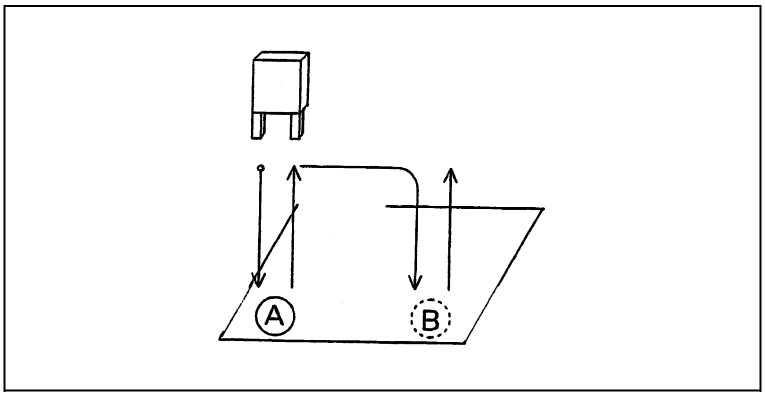
**★Caution★** During program running, always keep one hand free and ready to press the STOP key.

⇩

This completes the procedures required to run the robot with the teach pendant.

## Part 1  Practice Problems

**Exercise:** Create a program for moving a workpiece from point A to point B, and then performing an operation check.



Assuming that:

- I/O assignment  No. 64 (system output) … Close hand
  No. 65 (system output) … Open hand
- Speed ratio when collecting and putting a workpiece: 30%
  Speed ratio for other motions: 80%
- Each of the approach and depart distances: 50 mm
- Interpolation control: PTP

**Answer:**

```
'!TITLE "P&P"
PROGRAM PRO10
  TAKEARM
    APPROACH P , P5 , 50 , S=80
    MOVE P , P5 , S=30
    DELAY 500
    RESET IO[65]
    SET IO[64]
    DEPART P , 50 , S=80
    APPROACH P , P6 , 50 , S=80
    MOVE P , P6 , S=30
    DELAY 500
    RESET IO[64]
    SET IO[65]
    DEPART P , 50 , S=80
  GIVEARM
END
```

# Part 2
# Creating a Program on a PC in WINCAPSII

In Part 2, you will:

Start up the PC teaching system WINCAPSII on a personal computer and actually create and compile a program. You will then upload the compiled program to the robot controller.

Further, in Part 2, you will also place the robot controller in machine lock. This is in preparation for Part 3 where you will simulate the programmed robot motion on the PC screen without actually running the robot.

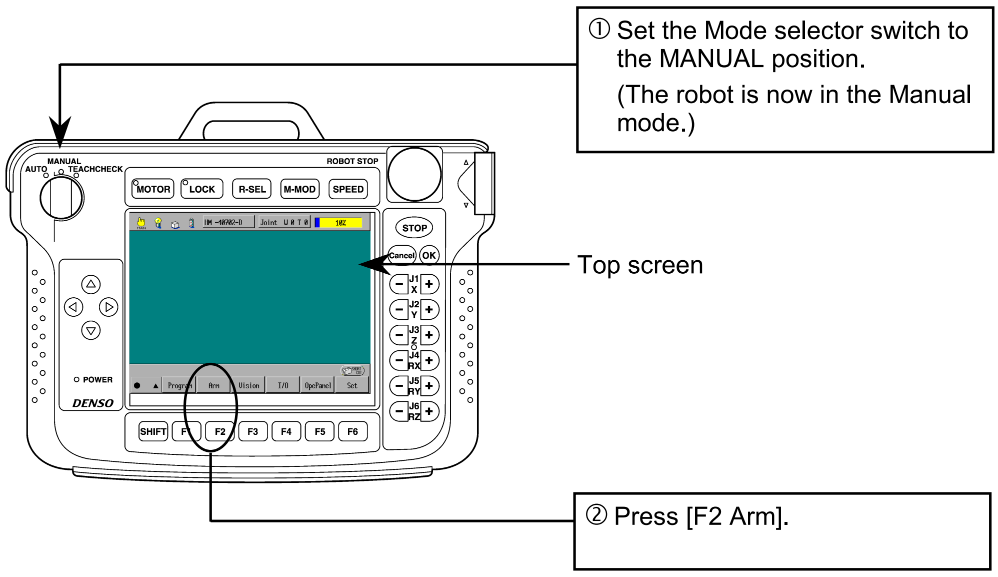
# Lesson 3 Setting Up the Robot Controller with the Teach Pendant

## 3.1 Performing calibration (CAL) (for **-D series only)

The **-E/-F/-G series and VM-6083D/60B1D robots require no CAL operation. Only the **-D series requires CAL to run the robot using accurate values. Performing CAL precisely actuates all robot axes.

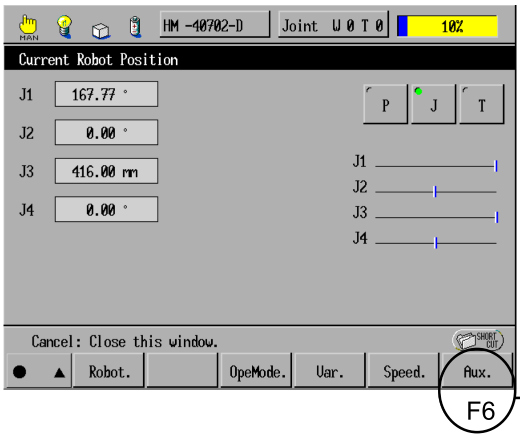
**Warning** Performing CAL operation will move the robot arm. Before proceeding with the CAL procedure, make sure that all workers have left the robot's restricted space and that there are no obstacles within.

**Step 1** **Placing the robot in Manual mode**



**Step 2**

## Starting CAL operation



HM -40702-D  Joint W 0 T 0  10%

Current Robot Position

J1 167.77 °
J2 0.00 °
J3 416.00 mm
J4 0.00 °

P J T

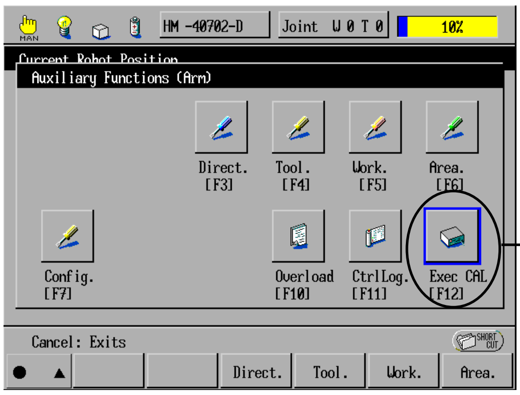
J1
J2
J3
J4

Cancel: Close this window.

Robot. OpeMode. Var. Speed. Aux.

F6

① Press [F6 Aux.].



HM -40702-D  Joint W 0 T 0  10%

Current Robot Position

Auxiliary Functions (Arm)

Direct. [F3]  Tool. [F4]  Work. [F5]  Area. [F6]

Config. [F7]  Overload [F10]  CtrlLog. [F11]  Exec CAL [F12]

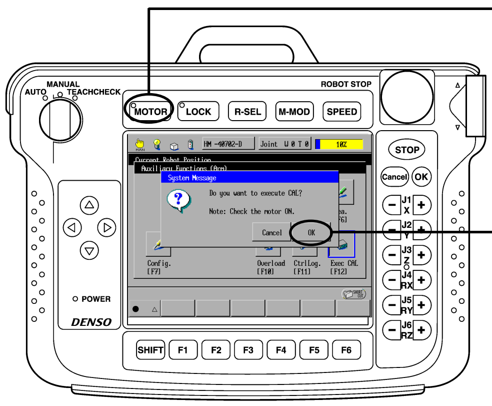
Cancel: Exits

Direct. Tool. Work. Area.

② Press [F12 Exec CAL].

**Step 3**

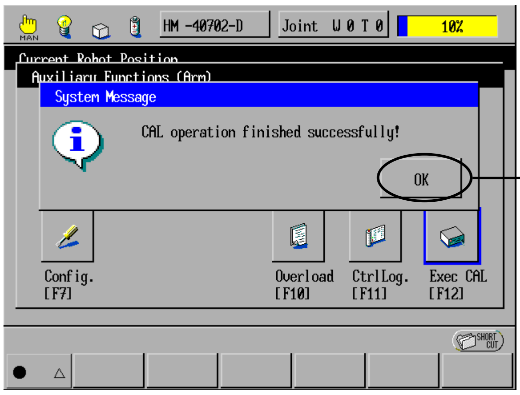
## Turning the motor ON



① Press [MOTOR].
The motor will be turned ON and the [MOTOR] lamp will light.

② Press [OK].

## Step 4 Ending CAL operation


HM -40702-D | Joint W 0 T 0 | 10%

Current Robot Position

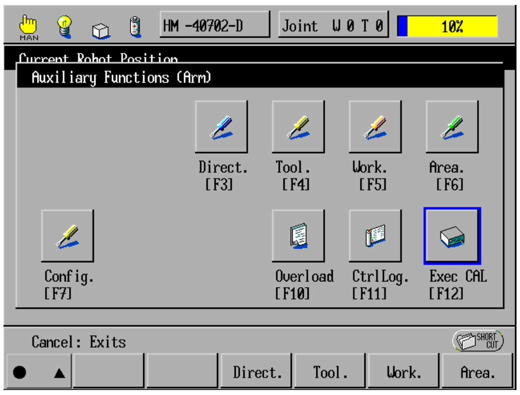
Auxiliary Functions (Arm)

System Message

CAL operation finished successfully!

OK

① Press [OK].

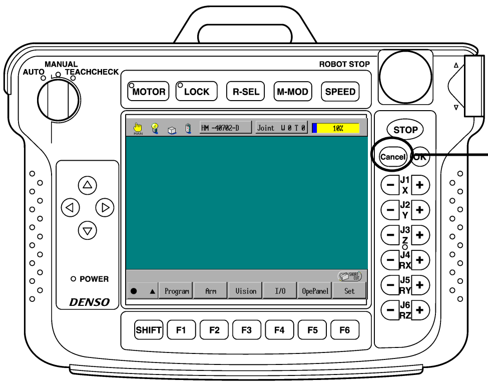

HM -40702-D | Joint W 0 T 0 | 10%

Current Robot Position

Auxiliary Functions (Arm)

Cancel: Exits

This completes CAL.



② Press [Cancel].
(The display returns to the [Robot Current Position] window.)

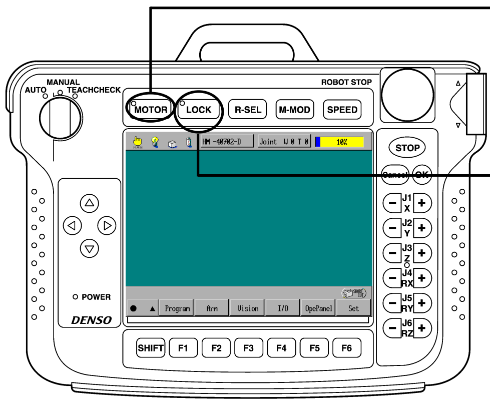
③ Press [Cancel].
The display returns to the top screen.)

## 3.2 Placing the robot controller in machine lock

You will now place the robot controller in machine lock. This enables you to simulate the programmed robot motion on the PC screen without actually running the robot in Part 3.

**Step 1** **Turning the motor OFF**

**Step 2** **Placing the robot in machine lock**



① Press [MOTOR].
(The motor power is turned OFF and the [MOTOR] lamp goes off.)

② Press [LOCK].
(The robot controller is locked and the [LOCK] lamp lights.)

**★Caution★** Before placing the robot controller in machine lock, ensure that the motor power is OFF; that is, check that the [MOTOR] lamp is off.

**★Tip★** [Ver. 1.4 or later]

If the machine is locked, you may restrict I/O output. For details, refer to the SETTING-UP MANUAL, Section 5.5 "Displaying I/O Signals and Simulating Robot Motion."

The dummy input icon on the status bar changes according to the I/O output restriction condition.

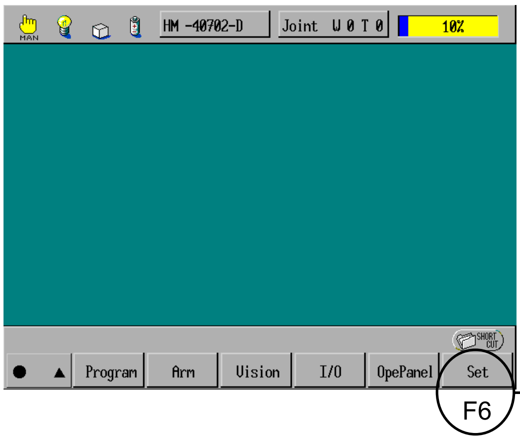
: No I/O output restricted : I/O output restricted

## 3.3 Setting the communications port of the robot controller

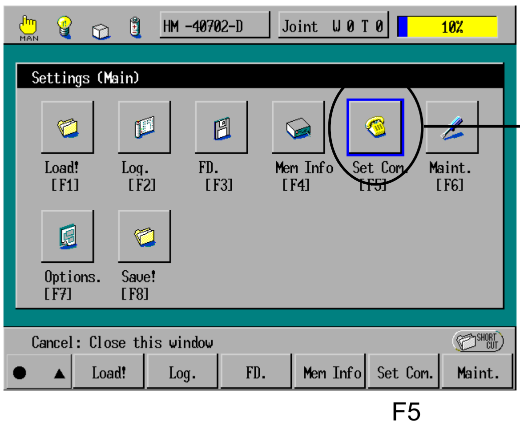
To enable the robot controller to communicate with the personal computer, you need to set up the communications port.

This subsection describes the most popular connection using the RS232C.

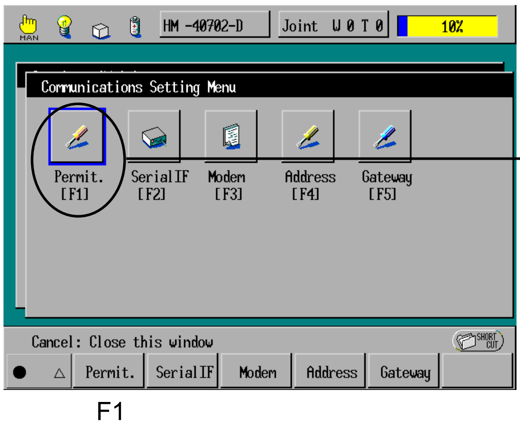
### Step 1 Setting the communication permission



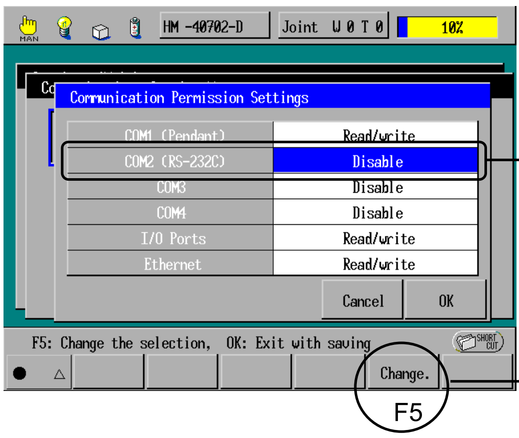
① Press [F6 Set].



② Press [F5 Set Com.].



③ Press [F1 Permit.].



HM -40702-D | Joint W 0 T 0 | 10%

Communication Permission Settings

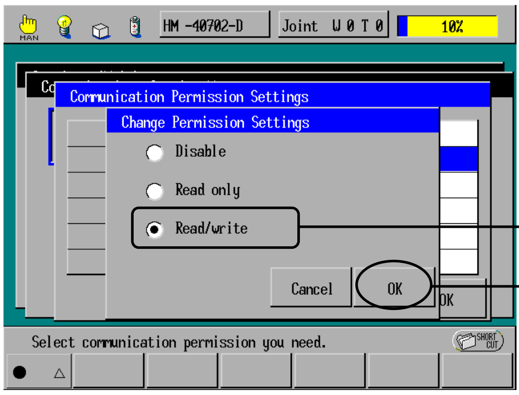
| | |
|---|---|
| COM1 (Pendant) | Read/write |
| COM2 (RS-232C) | Disable |
| COM3 | Disable |
| COM4 | Disable |
| I/O Ports | Read/write |
| Ethernet | Read/write |

Cancel OK

F5: Change the selection, OK: Exit with saving

④ Select the [COM2 (RS232C)] row.

⑤ Press [F5 Change.].



HM -40702-D | Joint W 0 T 0 | 10%

Communication Permission Settings

Change Permission Settings

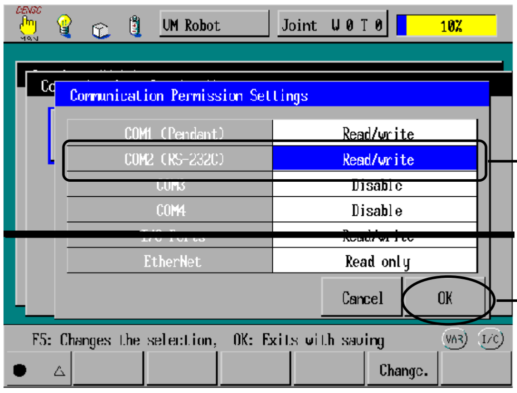
- Disable
- Read only
- ● Read/write

Cancel OK

Select communication permission you need.

⑥ Select [Read/write].

⑦ Press [OK].



UM Robot | Joint W 0 T 0 | 10%

Communication Permission Settings

| | |
|---|---|
| COM1 (Pendant) | Read/write |
| COM2 (RS-232C) | Read/write |
| COM3 | Disable |
| COM4 | Disable |
| I/O Ports | Read/write |
| EtherNet | Read only |

Cancel OK

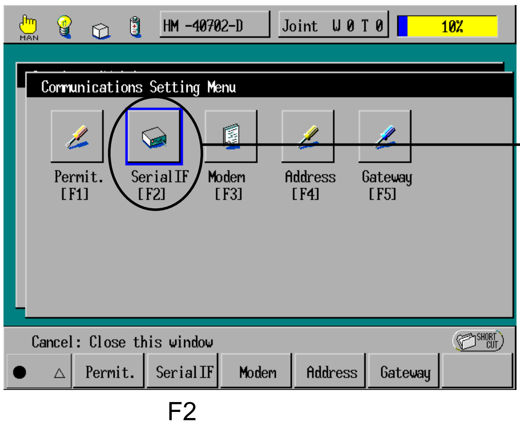
F5: Changes the selection, OK: Exits with saving

The [COM2 (RS232C)] column changes to [Read/write].

⑧ Press [OK].

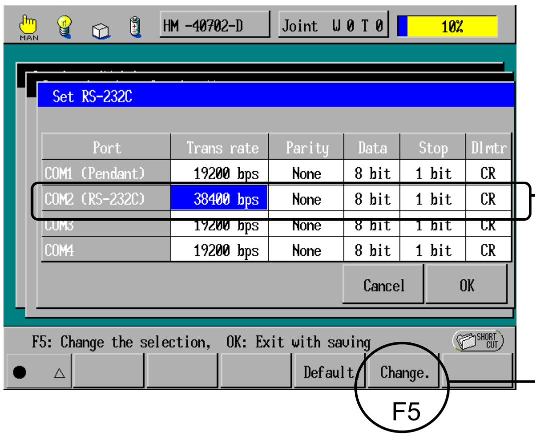
**Step 2**

## Setting the transmission rate



Cancel: Close this window

① Press [F2 Serial IF].

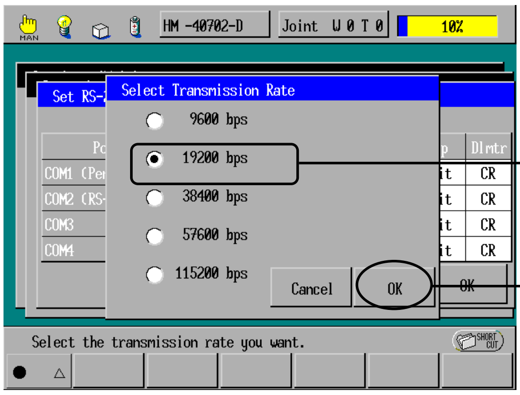


| Port | Trans rate | Parity | Data | Stop | Dlmtr |
|---|---|---|---|---|---|
| COM1 (Pendant) | 19200 bps | None | 8 bit | 1 bit | CR |
| COM2 (RS-232C) | 38400 bps | None | 8 bit | 1 bit | CR |
| COM3 | 19200 bps | None | 8 bit | 1 bit | CR |
| COM4 | 19200 bps | None | 8 bit | 1 bit | CR |

F5: Change the selection, OK: Exit with saving

② Select the [COM2 (RS232C)] row.

③ Press [F5 Change.].

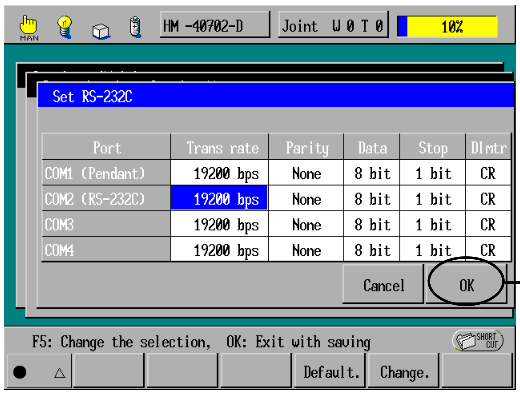


- ○ 9600 bps
- ● 19200 bps
- ○ 38400 bps
- ○ 57600 bps
- ○ 115200 bps

Select the transmission rate you want.

④ Select [19200 BPS].

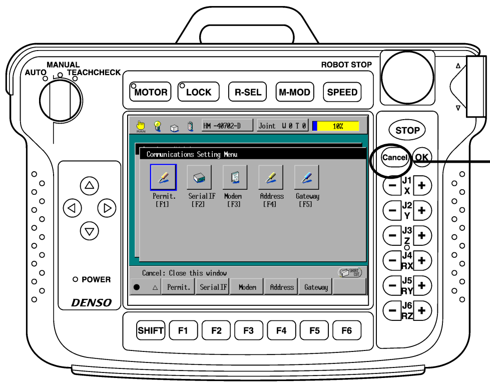
⑤ Press [OK].



| Port | Trans rate | Parity | Data | Stop | Dlmtr |
|---|---|---|---|---|---|
| COM1 (Pendant) | 19200 bps | None | 8 bit | 1 bit | CR |
| COM2 (RS-232C) | 19200 bps | None | 8 bit | 1 bit | CR |
| COM3 | 19200 bps | None | 8 bit | 1 bit | CR |
| COM4 | 19200 bps | None | 8 bit | 1 bit | CR |

F5: Change the selection, OK: Exit with saving

⑥ Press [OK].



⑦ Press [Cancel].

(The display returns to the [Communications Setting Menu].)

⑧ Press [Cancel].

(The display returns to the top screen.)

# Lesson 4 Starting up WINCAPSII and Creating a System Project

In this lesson, you will start up WINCAPSII with a PC and register a new system project. This is necessary in order to enter, edit and verify the program. You will also make settings for the communications port of the PC.

## 4.1 Starting the System Manager

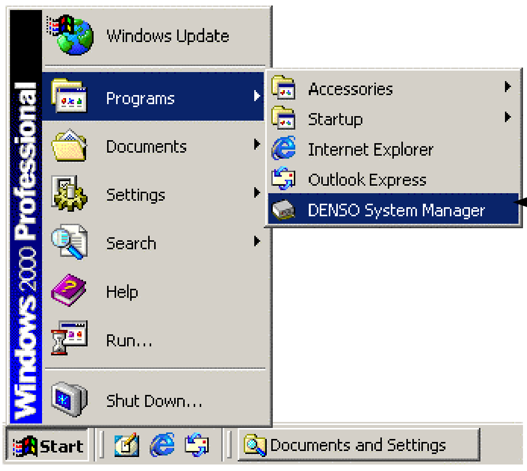
WINCAPSII consists of the following functional modules:

- PAC Program Manager
- Variable Manager
- DIO Manager
- Arm Manager
- Vision Manager
- Log Manager
- Communications Setting Manager

System Manager enables overall control of these functional modules. All functions in WINCAPSII may be called up from the System Manager.

To use the PC teaching system, first start the System Manager as follows:

**Step 1** **<u>Selecting System Manager</u>**

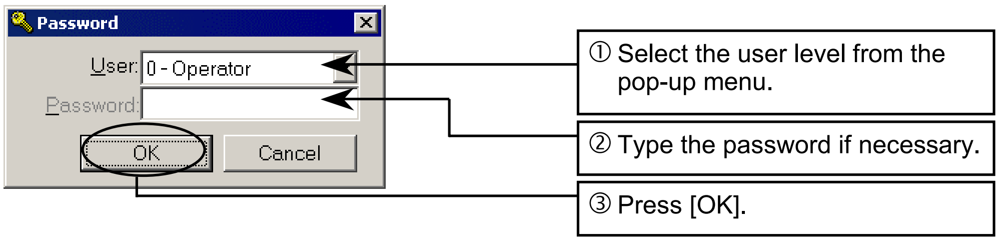


① From the Start button, access [System Manager] in the WINCAPSII folder.

**★Caution★**

- When starting System Manager first time after installation, you need to specify the file name of the program bank. The "Create a New Program Bank" dialog may appear as necessary. Refer to the WINCAPSII GUIDE, Chapter 5, Subsection 5.6.2.3 "Updating the program bank."
- When starting the System Manager for the first time, the Create New Project dialogue box appears because no system project has been defined yet. In this case, you must first perform "4.2 Registering a new system project" before proceeding to Step 2.

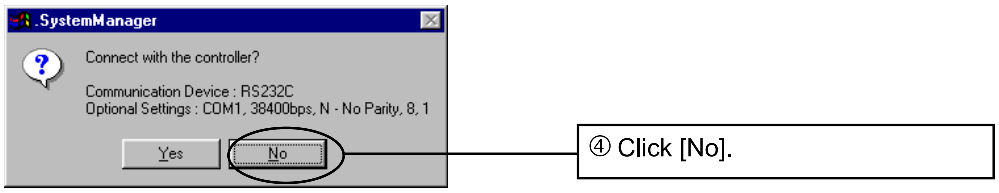
**Step 2** **Selecting the user level**



① Select the user level from the pop-up menu.

② Type the password if necessary.

③ Press [OK].

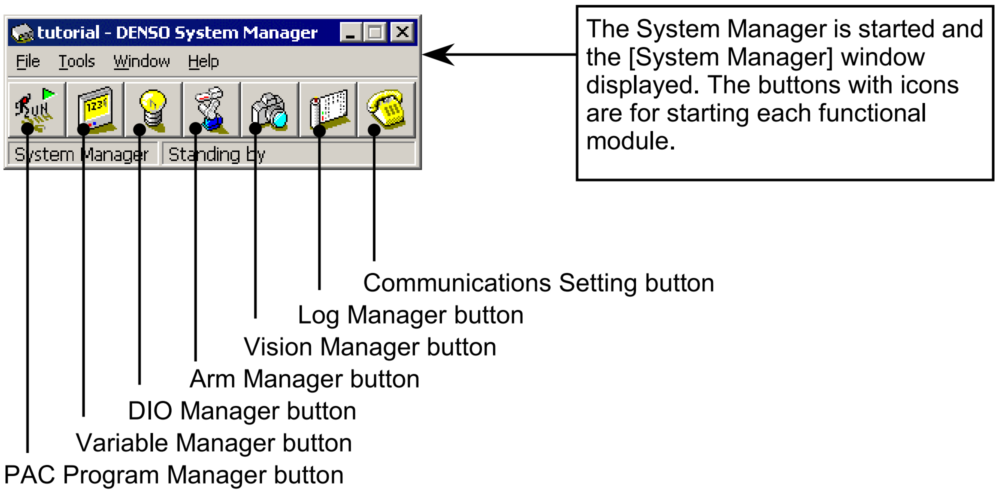


Connect with the controller?

Communication Device : RS232C
Optional Settings : COM1, 38400bps, N - No Parity, 8, 1

④ Click [No].

**★Point★** Shown below are the startup buttons of the functional modules on the tool bar.



The System Manager is started and the [System Manager] window displayed. The buttons with icons are for starting each functional module.

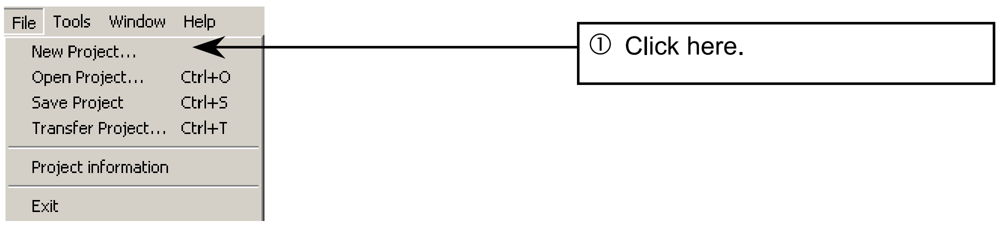
## 4.2 Registering a new system project

WINCAPSII controls more than one robot program in units of a project. To run a single robot, a set of combined programs will usually be used. Therefore, it is convenient to manage these programs as a set in one project.

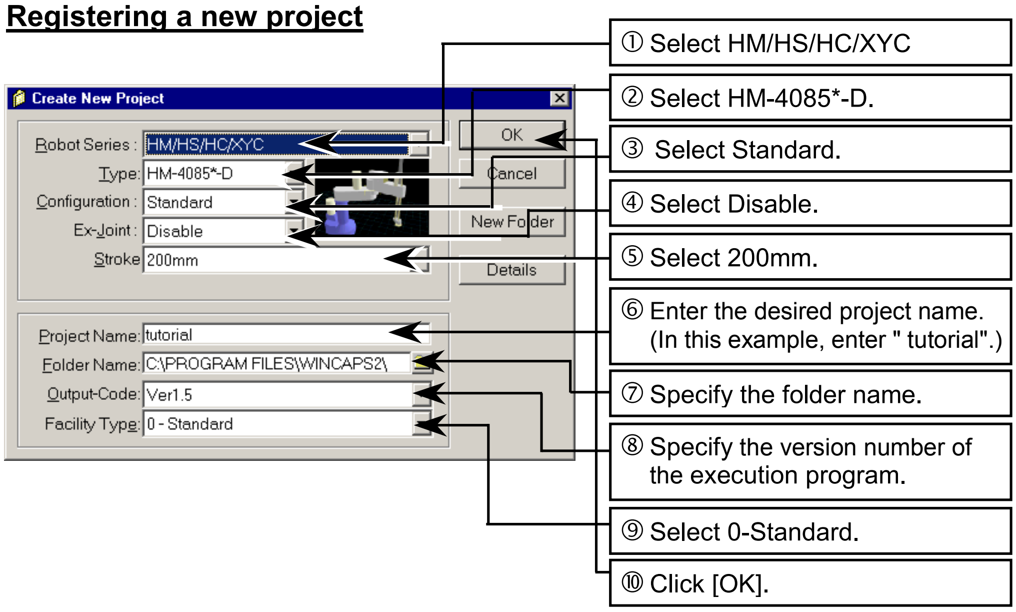
For creating a robot program, first you should register a new project.

**Caution:** When starting the System Manager for the first time, the [Create New Project] dialog box will appear, so first carry out "New Project Registration" before proceeding to Step 2 of "4.1 Starting the system manager".

**Step 1** **Selecting "New Project" from [File] menu**



**Step 2** **Registering a new project**



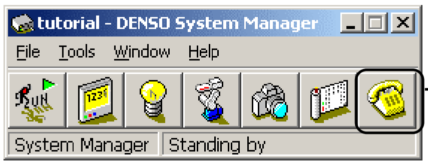
Robot Series : HM/HS/HC/XYC
Type: HM-4085*-D
Configuration : Standard
Ex-Joint : Disable
Stroke 200mm
Project Name: tutorial
Folder Name: C:\PROGRAM FILES\WINCAPS2\
Output-Code: Ver1.5
Facility Type: 0 - Standard

① Select HM/HS/HC/XYC
② Select HM-4085*-D.
③ Select Standard.
④ Select Disable.
⑤ Select 200mm.
⑥ Enter the desired project name. (In this example, enter " tutorial".)
⑦ Specify the folder name.
⑧ Specify the version number of the execution program.
⑨ Select 0-Standard.
⑩ Click [OK].

## 4.3 Setting the communications port of the PC

Make the WINCAPSII communications settings the same as those of the robot controller.

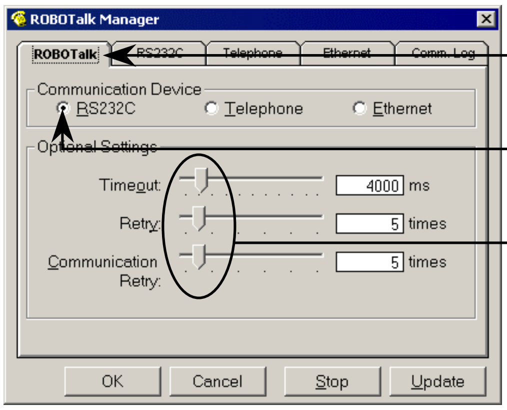
**Step 1** **Calling up the [ROBOTalk Manager] dialog box**



① Click on the [Communications Setting] button.

**★Caution★** If you have not yet entered the password, the [Password] dialog box will appear. You need to select the user level and enter the password. (Refer to "4.1 Starting the system manager".)

**Step 2** **Setting the communication device and optional settings**



Communication Device: [✓ RS232C] Telephone Ethernet

Optional Settings
Timeout: 4000 ms
Retry: 5 times
Communication Retry: 5 times

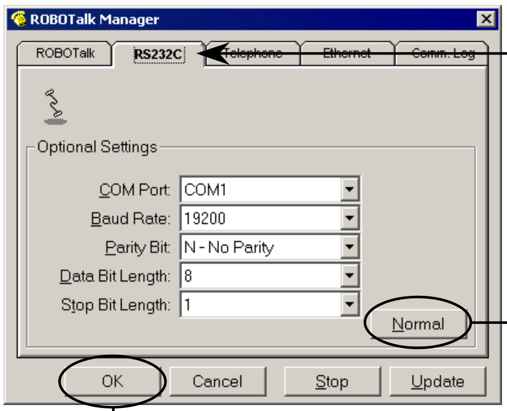
② Click the [ROBOTalk] tab.

③ Click here to select RS232C.

④ Make optional settings with these switches.
In this example, set the Timeout at 4000 msec, Retry at 5 times and Communication Retry at 5 times.

**★Caution★** If timeout occurs during data transmission, adjust the timeout to a longer period.

**Step 3** **Setting the RS232C communications options**



Optional Settings
COM Port: COM1
Baud Rate: 19200
Parity Bit: N - No Parity
Data Bit Length: 8
Stop Bit Length: 1

⑤ Click on the [RS232C] tab.

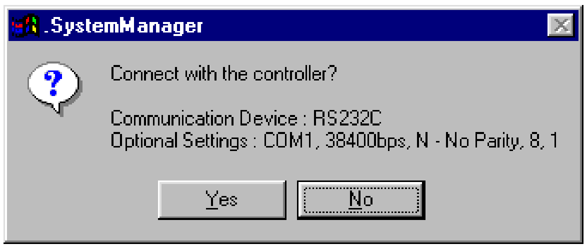
⑥ Click on [Normal].
The Optional Settings box changes to normal settings.

⑦ Click on [OK].
The settings become effective and the [ROBOTalk Manager] closes.

★**Caution**★ When making settings other than the normal settings, ensure the settings match the specifications of the robot controller or PC being used.

★**Caution**★ If [OK] is disabled, set the [Connect] button for all managers to OFF. If any of the managers is connected, it will not be possible to change the communications settings.

Once performed, the WINCAPSII communications settings will remain effective until you change them again. You do not need to perform the settings each time you start WINCAPSII. Just click on [Yes] in response to the [Resume connection?] dialog message that appears when WINCAPSII is started up. If you click on [No], no automatic settings will be made.



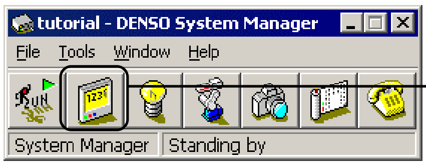
Connect with the controller?

Communication Device : RS232C
Optional Settings : COM1, 38400bps, N - No Parity, 8, 1

# Lesson 5  Defining Macros

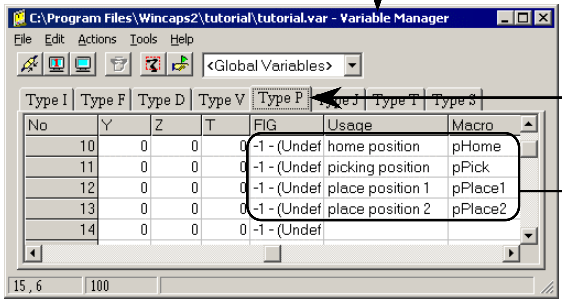
In this lesson you will create macro definition files by defining names and applications of variables and I/Os.

## Step 1  Creating a variable macro definition file



① Click on the Variable Manager button to start up the Variable Manager.

The [Variable Manager] window appears.

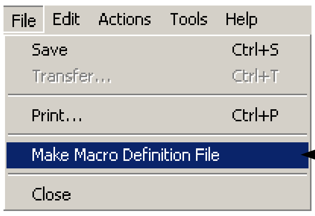

C:\Program Files\Wincaps2\tutorial\tutorial.var - Variable Manager
File Edit Actions Tools Help
<Global Variables>
Type I Type F Type D Type V Type P Type J Type T Type S
15 , 6 100

| No | Y | Z | T | FIG | Usage | Macro |
|---|---|---|---|---|---|---|
| 10 | 0 | 0 | 0 | -1 - (Undef | home position | pHome |
| 11 | 0 | 0 | 0 | -1 - (Undef | picking position | pPick |
| 12 | 0 | 0 | 0 | -1 - (Undef | place position 1 | pPlace1 |
| 13 | 0 | 0 | 0 | -1 - (Undef | place position 2 | pPlace2 |
| 14 | 0 | 0 | 0 | -1 - (Undef | | |

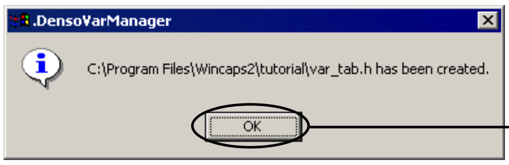
② Click on the [Type P] tab.

③ Double click on each box and enter the usage and macro names of the variables.

In this example, enter position variables P10 to P13.



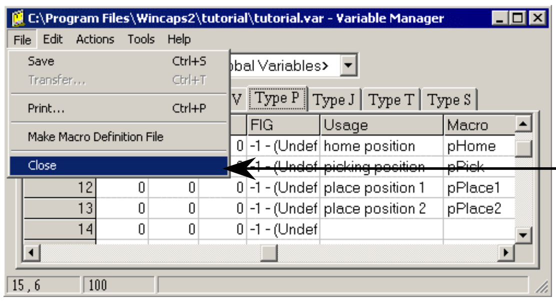
④ Click on [Make Macro Definition File].



C:\Program Files\Wincaps2\tutorial\var_tab.h has been created.

⑤ Press [OK].

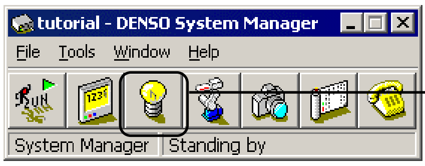
The macro definition file is now created.


C:\Program Files\Wincaps2\tutorial\tutorial.var - Variable Manager
File Edit Actions Tools Help
Save Ctrl+S
Transfer... Ctrl+T
Print... Ctrl+P
Make Macro Definition File
Close
bal Variables>
V Type P Type J Type T Type S
15 , 6 100

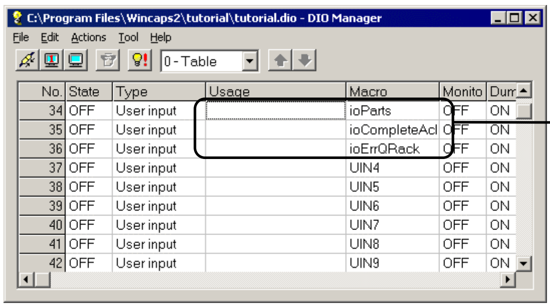
| | | | | FIG | Usage | Macro |
|---|---|---|---|---|---|---|
| | | | 0 | -1 - (Undef | home position | pHome |
| | | | | -1 - (Undef | picking position | pPick |
| 12 | 0 | 0 | 0 | -1 - (Undef | place position 1 | pPlace1 |
| 13 | 0 | 0 | 0 | -1 - (Undef | place position 2 | pPlace2 |
| 14 | 0 | 0 | 0 | -1 - (Undef | | |

⑥ Click here to exit Variable Manager.

## Step 2 Making an I/O macro definition file

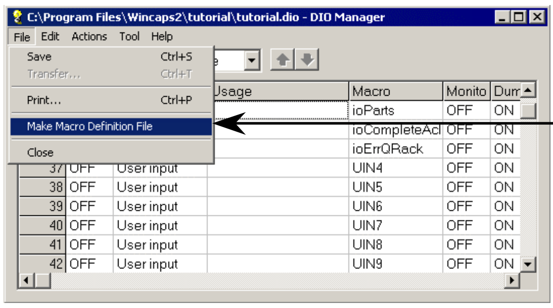


① Click on the DIO Manager button to start up the DIO Manager.


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
0 - Table

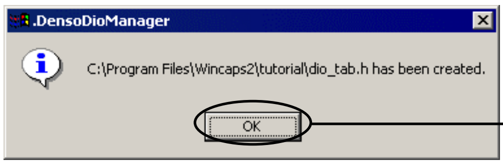
| No. | State | Type | Usage | Macro | Monito | Durr |
|---|---|---|---|---|---|---|
| 34 | OFF | User input | | ioParts | OFF | ON |
| 35 | OFF | User input | | ioCompleteAcl | OFF | ON |
| 36 | OFF | User input | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

② Double click on each box and enter the usage and macro names of the I/Os.


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
Save Ctrl+S
Transfer... Ctrl+T
Print... Ctrl+P
Make Macro Definition File
Close

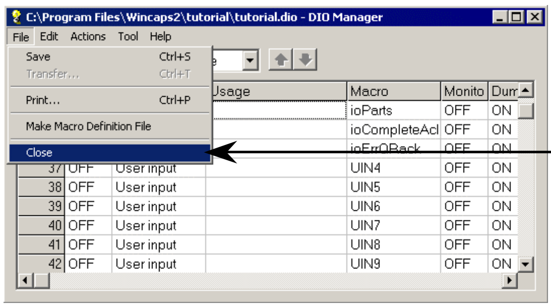
| No. | State | Type | Usage | Macro | Monito | Durr |
|---|---|---|---|---|---|---|
| | | | | ioParts | OFF | ON |
| | | | | ioCompleteAcl | OFF | ON |
| | | | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

③ Click on [Make Macro Definition File].



C:\Program Files\Wincaps2\tutorial\dio_tab.h has been created.

④ Press [OK].
The I/O macro definition file is now created.


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
Save Ctrl+S
Transfer... Ctrl+T
Print... Ctrl+P
Make Macro Definition File
Close

| No. | State | Type | Usage | Macro | Monito | Durr |
|---|---|---|---|---|---|---|
| | | | | ioParts | OFF | ON |
| | | | | ioCompleteAcl | OFF | ON |
| | | | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

⑤ Click [Close] to exit DIO Manager.

# Lesson 6  Inputting and Editing Programs

## 6.1      Sample program

Before starting the program input procedure, take a look over the coding list sample below.

Read through the process to get an understanding of the motion, while referring to the comments to the right.

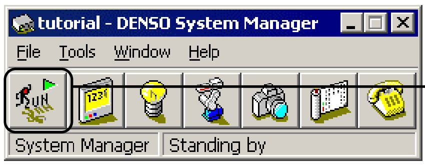
**Coding List "PRO1"**

```
'!TITLE "Pick & Place"
#INCLUDE  "dio_tab.h"              'Reads the DIO macro definition file.
#INCLUDE  "var_tab.h"              'Reads the variable macro definition file.

PROGRAM pro1
    TAKEARM                        Acquires arm semaphore.
    SET IO[ioComplate]
    MOVE P, P[pHome], S=50         'PTP control to home position at 50% internal speed
    SPEED 100                      'Changes to 100% speed.
    APPROACH P,P[pPick],200
    MOVE P,P[pPick]
    GOSUB *ChuckItem
    DEPART P,200
    APPROACH P,P[pPlace1],200
    MOVE P,P[pPlace1]
    GOSUB *UnchuckItem
    DEPART P,200
    SET IO[ioComplate]             'Issues a motion complete signal.
    GIVEARM                        'Releases arm semaphore.
    END

' ===== Chuck parts =====
*ChuckItem:
RESET IO[ioUnChuck]
SET IO[ioChuck]
RETURN

' ===== Unchuck parts =====
*UnchuckItem:
RESET IO[ioChuck]
SET IO[ioUnChuck]
RETURN
```

## 6.2 Opening the program edit window

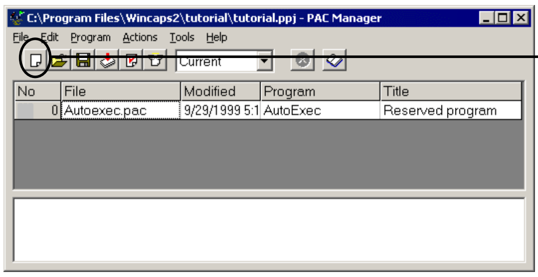
To input and edit task programs, use the Program Manager which is called up from the System Manager.

### Step 1 Starting PAC Program Manager



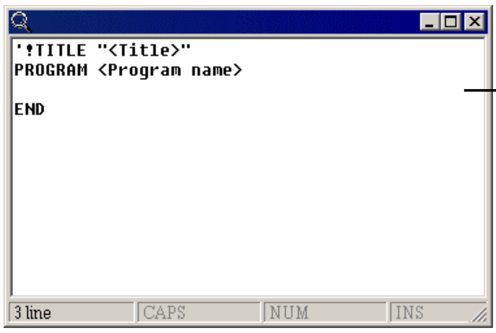
① Click on the PAC Program Manager button to start the PAC Program Manager.

### Step 2 Opening the program edit window


C:\Program Files\Wincaps2\tutorial\tutorial.ppj - PAC Manager
File
Edit
Program
Actions
Tools
Help
Current

| No | File | Modified | Program | Title |
|---|---|---|---|---|
| 0 | Autoexec.pac | 9/29/1999 5:1 | AutoExec | Reserved program |

② Click on the [New Program] button.

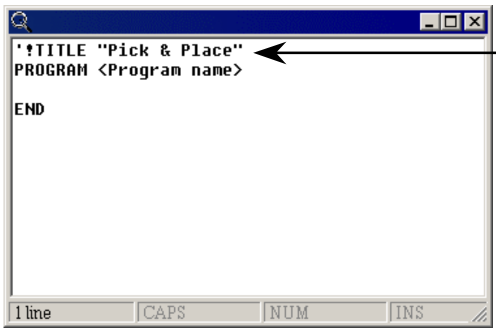


```
'!TITLE "<Title>"
PROGRAM <Program name>

END
```

A new edit window appears.

## 6.3 Inputting source code

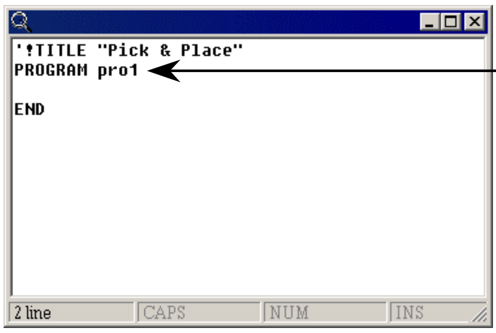
### Step 1 Typing the program title



```
'!TITLE "Pick & Place"
PROGRAM <Program name>

END
```

① Type the program title. (In this example, type "Pick & Place".)

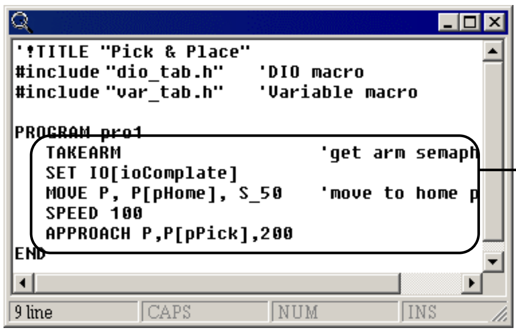
### Step 2 Typing the program name



```
'!TITLE "Pick & Place"
PROGRAM pro1

END
```

② Type the program name. (In this example, type “pro1”.)

### Step 3 Inputting the source code



```
'!TITLE "Pick & Place"
#include "dio_tab.h"    'DIO macro
#include "var_tab.h"    'Variable macro

PROGRAM pro1
    TAKEARM                       'get arm semaph
    SET IO[ioComplate]
    MOVE P, P[pHome], S_50        'move to home p
    SPEED 100
    APPROACH P,P[pPick],200
END
```

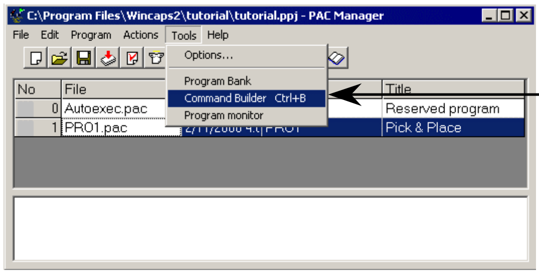
③ Input the "Pick & Place" source code.

## 6.4 Using the command builder

You may input source code to the program edit window by using the keyboard, just as with a word processor. However, the Program Manager is provided with the command builder function, allowing you to enter commands with ease. This section describes how to enter commands using the command builder.

### Step 1

**Selecting the command builder**

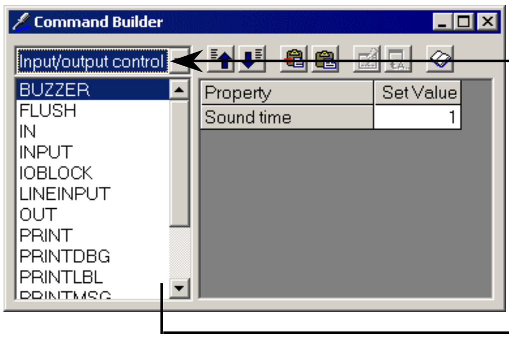


| No | File | Title |
|---|---|---|
| 0 | Autoexec.pac | Reserved program |
| 1 | PRO1.pac | Pick & Place |

① Select [Command Builder] from the [Tools] menu.

### Step 2

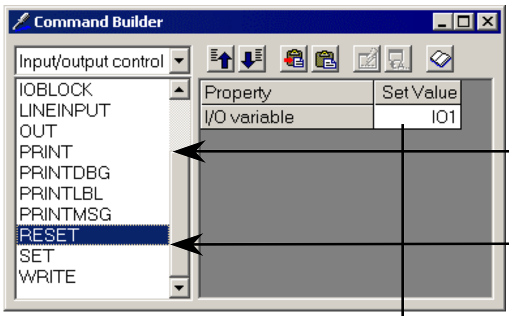
**Using the command builder**



| Property | Set Value |
|---|---|
| Sound time | 1 |

② Click on the [Class Selection] list in the [Command Builder] window and select "Input/output control" statements from the list.

Command list

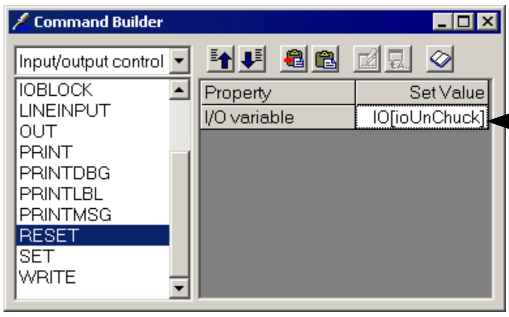


| Property | Set Value |
|---|---|
| I/O variable | IO1 |

③ Using the scroll bar in the Input/output control statements, scroll down to display the [RESET] command.

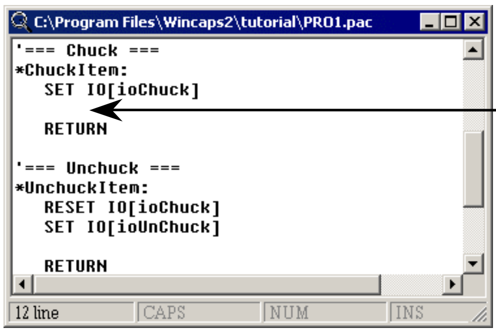
④ Click on the [RESET] command.

⑤ Double click on the [I/O variable] Set Value box.



| Property | Set Value |
|---|---|
| I/O variable | IO[ioUnChuck] |

⑥ Type "ioUnChuck" in the [I/O variable] Set Value box using the keyboard.

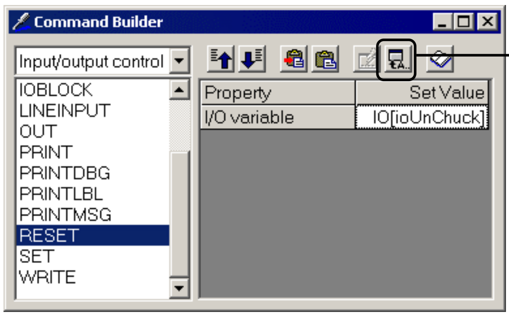


C:\Program Files\Wincaps2\tutorial\PRO1.pac

```
'=== Chuck ===
*ChuckItem:
   SET IO[ioChuck]

   RETURN

'=== Unchuck ===
*UnchuckItem:
   RESET IO[ioChuck]
   SET IO[ioUnChuck]

   RETURN
```

12 line  CAPS  NUM  INS

⑦ In the program edit window, click on the position where you want to enter commands.

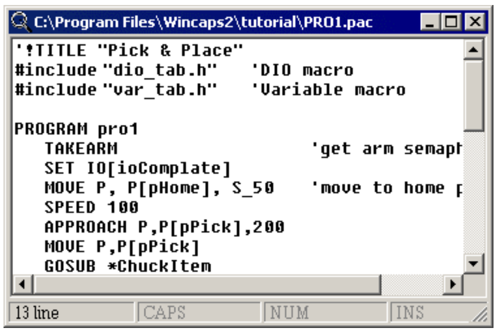


Command Builder

| Property | Set Value |
|---|---|
| I/O variable | IO[ioUnChuck] |

⑧ Click on the [Paste] button .

**Step 3** **Editing source code for ease of clarity**



C:\Program Files\Wincaps2\tutorial\PRO1.pac

```
'!TITLE "Pick & Place"
#include "dio_tab.h"    'DIO macro
#include "var_tab.h"    'Variable macro

PROGRAM pro1
   TAKEARM                    'get arm semapł
   SET IO[ioComplate]
   MOVE P, P[pHome], S_50     'move to home p
   SPEED 100
   APPROACH P,P[pPick],200
   MOVE P,P[pPick]
   GOSUB *ChuckItem
```

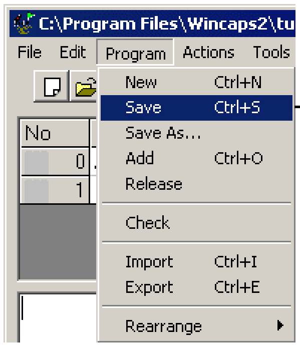
13 line  CAPS  NUM  INS

**★Caution★** To save the edited program, proceed to the procedure described in the following section "6.5 Saving the Program".

## 6.5 Saving the program

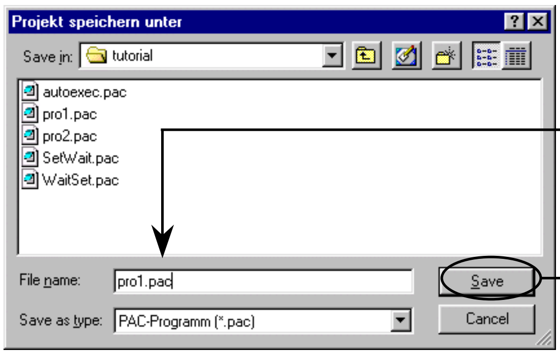
In this section, you will save PRO1, the program created in the previous section "6.4 Using the command builder".

**Step 1** **Selecting [Save]**



① Select [Save] from the [Program] menu.

**Step 2** **Entering the file name**



② Enter the file name in the [File name] box.
In this example, enter "pro1" as the file name.
(The extension is attached automatically.)

③ Click [Save]. The program source file "pro1.pac" is saved.

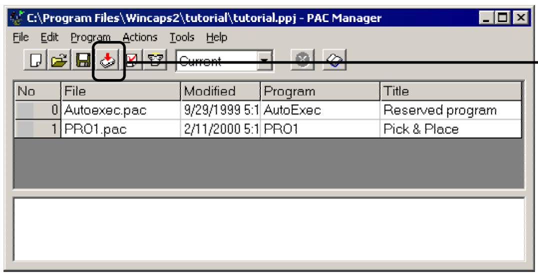
The file name of the [PAC Manager] window changes to "pro1.pac". This completes the inputting and saving procedure of program "pro1".

# Lesson 7  Compiling the Program into Run-time Format

To execute a program written in PAC language, it is necessary to convert (compile) it into run-time format so it is executable by the robot controller. The compiled program is referred to as an execution program.

**Step 1**

**Compiling the program into run-time format**


C:\Program Files\Wincaps2\tutorial\tutorial.ppj - PAC Manager
File Edit Program Actions Tools Help
Current

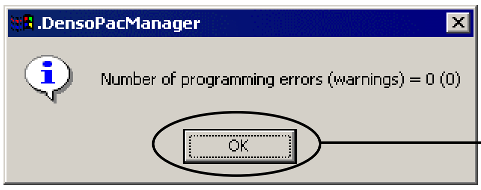
| No | File | Modified | Program | Title |
|---|---|---|---|---|
| 0 | Autoexec.pac | 9/29/1999 5:1 | AutoExec | Reserved program |
| 1 | PRO1.pac | 2/11/2000 5:1 | PRO1 | Pick & Place |

① Click on the [Make Exec. Program] button .

All the programs included in the currently selected project are converted to execution programs. The record of the compiling process is displayed in the message pane of the [DensoPACManager] window.

**Step 2**

**Checking that no error has occurred**



Number of programming errors (warnings) = 0 (0)

② Check that "Number of programming errors (warnings) = 0 (0)" is displayed and click [OK].

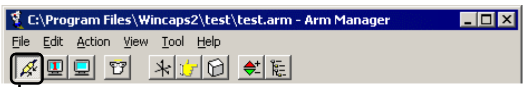
If an error is showing, return to "Lesson 6 Inputting and Editing Programs" and check for syntax errors.

# Lesson 8  Uploading the Program (PC → Robot controller)

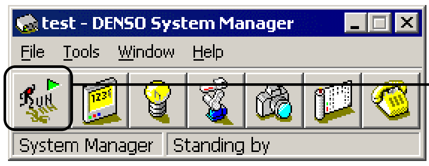
At present, the execution program complied in Lesson 7 is still in the PC. To run the program, it is necessary to transmit (upload) it to the robot controller.

Since you have already made communications settings on both the robot controller and PC in Lesson 3.3 and Lesson 4.3, respectively, you may now upload the program to the robot controller.

## Step 1 Establishing communications link between the Program Manager and robot controller

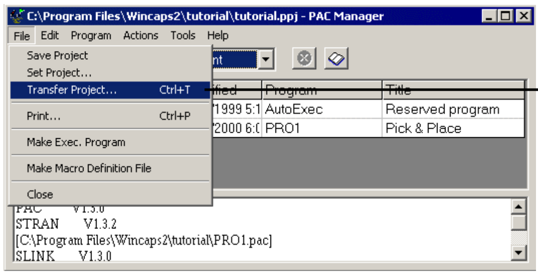

C:\Program Files\Wincaps2\test\test.arm - Arm Manager
File Edit Action View Tool Help

① Click on the [Connections] button .



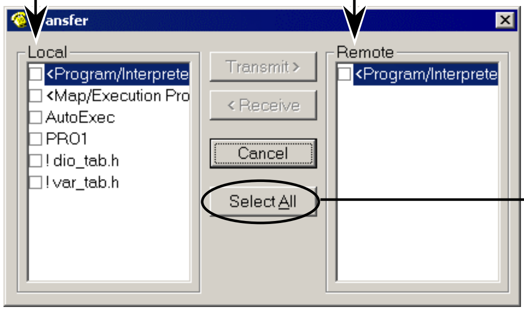
In System Manager, the icon of the [PAC Program Manager] button changes to this.

## Step 2 Selecting the program to be uploaded


C:\Program Files\Wincaps2\tutorial\tutorial.ppj - PAC Manager
File Edit Program Actions Tools Help
Save Project
Set Project...
Transfer Project... Ctrl+T
Print... Ctrl+P
Make Exec. Program
Make Macro Definition File
Close
Program
Title
AutoExec
Reserved program
PRO1
Pick & Place

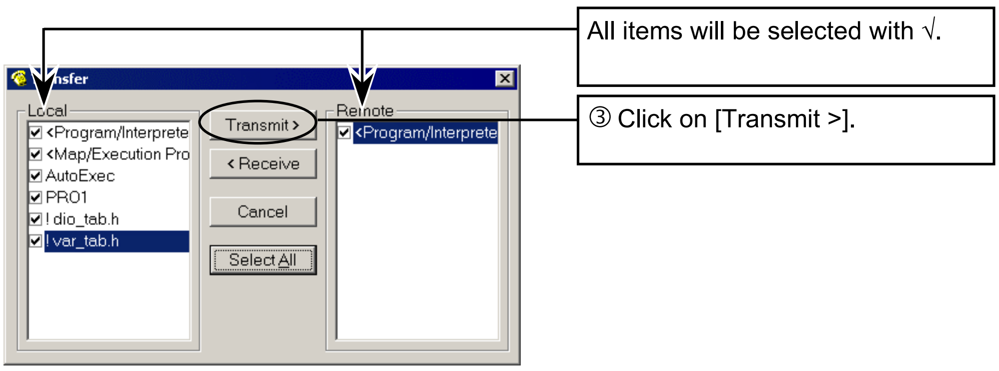
```
PAC      V1.3.0
STRAN     V1.3.2
[C:\Program Files\Wincaps2\tutorial\PRO1.pac]
SLINK     V1.3.0
```

① Select [Transfer Project…].

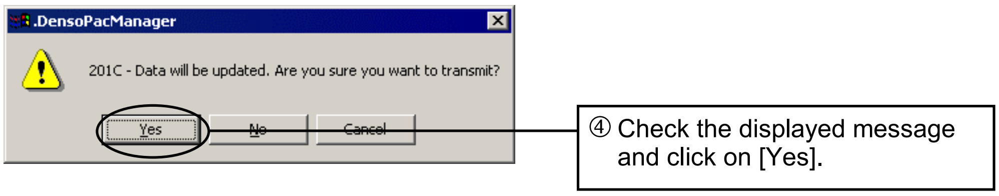


"Local" refers to the PC side and "Remote" to the robot controller side.

② Click on [Select All].



**Step 3** **Uploading the selected program**



201C - Data will be updated. Are you sure you want to transmit?

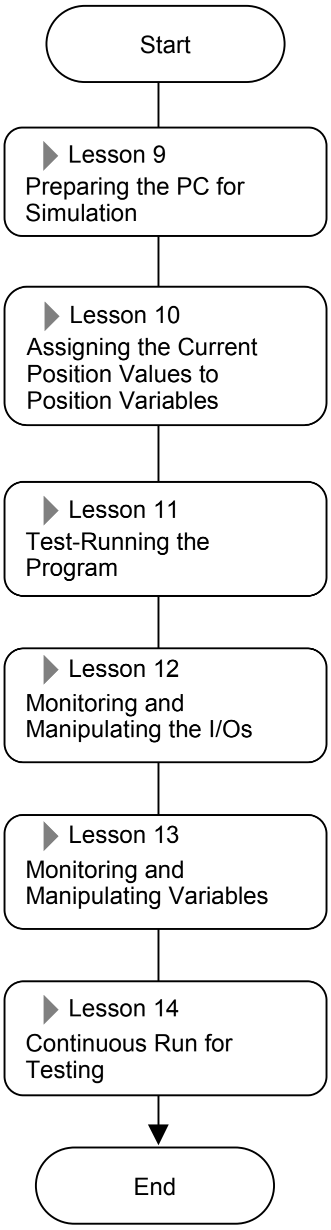
The program file is now uploaded to the robot controller.

# Part 3 Simulating the Robot Motion on a PC with the Program Created

In Part 3, you will:

Run the program, which you have created on a PC and uploaded to the robot controller, in machine lock in order to simulate the robot motion on the PC screen.

Simulation allows you to verify the program before actually running the robot, helping you improve safety and the efficiency of program development.


Start
Lesson 9
Preparing the PC for Simulation
Lesson 10
Assigning the Current Position Values to Position Variables
Lesson 11
Test-Running the Program
Lesson 12
Monitoring and Manipulating the I/Os
Lesson 13
Monitoring and Manipulating Variables
Lesson 14
Continuous Run for Testing
End

• • • You will start up WINCAPSII Arm Manager, which is necessary in order to simulate the robot motion on the PC screen.

• • • You will assign the values to the position variable representing where the robot arm is to move to.

(1) Move the robot arm to the specified position with manual operation.
(2) Read in the current position to the position variable.
(3) Modify the value assigned to the position variable if necessary.

• • • Start a single-cycle run and check the robot motion.

• • • Continue the program to the next step by manipulating the I/Os being used by the interlock in the program.

• • • Check the contents of the variables being used by the program.

• • • Run the program with Continuous Run.

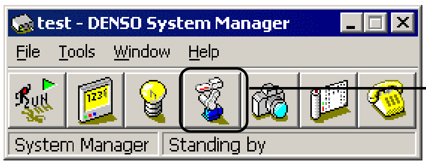
# Lesson 9 Preparing the PC for Simulation

You will start Arm Manager and establish the communications link with the robot controller.

## 9.1 Starting Arm Manager

You need to start Arm Manager in order to display the simulated robot images. The Arm Manager is called up from the System Manager.

**Step 1**

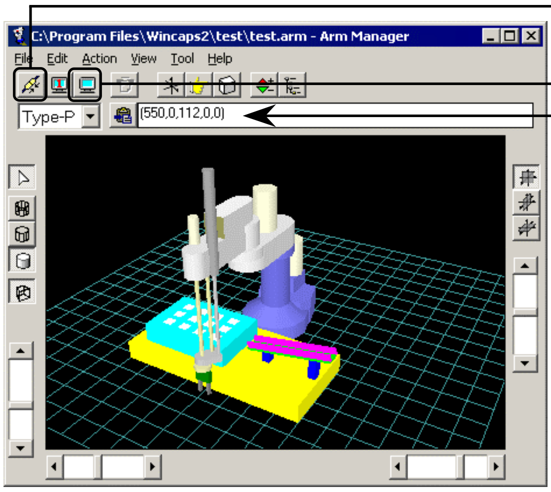


① Click the [Arm] button.

## 9.2 Establishing the communications link with the robot controller in Arm Manager

You establish the communications link with the robot controller so that the PC may always exchange data with the robot controller

**Step 1**


C:\Program Files\Wincaps2\test\test.arm - Arm Manager
File Edit Action View Tool Help
Type-P
(550,0,112,0,0)

① Click on the [Connect] button.
(The [Connect] button appears depressed.)

② Click on the [Monitor] button.
(The [Monitor] button appears depressed.)

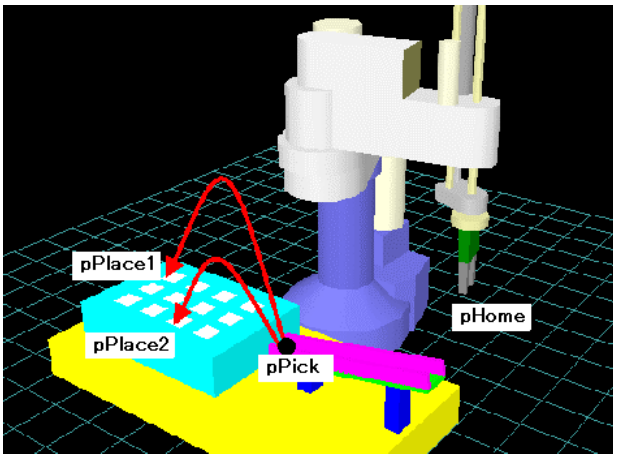
The communications link is established between Arm Manager and the robot controller, enabling data exchange between them.

The current robot position is displayed in the [Current Robot Position] window.

# Lesson 10 Assigning the Current Position Values to Position Variables

Before running the program, it is necessary to determine the values to be assigned to the respective position variables "pHome", "pPick", "pPlace1" and "pPlace2" for the program which you created in "Lesson 6 Entering and Editing Programs" of Part 2 "Creating a Program on a PC in WINCAPSII".

In this lesson you will enter the values from the teach pendant through "Point Teaching".

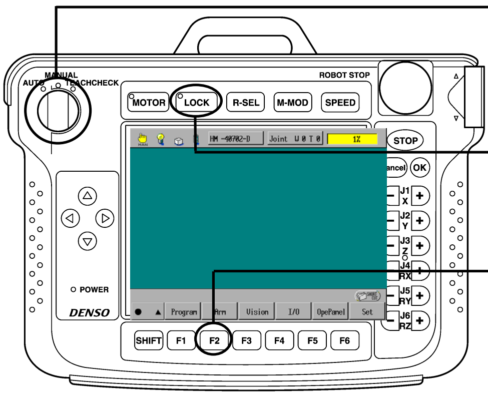


**★Caution★** Refer to "Lesson 5 Defining Macros" for each of the position variable numbers for "pHome", "pPick", "pPlace1" and "pPlace2".

## 10.1 Simulating the robot motion manually

While monitoring the simulation images displayed in Arm Manager, move the robot arm manually to the position values assigned to the "pHOme" position variable, according to the procedure given below.

**Step 1** **<u>Placing the robot in Manual mode and displaying the current robot position</u>**



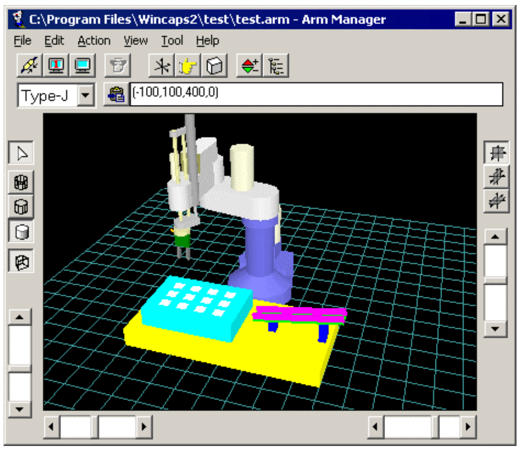
① Set the mode selector switch to the MANUAL position to switch to Manual mode.

② Press [LOCK].

③ Press [F2 Arm] on the top screen.

The Current Robot Position window appears where you should switch to the joint variables window.

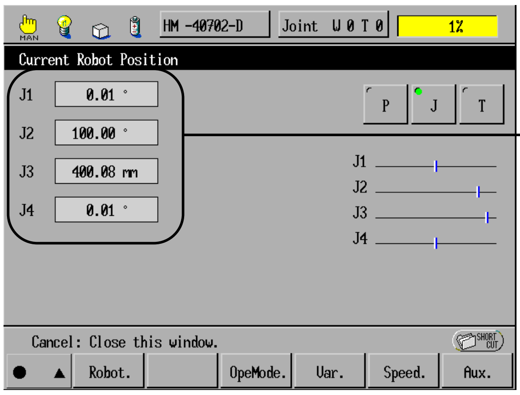
## Step 2 Moving the robot arm


C:\Program Files\Wincaps2\test\test.arm - Arm Manager
File Edit Action View Tool Help
Type-J
(-100,100,400,0)

④ While monitoring the robot simulation image, move the robot arm to "pHome" using the deadman switch and the arm traverse keys so that the following values will apply:

- J1: Approx. 0 Deg.
- J2: Approx. 100 Deg.
- J3: Approx. 400 mm
- J4: 0.00 Deg. (no move)

## Step 3 Checking the current position



| | |
|---|---|
| J1 | 0.01 ° |
| J2 | 100.00 ° |
| J3 | 400.08 mm |
| J4 | 0.01 ° |

⑤ Check the position of each axis in the [Current Robot Position window].

## 10.2 Getting the current position into a position variable

Let's get the current position, to which the robot arm has moved in Arm Manager, into a position variable.

There are the following three ways to get the current position into a position variable:

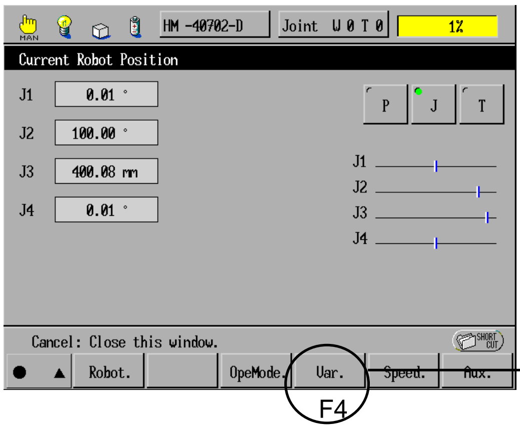
[ 1 ] Using the [Position Variables] window on the teach pendant

[ 2 ] Using the program edit window (coding list) on the teach pendant, in which you get the current position into a position variable defined in the specified program

[ 3 ] Using the operating panel

### [ 1 ] Using the [Position Variables] window on the teach pendant

**Step 1** <u>**Calling up the [Position Variables] window**</u>



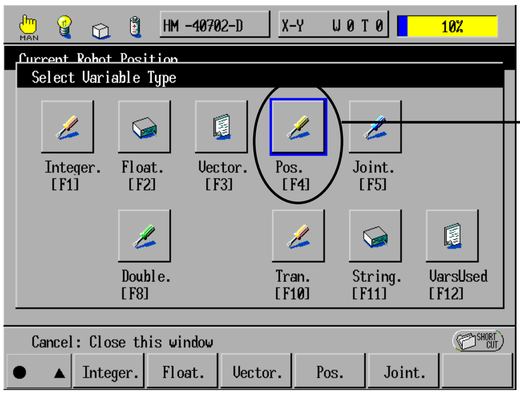
HM -40702-D | Joint U 0 T 0 | 1%

Current Robot Position

| | |
|---|---|
| J1 | 0.01 ° |
| J2 | 100.00 ° |
| J3 | 400.08 mm |
| J4 | 0.01 ° |

Cancel: Close this window.

① Press [F4 Var.] with the [Current Robot Position] window displayed.



HM -40702-D | X-Y U 0 T 0 | 10%

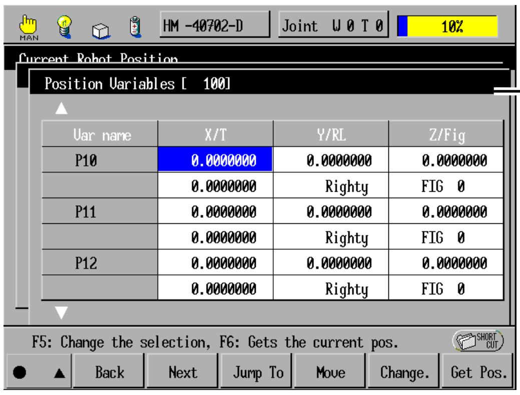
Current Robot Position

Select Variable Type

Cancel: Close this window

F4

② On the [Select Variable Type] window, press [F4 Pos.].



HM -40702-D | Joint U 0 T 0 | 10%

Current Robot Position

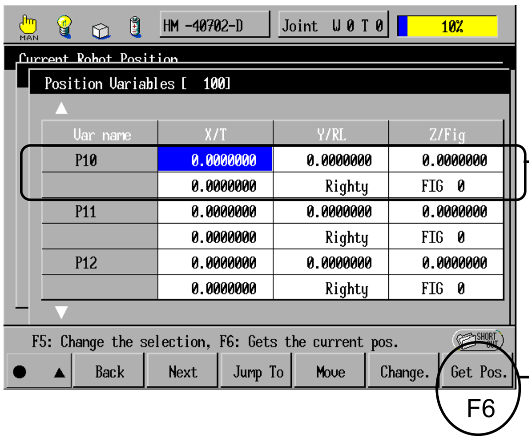
Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P10 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P11 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P12 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

F5: Change the selection, F6: Gets the current pos.

The [Position Variables] window appears.

## Step 2 Getting the current position into a position variable


HM -40702-D Joint W 0 T 0 10%

Current Robot Position

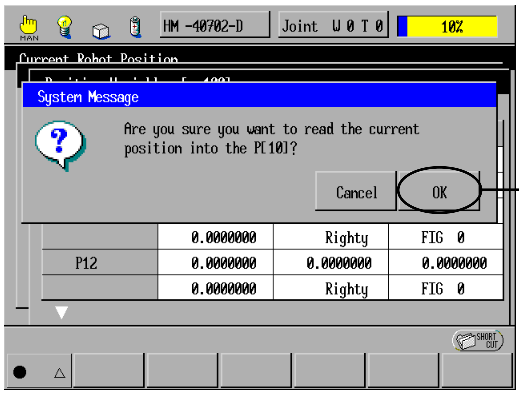
Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P10 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P11 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P12 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

F5: Change the selection, F6: Gets the current pos.

③ Using the jog dial or cursor keys, select "P10" box.

④ Press [F6 Get Pos.].


HM -40702-D Joint W 0 T 0 10%

Current Robot Position

System Message

Are you sure you want to read the current position into the P[10]?

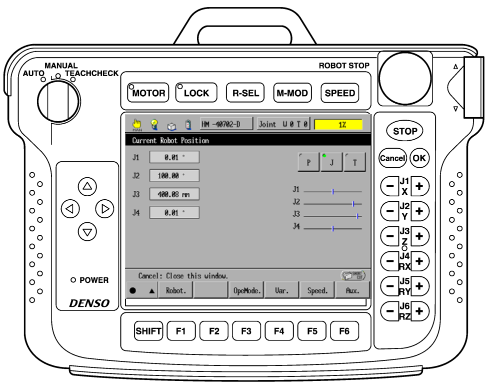
| | | | |
|---|---|---|---|
| | 0.0000000 | Righty | FIG 0 |
| P12 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

⑤ Press [OK].
This gets the current position of the robot arm into position variable P[10].

⑥ Press [Cancel] twice.
The display returns to the [Robot Current Position] window.

⑦ Go back to Step 2 "Moving the robot arm" in Lesson 10.1.

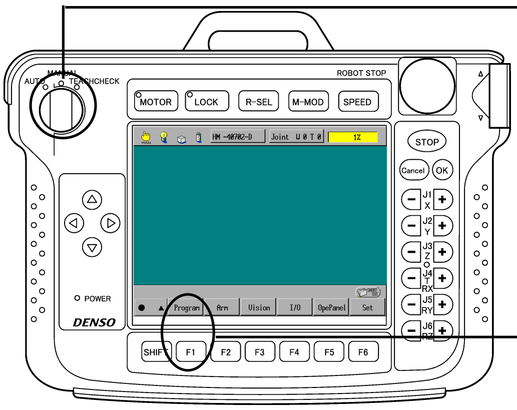
## Step 3 Getting other arm positions into position variables



As in Steps 1 and 2, get other arm positions.

## [ 2 ] Using the program edit window (coding list) on the teach pendant, in which you get the current position into a position variable defined in the specified program

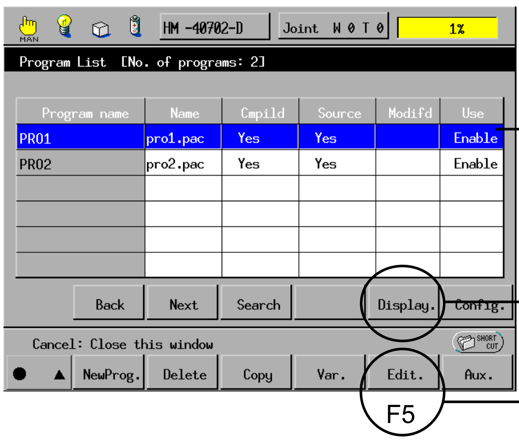
### Step 1 Calling up the Program List window



① Set the mode selector switch MANUAL.

② On the top screen, press [F1 Program] to call up the [Program List] window.

### Step 2 Selecting the target program



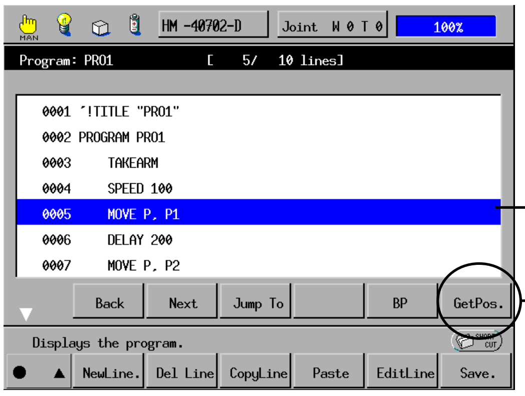
```
HM -40702-D   Joint  W 0 T 0   1%
Program List  [No. of programs: 2]
```

| Program name | Name | Cmpild | Source | Modifd | Use |
|---|---|---|---|---|---|
| PRO1 | pro1.pac | Yes | Yes | | Enable |
| PRO2 | pro2.pac | Yes | Yes | | Enable |

③ Select the target program.

④ Press [Display.] or [F5 Edit.] to show the coding list.

### Step 3 Selecting the program line containing the target variable



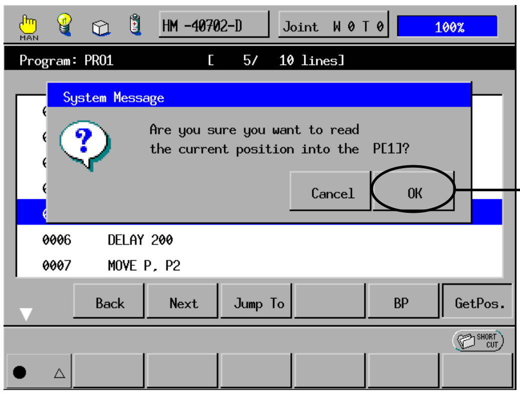
```
HM -40702-D   Joint  W 0 T 0   100%
Program: PRO1              [    5/   10 lines]

0001 '!TITLE "PRO1"
0002 PROGRAM PRO1
0003     TAKEARM
0004     SPEED 100
0005     MOVE P, P1
0006     DELAY 200
0007     MOVE P, P2
```

⑤ Select the line containing the target variable.

⑥ Press [GetPos.].

## Step 4 Getting the current position into a variable contained in the line

If the program line contains a single variable candidate, the system message will appear as shown below.



Program: PRO1 [ 5/ 10 lines]

System Message

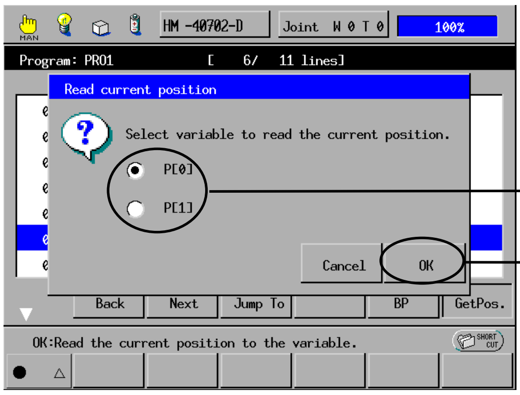
Are you sure you want to read the current position into the P[1]?

Cancel OK

0006 DELAY 200
0007 MOVE P, P2

⑦ Press [OK]. This gets the current position into the variable.

If the program line contains more than one variable candidate, the system message will appear as shown below. Select the desired variable.



Program: PRO1 [ 6/ 11 lines]

Read current position

Select variable to read the current position.

● P[0]
○ P[1]

Cancel OK

OK:Read the current position to the variable.

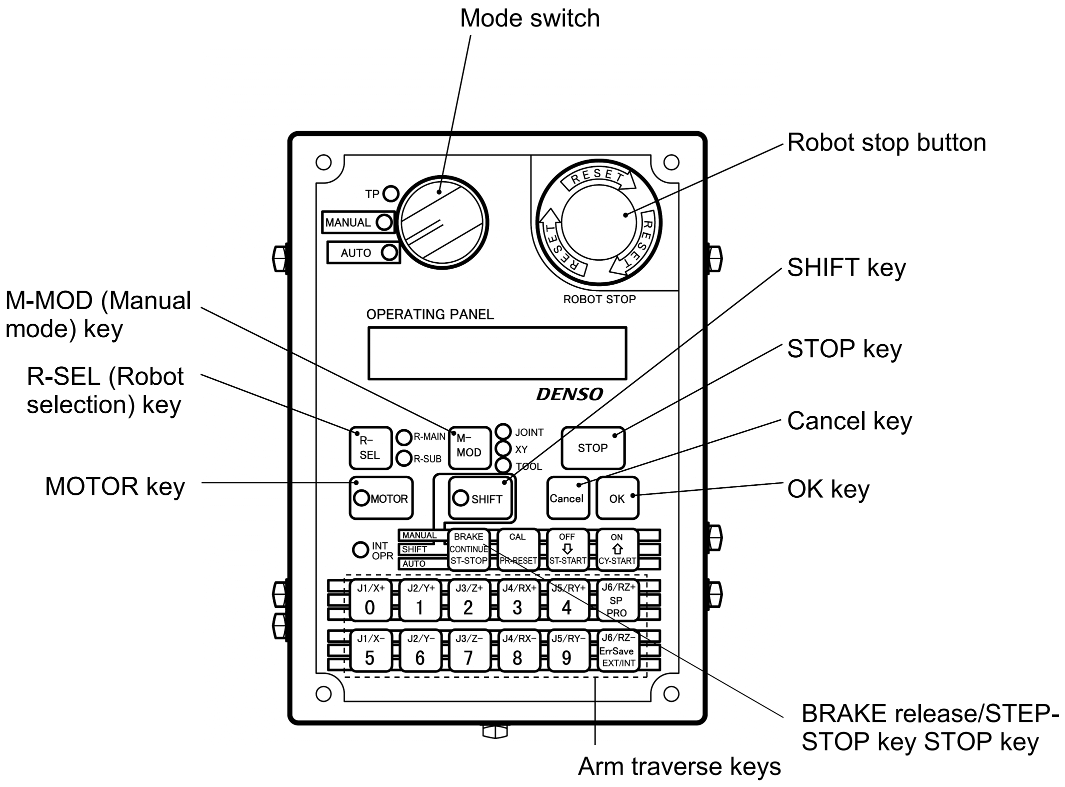
⑧ Select the desired variable by using the cursor keys or jog dial.

⑨ Press [OK]. This gets the current position into the selected variable.

If a variable(s) contained in the selected program line is not appropriate, no variable will be displayed as a candidate for teaching.

Only the heading three variables in the program line will be displayed as a candidate.

## [ 3 ] Using the operating panel



**Step 1** **Switching to the Manual mode**

Turn the mode switch to the MANUAL position and check that its LED comes on.

**Step 2** **Entering the function selection mode**

Press the SHIFT key and check that its LED comes on.

Then press the BRAKE release/STEP-STOP key to make the operating panel enter the function selection mode.

**Step 3** **Choosing the get-position function**

Use the ⇧ and ⇩cursor keys to choose the desired function.

| To get the current position into: | Choose: |
|---|---|
| Position variable | [F7: Set CurPos P] |
| Joint variable | [F8: Set CurPos J] |
| Trans. variable | [F9: Set CurPos T] |

After choosing the desired function, press the OK key.

**Step 4**

### Choosing a target variable number

Use the numerical keys to enter a variable number into which you want to get the current position.

Press the OK key. (To cancel the number you entered, press the Cancel key. The LCD returns to Step 3.)

**Step 5**

### Getting the current position

The LCD shows [SetCurrentPos?].

Press the OK key. (To cancel the operation, press the Cancel key. The LCD returns to Step 4.)

Upon completion of getting of the current position, the LCD shows [Set VarVal OK].

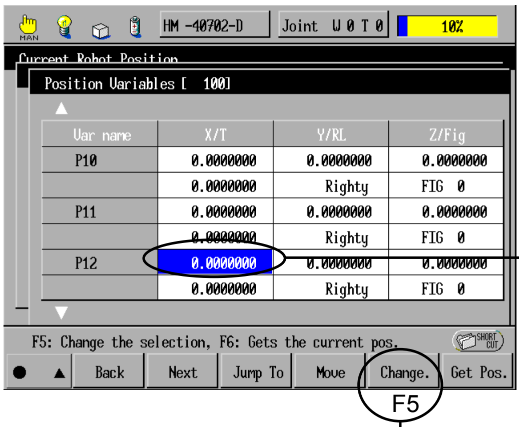
## 10.3 Editing position variables

You may edit position variables as required. As a simple example, you will modify the value assigned to a position variable.

The teach pendant displays the Position Variables window as shown below.

**Step 1**

### Selecting Position Variables



Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P10 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P11 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P12 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

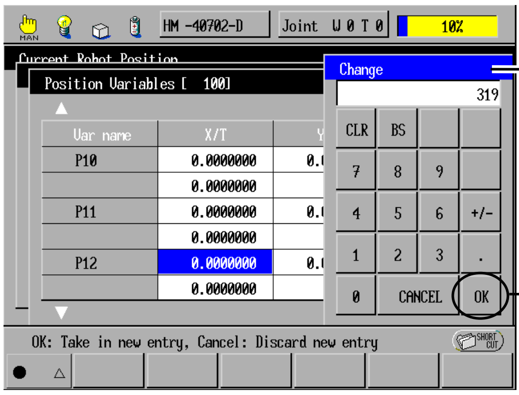
F5: Change the selection, F6: Gets the current pos.

① Using the jog dial or the cursor keys, select the [X/T] column of position variable "P12".

② Press [F5 Change.].
The numeric keypad appears as shown on the next page.

**Step 2**

### Modifying the value assigned to "P12"



Position Variables [ 100]

| Var name | X/T | Y |
|---|---|---|
| P10 | 0.0000000 | 0. |
| | 0.0000000 | |
| P11 | 0.0000000 | 0. |
| | 0.0000000 | |
| P12 | 0.0000000 | 0. |
| | 0.0000000 | |

Change

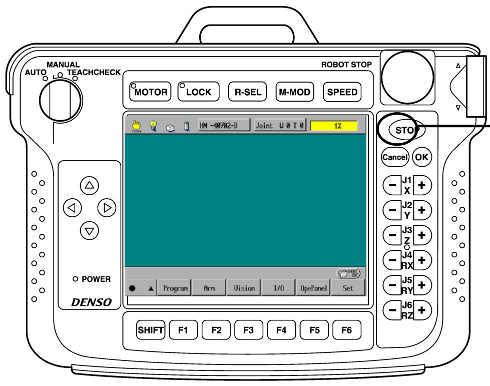
319

OK: Take in new entry, Cancel: Discard new entry

① Using the numerical keys on the numeric keypad, enter a new value.
As an error will result if you enter too different a value, enter "319" which is close to the original value.

② Press [OK].
"319" is entered into the [X/T] column of position variable "P12".

### Returning to the top screen



③ Press [Cancel] 3 times to return to the top screen.
This completes the editing of position variables.

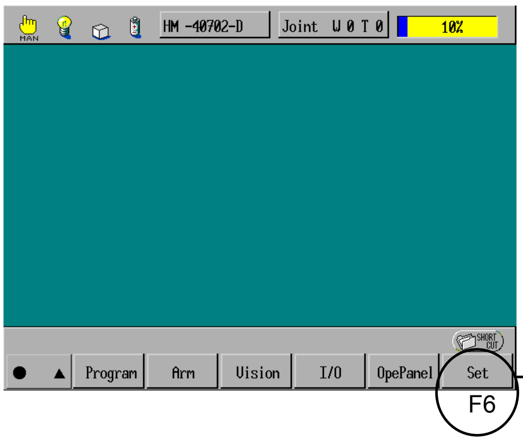
# Lesson 11 Test-running the Program

You will test-run program "PRO1" by a single-cycle run.

## 11.1 Loading the program

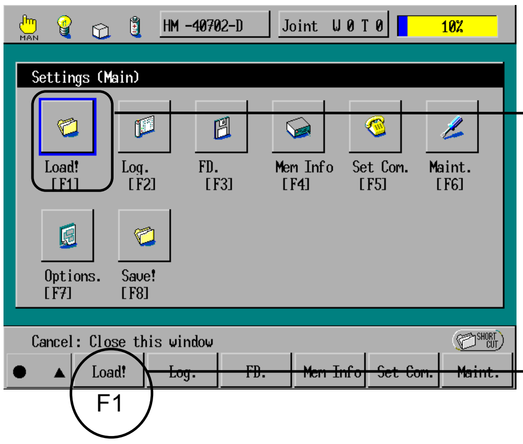
Even if you load a compiled program from the PC to the robot controller, the controller cannot execute it. The program needs to be loaded to the memory area where it can be executed.

**Step 1** **Displaying the load screen**

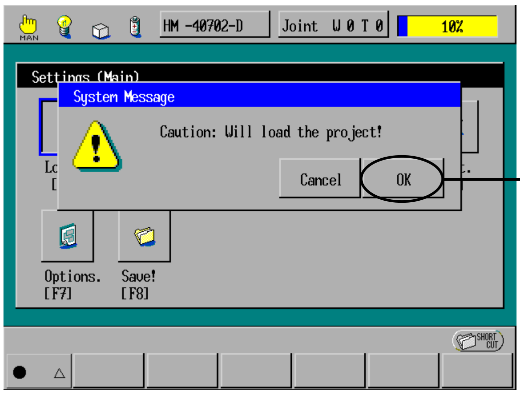


① Press [F6 Set] in the top screen.

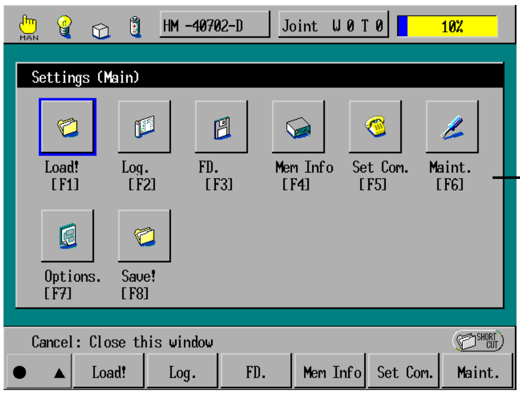
**Step 2** **Loading the project**



② Press either the [F1 Load!] button or [Load!].

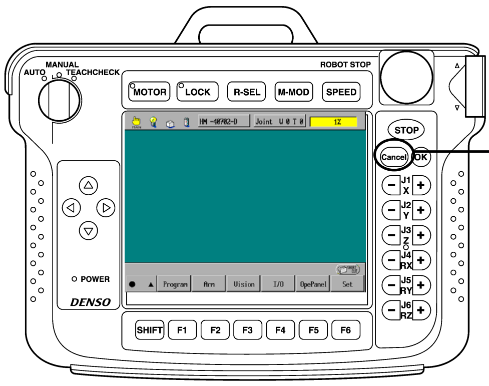


③ Check the system message and press [OK].



Upon completion of loading, the screen returns to the [Setting (Main)] window.

## Returning to the top screen



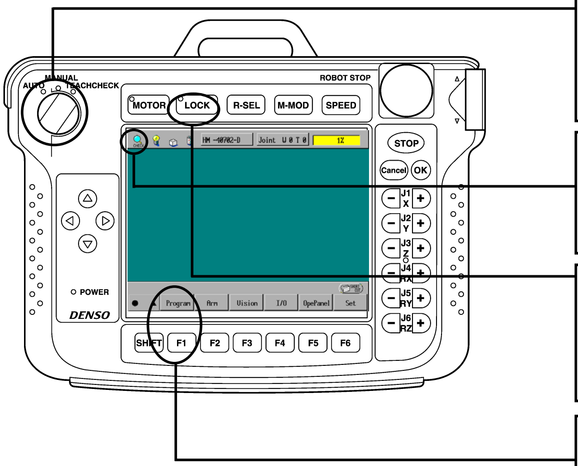
⑤ Press [Cancel] to return to the top screen.

## 11.2 Starting the program

Now you will place the robot controller in machine lock and start the loaded program by a single-cycle run in order to simulate the robot motion on the PC screen.

The program will come to a halt and wait for part supply confirmation I/O signals; however, proceed to the following step as you will manipulate I/Os in Lesson 12.

### Step 1 Displaying the Program List window in Teach Check mode



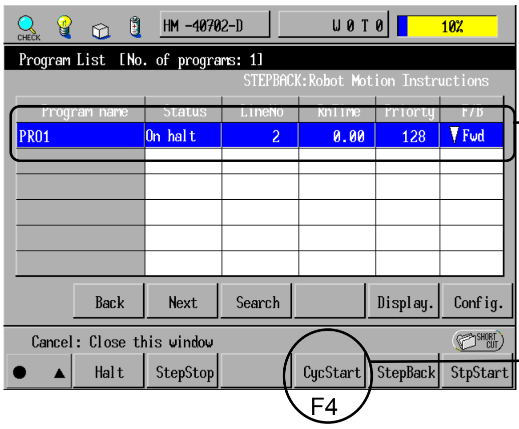
① Set the mode selection switch to the TEACHCHECK position.
(The robot controller changes to Teach Check mode.)

② Check that the mode icon at the top left of the screen has changed to [CHECK].

③ Press [LOCK] to place the robot in machine lock. Check that the [LOCK] LED has come on.

④ Press [F1 Program] to display the [Program List] window.

### Step 2 Selecting the program to start it



HM -40702-D　　U 0 T 0　　10%

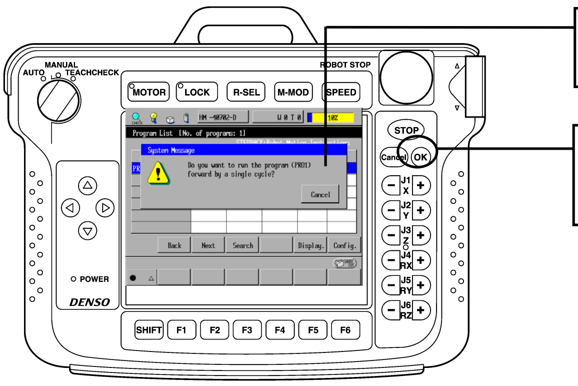
Program List [No. of programs: 1]

STEPBACK:Robot Motion Instructions

| Program name | Status | LineNo | RnTime | Priority | F/B |
|---|---|---|---|---|---|
| PRO1 | On halt | 2 | 0.00 | 128 | Fwd |

⑤ Select the "PRO1" line.

⑥ Press [F4 CycStart].



⑦ Check the system message.

⑧ Press [OK].
"PRO1" runs a single-cycle.

# Lesson 12 Monitoring and Manipulating the I/Os

Cycle-started program "PRO1" is now on halt, waiting for the macro I/O "ioParts" for confirming parts supply.

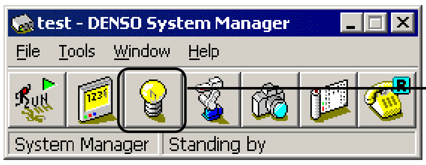
In this lesson, to test the program, you will use the I/O Manager for monitoring and manipulating the I/Os to continue program execution.

**Note:** Refer to "Lesson 5 Defining Macros" for the I/O numbers of "ioParts".

## 12.1 Starting the DIO Manager and establishing the communications link with the robot controller

You will start the DIO Manager and establish the communications link with the robot controller so that the PC may always exchange data with the robot controller.

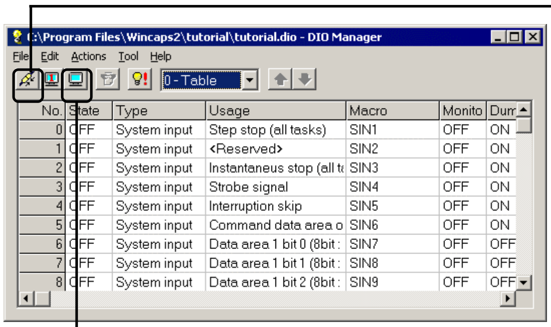
**Step 1** **Starting the DIO Manager**



① Click on the [I/O] button in the DENSO System Manager window.

The [DIO Manager] window appears.

**Step 2** **Connecting with the robot controller to start continuous monitoring**


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
0 - Table

| No. | State | Type | Usage | Macro | Monito | Dum |
|---|---|---|---|---|---|---|
| 0 | OFF | System input | Step stop (all tasks) | SIN1 | OFF | ON |
| 1 | OFF | System input | <Reserved> | SIN2 | OFF | ON |
| 2 | OFF | System input | Instantaneus stop (all t | SIN3 | OFF | ON |
| 3 | OFF | System input | Strobe signal | SIN4 | OFF | ON |
| 4 | OFF | System input | Interruption skip | SIN5 | OFF | ON |
| 5 | OFF | System input | Command data area o | SIN6 | OFF | ON |
| 6 | OFF | System input | Data area 1 bit 0 (8bit : | SIN7 | OFF | OFF |
| 7 | OFF | System input | Data area 1 bit 1 (8bit : | SIN8 | OFF | OFF |
| 8 | OFF | System input | Data area 1 bit 2 (8bit : | SIN9 | OFF | OFF |

② Click the [Connect] button in the DIO Manager window.

The [Connect] button appears depressed.

③ Click on the [Monitor] button.

The communications link is established between the DIO Manager and robot controller, enabling data exchange between them.

The [Monitor] button appears depressed.

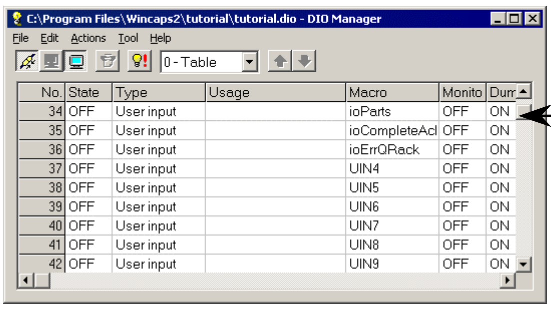
## 12.2 Monitoring the I/Os

You will monitor the I/Os with the DIO Manager.

The DIO Manager can show the I/O status in three types of display formats--table type, oscilloscope type, and panel type. In this lesson, use the table type display to check "ioParts" on I/O No. 34.

The table type display is just like a list and appears initially when you open the [DIO Manager] window. If the display is in any other display type, select the table type from the [Variable Scope] list.

### Step 1 Displaying the I/Os

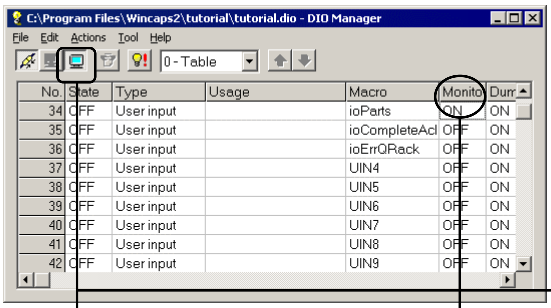

C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
0 - Table

| No. | State | Type | Usage | Macro | Monito | Durr |
|---|---|---|---|---|---|---|
| 34 | OFF | User input | | ioParts | OFF | ON |
| 35 | OFF | User input | | ioCompleteAcl | OFF | ON |
| 36 | OFF | User input | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

① Scroll down the [DIO Manager] window to display I/O No. 34.

(Use the scroll bar at the right side of the window to scroll.)

### Step 2 Setting the monitor type


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
0 - Table

| No. | State | Type | Usage | Macro | Monito | Durr |
|---|---|---|---|---|---|---|
| 34 | OFF | User input | | ioParts | ON | ON |
| 35 | OFF | User input | | ioCompleteAcl | OFF | ON |
| 36 | OFF | User input | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

② Click the [Continuous Monitor] button in the [DIO Manager] window to set the monitor to OFF. (It is not possible to change the set contents while the monitor is set to ON.)

③ Double-click on the [Monitor] box of I/O No. 34 that is displayed in the DIO Manager window.

The setting in the [Monitor] box changes from "OFF" to "ON" making it now possible to monitor I/O No. 34.

④ Click on the [Continuous Monitor] button to set the monitor to ON.

The "State" of I/O No. 34 is displayed as "ON" or "OFF", according to the changes made to the I/O.

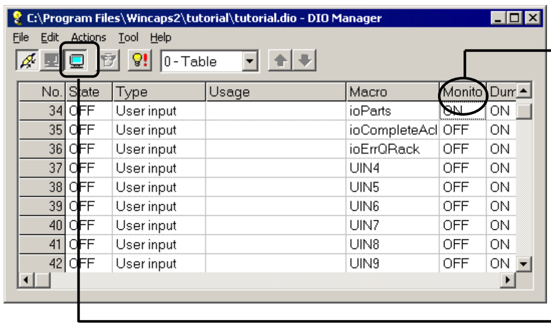
In this example, the "ioParts" for I/O No. 34 is displayed as "OFF".

## 12.3 Turning the I/O dummy switches ON/OFF

In DIO Manager, you may falsely turn the I/Os on or off.

The program PRO1 "PICK & PLACE" is waiting for I/O No. 34 "ioParts" to turn ON and will not proceed to the next step in this status. By managing the I/O falsely, you may perform the operation test.

**Step 1** **Stopping monitoring**

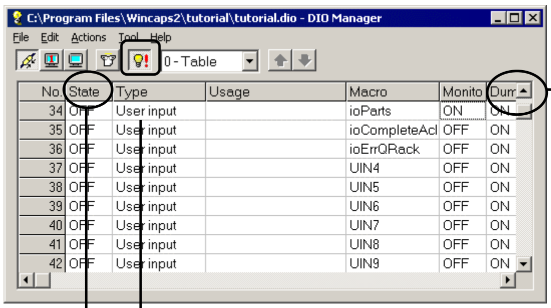

C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File
Edit
Actions
Tool
Help
0 - Table

| No. | State | Type | Usage | Macro | Monito | Dum |
|---|---|---|---|---|---|---|
| 34 | OFF | User input | | ioParts | ON | ON |
| 35 | OFF | User input | | ioCompleteAcl | OFF | ON |
| 36 | OFF | User input | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

① In the [DIO Manager] window, Check that the [Monitor] box for I/O No. 34 is set to "ON".

② Click here to set the [Continuous Monitor] to "OFF".

**Step 2** **Setting the dummy switch**


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File
Edit
Actions
Tool
Help
0 - Table

| No. | State | Type | Usage | Macro | Monito | Dum |
|---|---|---|---|---|---|---|
| 34 | OFF | User input | | ioParts | ON | ON |
| 35 | OFF | User input | | ioCompleteAcl | OFF | ON |
| 36 | OFF | User input | | ioErrQRack | OFF | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

③ Double-click the [Dummy SW] box of I/O No. 34 in the [DIO Manager] window to set it "ON".

④ To enable the dummy I/O:

Click the [Dummy I/O] button in the DIO Manager window.

The [Dummy I/O] button appears depressed.

⑤ Double-click the [State] box of I/O No. 34 to set it "ON".

The program recognizes I/O No. 34 "ioParts" as being "ON", so it proceeds onto the next step. If no problem is detected, the program will run through to the end and stop.

# Lesson 13 Monitoring and Manipulating Variables

The Variable Manager allows you to monitor variables.

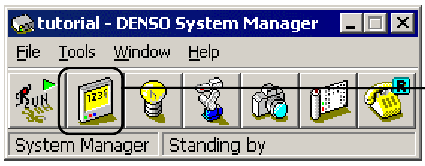
In this lesson, you will examine the integer variable [0] of macro "iParts", which is used in the operation count.

**NOTE:** Refer to "Lesson 5 Defining Macros" for the variable number 0 of "ioParts".

## 13.1 Starting the Variable Manager and establishing the communications link with the robot controller

Start the Variable Manager and connect communications so that data exchange is constantly performed with the robot controller.

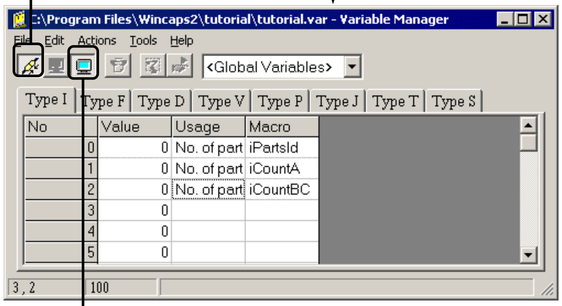
### Step 1 Start the Variable Manager



① Click on the Variable button in the [DENSO System Manager] window.

### Step 2

The [Variable Manager] window appears.



C:\Program Files\Wincaps2\tutorial\tutorial.var - Variable Manager
File Edit Actions Tools Help
<Global Variables>
Type I
Type F
Type D
Type V
Type P
Type J
Type T
Type S

| No | Value | Usage | Macro |
|---|---|---|---|
| 0 | 0 | No. of part | iPartsId |
| 1 | 0 | No. of part | iCountA |
| 2 | 0 | No. of part | iCountBC |
| 3 | 0 | | |
| 4 | 0 | | |
| 5 | 0 | | |

3 , 2 | 100

② Click on the [Connect] button to establish the communications link with the robot controller.

The [Connect] button appears depressed.

③ Click on the [Continuous Monitor] button to start the monitor.

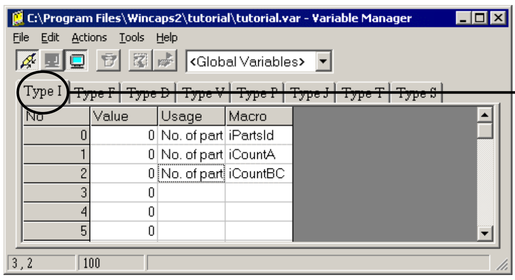
The [Continuous Monitor] button appears depressed.

The above process has established the communications link so that the PC may always exchange data with the robot controller.

## 13.2 Monitoring variables

Establishing the communications link with the robot controller allows you to monitor the variables used in the robot controller.

**Step 1** **Displaying the variables**



C:\Program Files\Wincaps2\tutorial\tutorial.var - Variable Manager

| No | Value | Usage | Macro |
|---|---|---|---|
| 0 | 0 | No. of part | iPartsId |
| 1 | 0 | No. of part | iCountA |
| 2 | 0 | No. of part | iCountBC |
| 3 | 0 | | |
| 4 | 0 | | |
| 5 | 0 | | |

① Scroll down the [Variable Manager] window to display integer variable [0].

To do so, use the scroll bar at the right side of the window.

The [Value] column for integer variable [0] reflects the changes made to the variable.

# Lesson 14 Continuous Run for Testing

Up to the previous lesson, you have checked the PRO1 with a single-cycle run.

Continuous run makes the robot perform the programmed motion repeatedly. If I/O No. 34 "ioParts" is kept at ON with the dummy switch, continuous run is possible even if no actual entry is given to I/O No. 34.

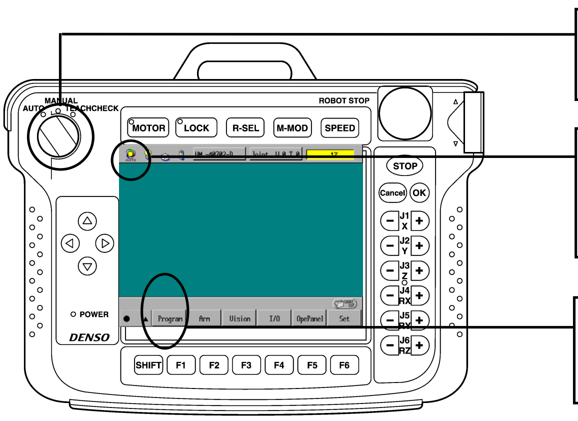
Before actually running the robot, place the robot controller in machine lock and conduct a continuous run for testing while monitoring the simulated robot images in the Arm Manager.

Here we will assume that I/O No. 34 "ioParts" remains ON from the preceding section.

## 14.1 Continuous run

Start the Variable Manager and establish the communications link with the robot controller so that the Variable Manager may always exchange data with the robot controller.

**Step 1** **Displaying the Program List window in Auto mode**

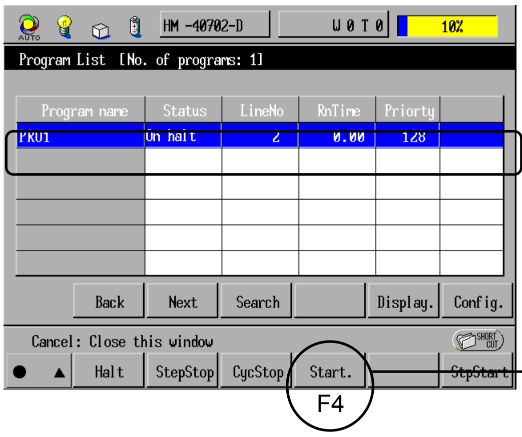


① Set the mode selector switch to the AUTO position.

② Check that the mode icon at the top left of the screen has changed to [ AUTO ].

③ Press [F1 Program] on the top screen to display the [Program List] window.

**Step 2** **Selecting a program to be executed**



```
AUTO  HM -40702-D   U 0 T 0   10%
Program List [No. of programs: 1]
```

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | On halt | 2 | 0.00 | 128 |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

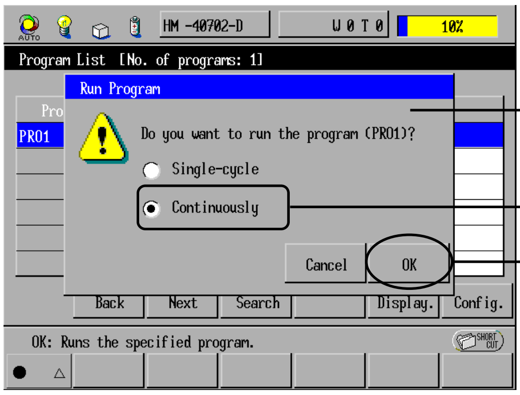
```
Back  Next  Search  Display.  Config.
Cancel: Close this window
●  ▲  Halt  StepStop  CycStop  Start.  StpStart
```

F4

④ Select the "PRO1" line.

⑤ Press [F4 Start.].

**Step 3** **Continuous run**


HM -40702-D
U 0 T 0
10%
Program List [No. of programs: 1]
Run Program
PRO1
Do you want to run the program (PRO1)?
Single-cycle
Continuously
Cancel
OK
Back
Next
Search
Display.
Config.
OK: Runs the specified program.

The "Single-cycle" and "Continuously" selection screen appears.

⑥ Select [Continuously] and press [OK].
"PRO1" will start a continuous run.

## 14.2 Continuous monitoring of the I/Os

In this section, you will monitor the I/O status during the continuous run with the DIO Manager.

The DIO Manager has three types of I/O display formats--table type, oscilloscope type, and panel type.

When the [Monitor] button is pressed, the I/O status with the [Monitor] box "ON" is displayed continuously in real time.

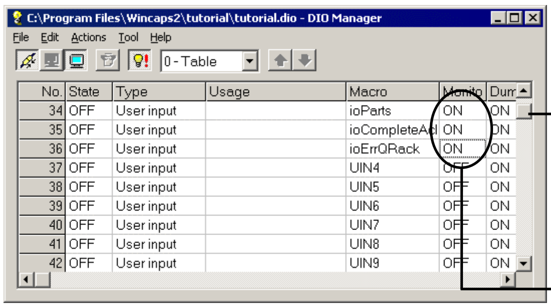
You will monitor the I/Os used in the program.

The DIO Manager that was opened in Lesson 13 should have remained open. If closed, however, click on the [Variable] button in the System Manager to start it.

### Table Type

The table type I/O display format is similar to the list that initially appears when you open the DIO Manager.

If the display is other than the table type, select the table type from the [Display Switch] list.


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File
Edit
Actions
Tool
Help
0 - Table

| No. | State | Type | Usage | Macro | Monito | Dur |
|---|---|---|---|---|---|---|
| 34 | OFF | User input | | ioParts | ON | ON |
| 35 | OFF | User input | | ioCompleteAc | ON | ON |
| 36 | OFF | User input | | ioErrQRack | ON | ON |
| 37 | OFF | User input | | UIN4 | OFF | ON |
| 38 | OFF | User input | | UIN5 | OFF | ON |
| 39 | OFF | User input | | UIN6 | OFF | ON |
| 40 | OFF | User input | | UIN7 | OFF | ON |
| 41 | OFF | User input | | UIN8 | OFF | ON |
| 42 | OFF | User input | | UIN9 | OFF | ON |

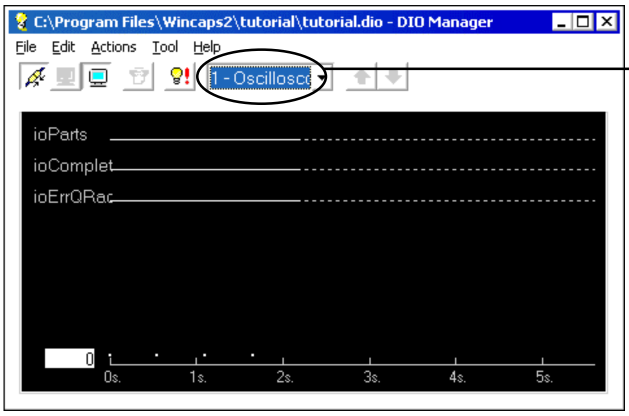
① Scroll down the [DIO Manager] window to display I/O Nos. 34 to 36.
(Use the scroll bar at the right side of the window to scroll.)

② Double-click on the [Monitor] boxes for I/O No.'s 34 to 36 displayed in the [DIO Manager] window to set them to "ON".
The [State] box display changes to "ON" or "OFF" according to the changes made to the I/O.

### Oscilloscope Type

You can visually monitor the I/O status with [Monitor] turned "ON" just as you watch an oscilloscope in the table type display of the [DIO Manager] window.

Proceed with the following operation, continuing on from that described in the previous lesson.


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO Manager
File Edit Actions Tool Help
1 - Oscillosco
ioParts
ioComplet
ioErrQRac
0
0s. 1s. 2s. 3s. 4s. 5s.

① Click on the button and select [1 – Oscilloscope] from the list displayed.

The statuses of I/O Nos. 34 to 36 are displayed like an oscilloscope.

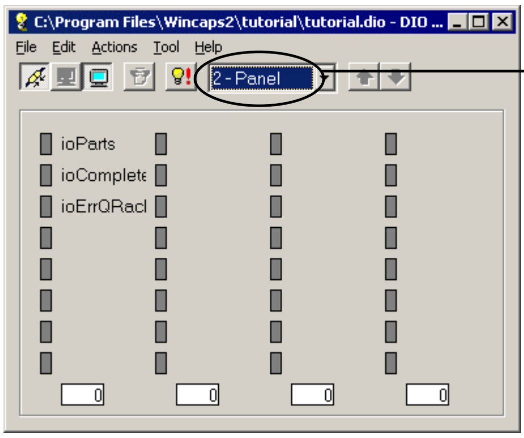
② The oscilloscope type display is suitable for examining the changing state of I/O signals with the lapse of time.

Check if each I/O is turned ON or OFF according to the program.

### Panel Type

With the DIO Manager, the I/O statuses can also be displayed as a panel type display.

Here you will display the I/O statuses of which the [Monitor] is "ON" in the table type format.


C:\Program Files\Wincaps2\tutorial\tutorial.dio - DIO ...
File Edit Actions Tool Help
2 - Panel
ioParts
ioComplete
ioErrQRacl
0
0
0
0

① Click on the button of the [Display Switch] list and select [2 – Panel] from the list displayed.

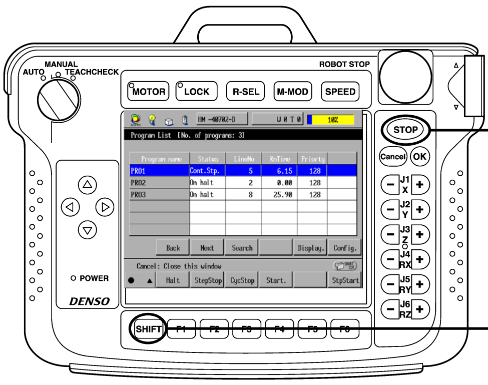
The statuses of I/O No.'s 34 to 36 are displayed like an oscilloscope.

② The panel type display is suitable for simultaneously monitoring the changes to the statuses of multiple I/O signals.

Check if each I/O is turned ON or OFF according to the program.

## 14.3 Stopping the running program

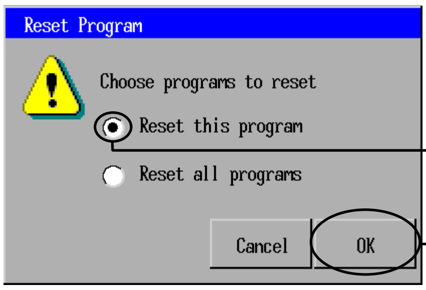
Up to this point, you have checked continuous run of "PRO1". Before proceeding onto Part 4, stop the running program.



① Press [STOP]

The program stops instantly and the [Status] box of the program changes from "Running" to "Cont.Stp."

② Press [SHIFT] and then [F7 ProgRst.].



Reset Program

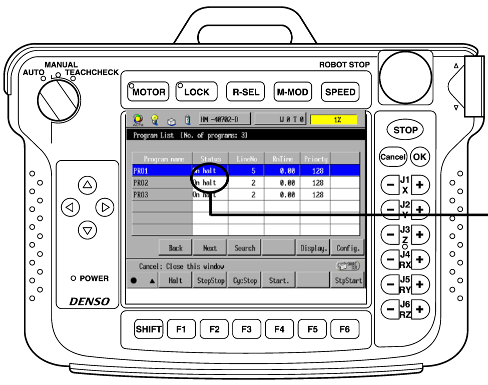
Choose programs to reset

[✓ Reset this program]

Reset all programs

Cancel OK

③ The Reset Program window appears. Choose "Reset this program" and press [OK].



The [Status] box shows "On halt" and the programmed operation stops.

# Part 4
# Running the Robot Using Programs

In Part 4, you will:

- Run the robot using a program.
- Learn palletizing, which is one of the main applications on 4-axis robots, and related programming.
- Learn how to effectively use PAC libraries, which are the easiest way to develop robot programs.

# Lesson 15 Running the Robot in Practice

You learnt how to check program operation in Parts 1 to 3. In Lesson 15, let's practice running the robot using programs.

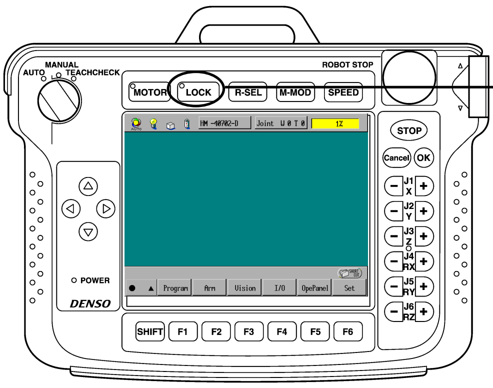
First of all, you should perform a safety check. Then set the robot at a low speed and run it slowly to check its motion.

You should also test-run the robot using dummy I/O signals.  In this lesson you will connect the actual hardware to the robot controller and check its motion.

Refer to the INSTALLATION & MAINTENANCE GUIDE for details on setting up your hardware.

## 15.1 Releasing machine lock

**Step 1**



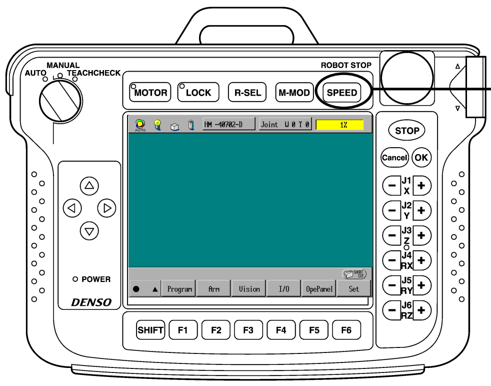
① Press [LOCK].

② Check that the green LED on the [LOCK] button comes on.

This releases machine lock.

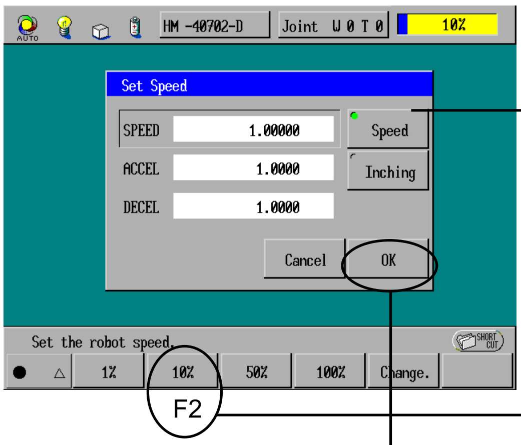
## 15.2 Setting robot speed and acceleration

### Step 1



① Press [SPEED].

### Step 2



SPEED 1.00000
ACCEL 1.0000
DECEL 1.0000

Set the robot speed.

The [Set Speed] window is displayed.

② Press [F2 10%].

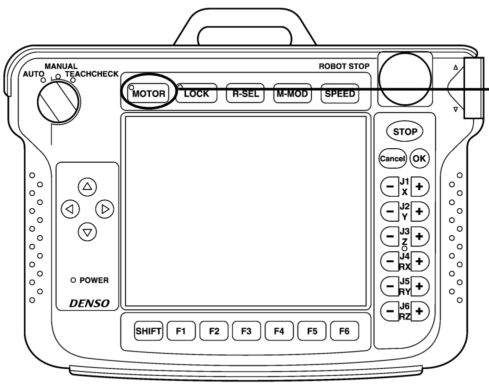
The set values appear in the [SPEED], [ACCEL], and [DECEL] fields.

③ Press [OK] to accept these settings.

This completes the procedure for setting the speed and acceleration of the robot.

## 15.3 Turning the drive motor ON

**Step 1**



① Press [MOTOR].

② Check that the LED on the [MOTOR] button comes on.

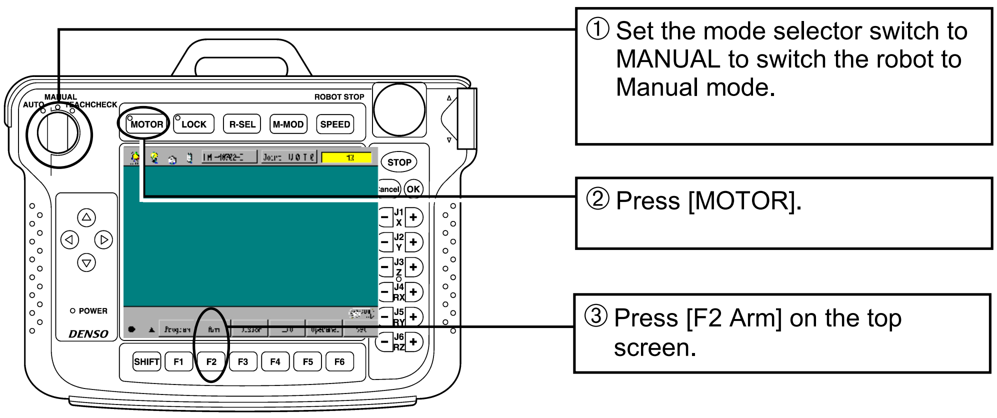
The power to the motor is now on.

## 15.4 Teaching

You entered values for the position variable values in Lesson 10 "Entering position variables", but positioning in the Arm Manager will not necessarily be exact. You will need to perform repositioning here by using the robot in practice, and assigning accurate values to the position variables.

You will follow almost same as that described in Lesson 10 "Inputting Values to Position Variables"; however, here you have to enter actual values observing the current robot position and figure, not to seek the position representing the target values.

### Step 1 Selecting Manual mode and displaying the current robot position

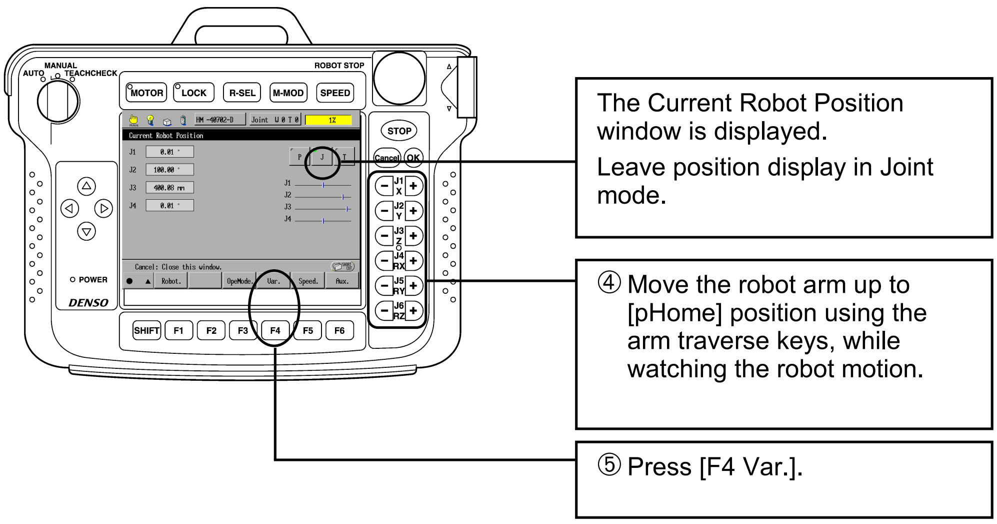


① Set the mode selector switch to MANUAL to switch the robot to Manual mode.

② Press [MOTOR].

③ Press [F2 Arm] on the top screen.

### Step 2 Moving the robot arm



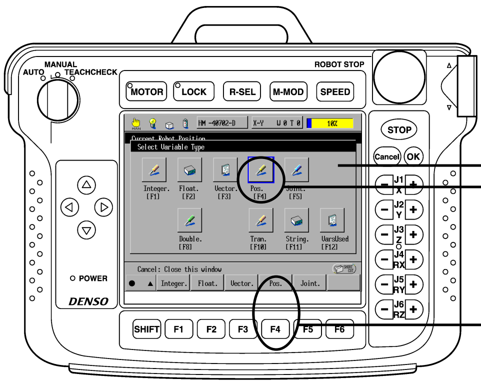
The Current Robot Position window is displayed.

Leave position display in Joint mode.

④ Move the robot arm up to [pHome] position using the arm traverse keys, while watching the robot motion.

⑤ Press [F4 Var.].

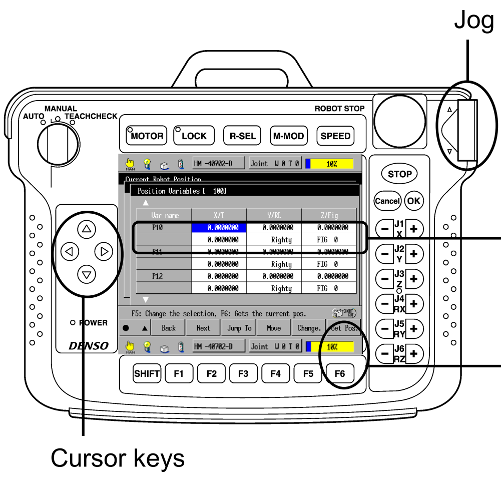
## Step 3 Displaying position variables



The [Select Variable Type] window is displayed.

⑥ Press [F4 Pos.].

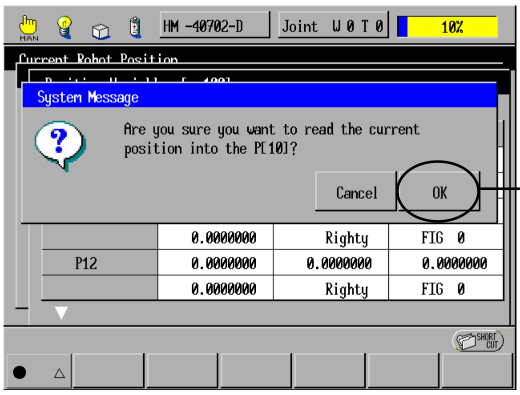
## Step 4 Assigning the current robot position to position variables



The Position Variables window appears.

⑦ Select the right column in [P10] by using the jog dial or cursor keys.

⑧ Press [F6 Get Pos.].



HM -40702-D Joint U 0 T 0 10%

Current Robot Position

System Message

Are you sure you want to read the current position into the P[10]?

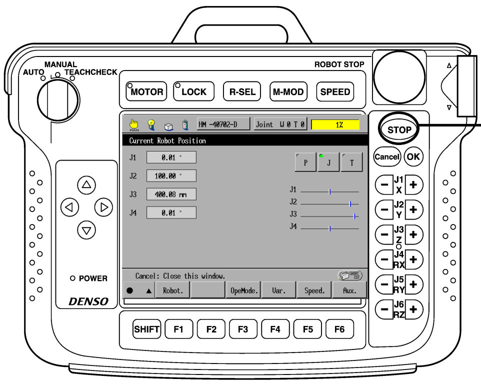
Cancel OK

|  |  |  |  |
|---|---|---|---|
|  | 0.0000000 | Righty | FIG 0 |
| P12 | 0.0000000 | 0.0000000 | 0.0000000 |
|  | 0.0000000 | Righty | FIG 0 |

⑨ Check the system message and press [OK].

The value of the current position will be assigned to position variable [P10].

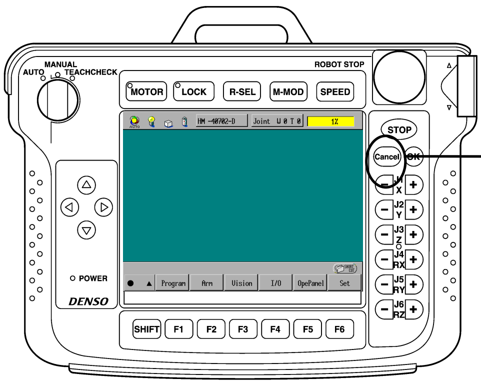
**Step 5** **Assigning other position values to position variables**



⑩ Press [Cancel] twice.
The screen returns to the [Current Robot Position]

⑪ Repeat Steps 2 to 4 to assign the position data to [P11], [P12] and [P13] into other three variables [pPlace1], [pPlace2] and [P13], respectively.

**Step 6** **Returning to the top screen**



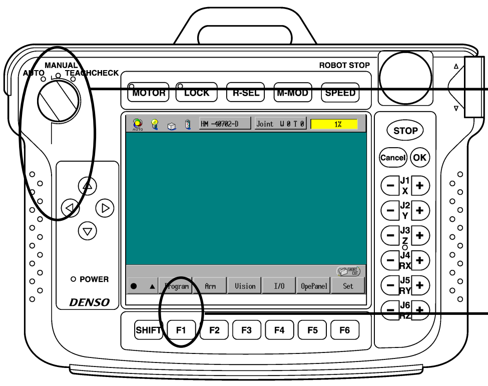
⑫ Press [Cancel] to return to the top screen after finishing all of the above procedures.

## 15.5 Starting programs

You will now actually run the robot using a program.

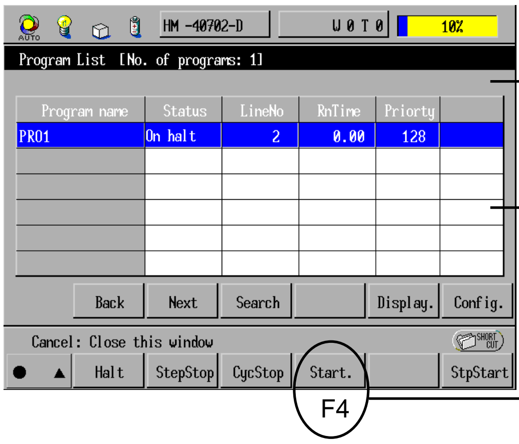
**Caution**: Always keep one hand free, ready to press the STOP key.

**Step 1** **Switching to Auto mode to display the program list window**



① Set the mode selector switch to the AUTO position to switch the robot to Auto mode.

② Press [F1 Program].


HM -40702-D U 0 T 0 10%

Program List [No. of programs: 1]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | On halt | 2 | 0.00 | 128 |

Back Next Search Display. Config.

Cancel: Close this window

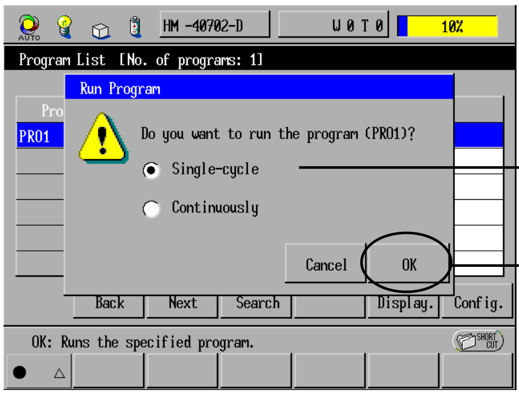
Halt StepStop CycStop Start. StpStart

The [Program List] window is displayed.

③ Select the [PRO1] line.

④ Press [F4 Start.].

**Step 2**


HM -40702-D U 0 T 0 10%

Program List [No. of programs: 1]

Run Program

Do you want to run the program (PRO1)?

[✓ Single-cycle] Continuously

Cancel OK

OK: Runs the specified program.

⑤ Select [Single-cycle] and press [OK].

The task program [PRO1] will start in single cycle.

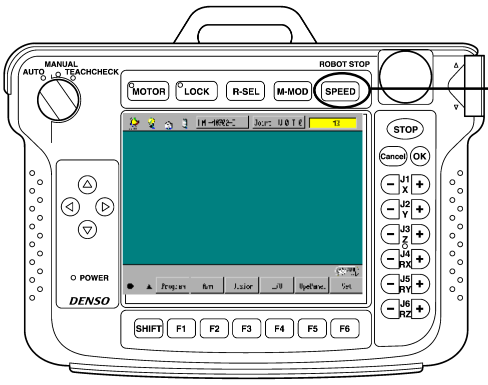
**Step 3** If you select [Continuously] instead of [Single-cycle], the robot will start in the continuous mode.

If operation in single cycle mode is successful, try operation in the continuous start mode.

## 15.6 Changing the robot speed

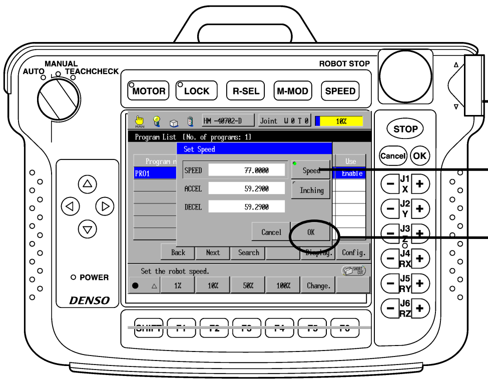
Let the robot run faster by gradually changing the speed settings if you have been successful in both the single cycle run and continuous run.

**Step 1**



① Press [SPEED].

**Step 2**



The [Set Speed] window appears.

② Turn the jog dial to select the desired speed.
When a value is set in the [SPEED] column, default values will be automatically entered in the [ACCEL] and [DECEL] columns.

③ Press [OK].

Robot speed, acceleration and deceleration are now set.

## 15.7 Stopping the robot in continuous running

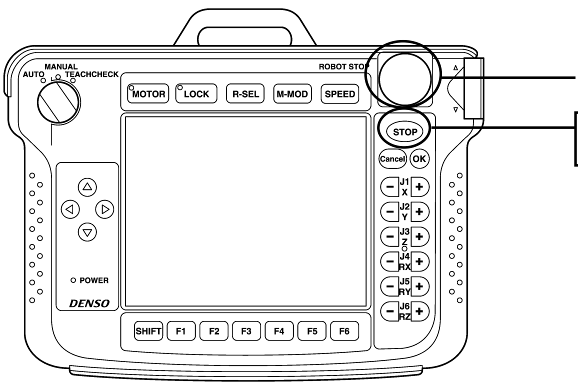
If you stop the robot during continuous running by pressing the [STOP] button, the robot will change to [Continue Stop] status. In this state you can restart the program from the line where the program stopped.

If you want to abort the program completely, you need to perform [Program Reset].

**Caution:** In emergencies, press the Robot Stop button.

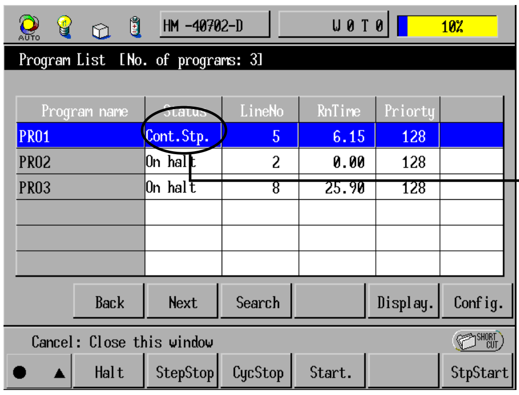
### ■ Making the program [Continue Stop] and [Continue Start]

**Step 1** **Making the task program [Continue Stop]**



Robot Stop button

① Press [STOP].

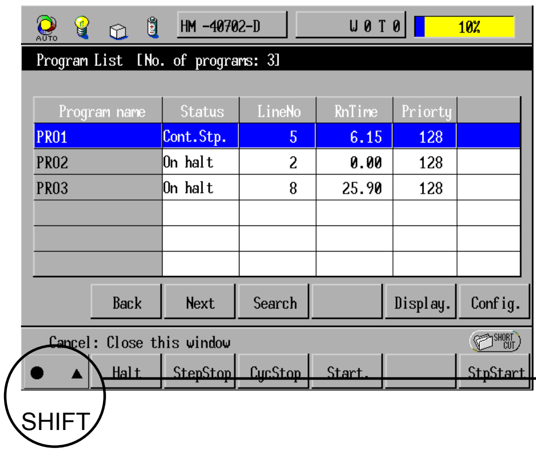

HM -40702-D　　U 0 T 0　　10%

Program List [No. of programs: 3]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Cont.Stp. | 5 | 6.15 | 128 |
| PRO2 | On halt | 2 | 0.00 | 128 |
| PRO3 | On halt | 8 | 25.90 | 128 |

The program stops immediately and [Cont. Stp.] will be displayed in the [Status] column.

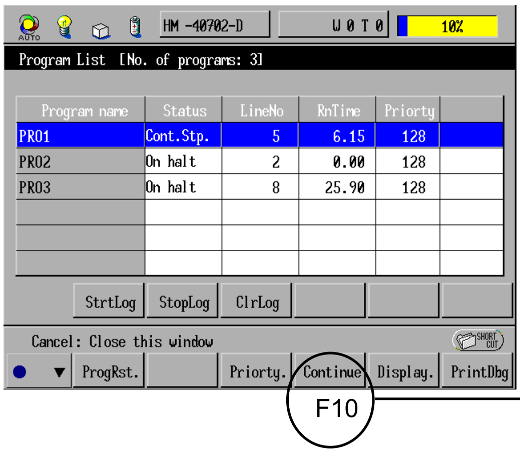
**Step 2** **Making the program [Continue Start]**


HM -40702-D　　U 0 T 0　　10%

Program List [No. of programs: 3]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Cont.Stp. | 5 | 6.15 | 128 |
| PRO2 | On halt | 2 | 0.00 | 128 |
| PRO3 | On halt | 8 | 25.90 | 128 |

② Press [Shift].

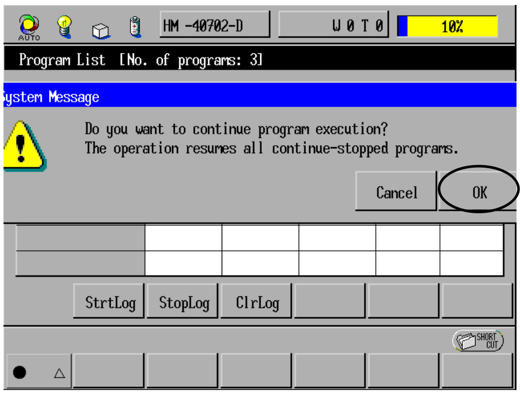


HM -40702-D  W 0 T 0  10%

Program List [No. of programs: 3]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Cont.Stp. | 5 | 6.15 | 128 |
| PRO2 | On halt | 2 | 0.00 | 128 |
| PRO3 | On halt | 8 | 25.90 | 128 |

③ Press [F10 Continue].



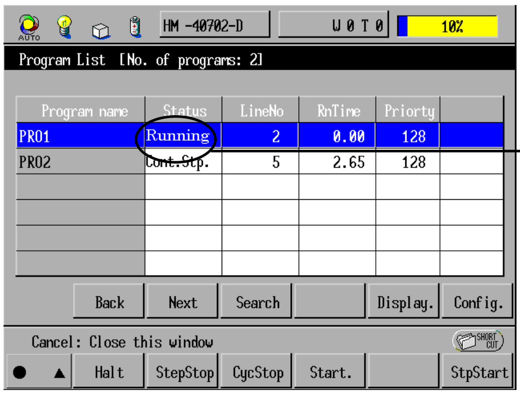
HM -40702-D  W 0 T 0  10%

Program List [No. of programs: 3]

ystem Message

Do you want to continue program execution?
The operation resumes all continue-stopped programs.

④ Press [OK].



HM -40702-D  W 0 T 0  10%

Program List [No. of programs: 2]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Running | 2 | 0.00 | 128 |
| PRO2 | Cont.Stp. | 5 | 2.65 | 128 |

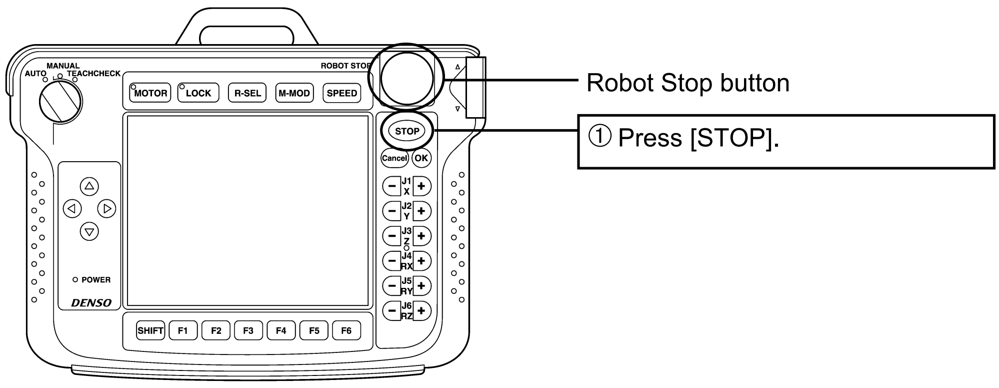
The program restarts and [Running] is displayed in the [Status] column.

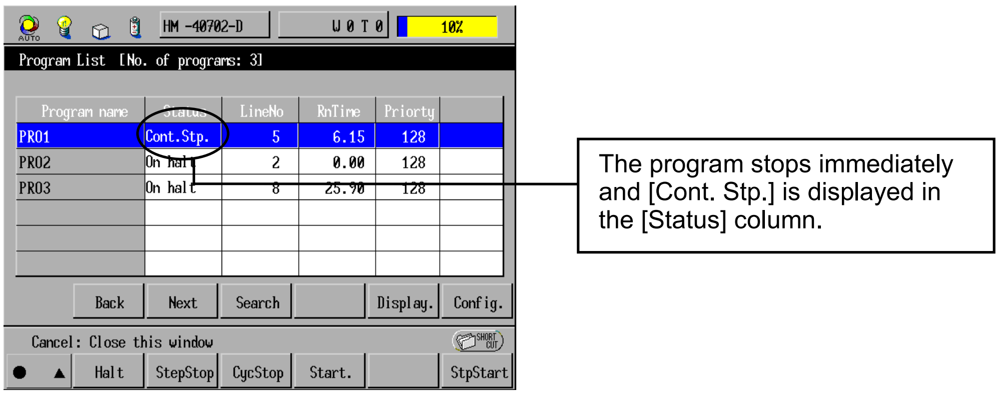
Now you are ready to restart the program.

## ■ Stopping the program completely ([Program Reset])

### Step 1

**Making the program [Continue Stop]**



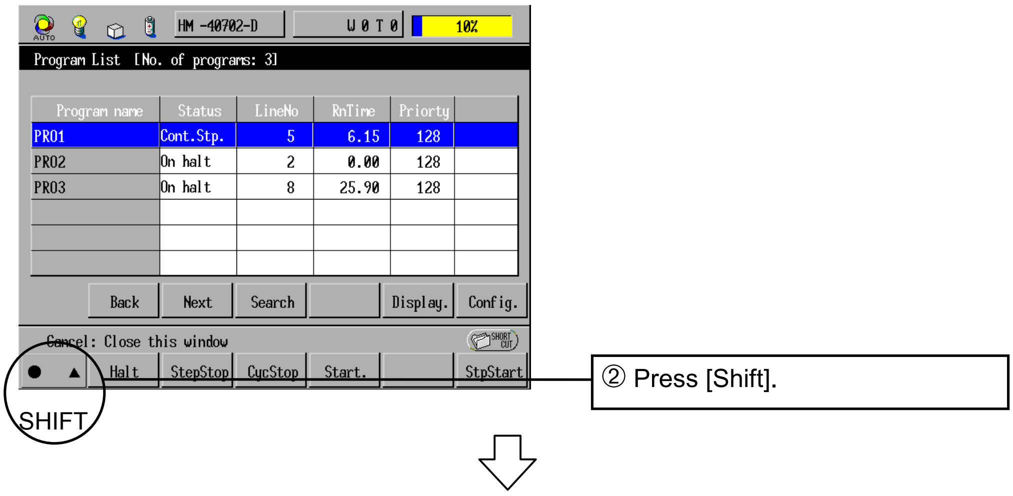

```
HM -40702-D        U 0 T 0   10%
Program List  [No. of programs: 3]
```

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Cont.Stp. | 5 | 6.15 | 128 |
| PRO2 | On halt | 2 | 0.00 | 128 |
| PRO3 | On halt | 8 | 25.90 | 128 |

The program stops immediately and [Cont. Stp.] is displayed in the [Status] column.

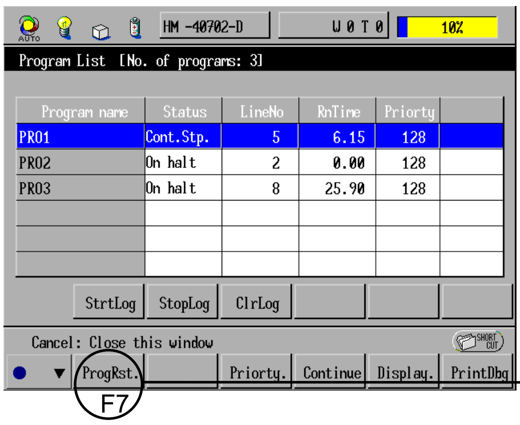
### Step 2

**Resetting the program ([Program Reset])**


```
HM -40702-D        U 0 T 0   10%
Program List  [No. of programs: 3]
```

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Cont.Stp. | 5 | 6.15 | 128 |
| PRO2 | On halt | 2 | 0.00 | 128 |
| PRO3 | On halt | 8 | 25.90 | 128 |

② Press [Shift].



HM -40702-D | U 0 T 0 | 10%

Program List [No. of programs: 3]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Cont.Stp. | 5 | 6.15 | 128 |
| PRO2 | On halt | 2 | 0.00 | 128 |
| PRO3 | On halt | 8 | 25.90 | 128 |

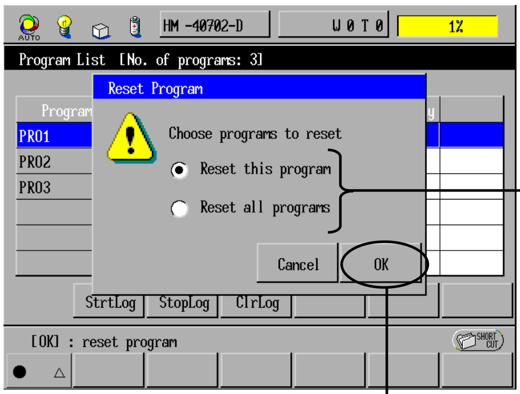
StrtLog | StopLog | ClrLog

Cancel: Close this window

ProgRst. | Priorty. | Continue | Display. | PrintDbg

F7

③ Press [F7 ProgRst.].



HM -40702-D | U 0 T 0 | 1%

Program List [No. of programs: 3]

Reset Program

Choose programs to reset

● Reset this program

○ Reset all programs

Cancel | OK

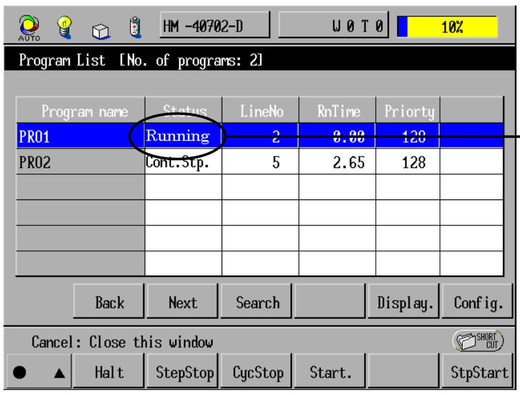
[OK] : reset program

④ Select the program you want to abort.

If you select [Reset this program], only the current program will be aborted.

If you select [Reset all programs], all programs will be aborted.

⑤ Press [OK].



HM -40702-D | U 0 T 0 | 10%

Program List [No. of programs: 2]

| Program name | Status | LineNo | RnTime | Priorty |
|---|---|---|---|---|
| PRO1 | Running | 2 | 0.00 | 128 |
| PRO2 | Cont.Stp. | 5 | 2.65 | 128 |

Back | Next | Search | Display. | Config.

Cancel: Close this window

Halt | StepStop | CycStop | Start. | StpStart

The program(s) is (are) aborted and [On halt] is displayed in the [Status] column.

The program(s) is (are) now aborted.

# Lesson 16 Palletizing

## 16.1 What is palletizing?

Palletizing refers to placing parts in/removing parts from a partitioned pallet (shown below) in programmed order.

You can easily use library programs for palletizing. To use these programs you have to only know the number of partitions provided in the pallet and the positions of each of the 4 corners of the pallet, and teach this information to the robot.

The palletizing programs update the partition information as each position is called to enable the robot to know which partition it should place the next part in/remove the next part from.

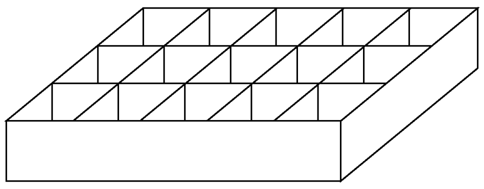


**Figure 16-1 Partitioned pallet**

## Palletizing Program "PRO1"

Using the library, you can find a typical procedure to build a program in program "PRO1" under the title "Palletizing template 2," although there would actually be many different possible programs for palletizing depending on the applications and the circumstances in which they are being used.

Use this "palletizing template 2" effectively as your template by adding/deleting the items necessary for your applications.

Listed below is a sample template program named "PRO1."

This sample template assumes that

- the palletizing points should be at P50 to P55, and
- the program to be built would control the robot to move to the position P50 which is the work pick-up position, to run the palletizing program "0", to move to the position P51 which is the work piece mount position, to unchuck the work piece, to check end of the pallet, to replace the pallet if end of work signal detected and if needed, and to end the program if no work piece remains.

When you create "New Project" by selecting "Palletizing" in "Device Type," the system manager will automatically register the "Palletizing template 2" with program name "PRO1" and the "Palletizing initialization template 1" with program name "PRO2."

```
' !TITLE   "Palletizing template 2"
' !AUTHOR   "DENSO CORPORTION"
#DEFINE pltIndex      0
#DEFINE ChuckNG  40

PROGRAM PRO1                           ' Rename PRO1 as desired.
  TAKEARM                               ' Obtains the arm semaphore.
  MOVE P, P50                          ' Move to the position P50.
                                       ' Move to the palletizing position P50.
  IF IO[ChuckNG] = ON THEN             ' Check status of previous picking-up.
    CALL pltDecCnt(pltIndex)           ' Subtract by 1 for the total counter.
  END IF
  CALL pltMove(pltIndex)               ' Execute palletizing 0.
  MOVE P, P51                          ' Move to the mount position P51.
                                       ' Move to the palletizing position P51.
  '<--- Insert unchucking or other operations here --->
  CALL pltGetPLT1END(pltIndex,0)       ' Acquires the end of 1st pallet signal on I[0].
  IF I[0] THEN                         ' If ON (that is when <>0),
    '<--- Insert pallet replacing or other operations here --->
    CALL pltResetPLT1END(pltIndex)     ' Clears the end of 1st pallet signal.
  END IF
  GIVEARM                              ' Releases the arm semaphore.
END
```

**Figure 16-2  Program [PRO1], "Palletizing template 2"**

## Palletizing parameters

Figure 16-3, 16-4, 16-5 and Table 16-1 show the parameters needed for palletizing.

PAC language retains these parameters as value sets of variables.

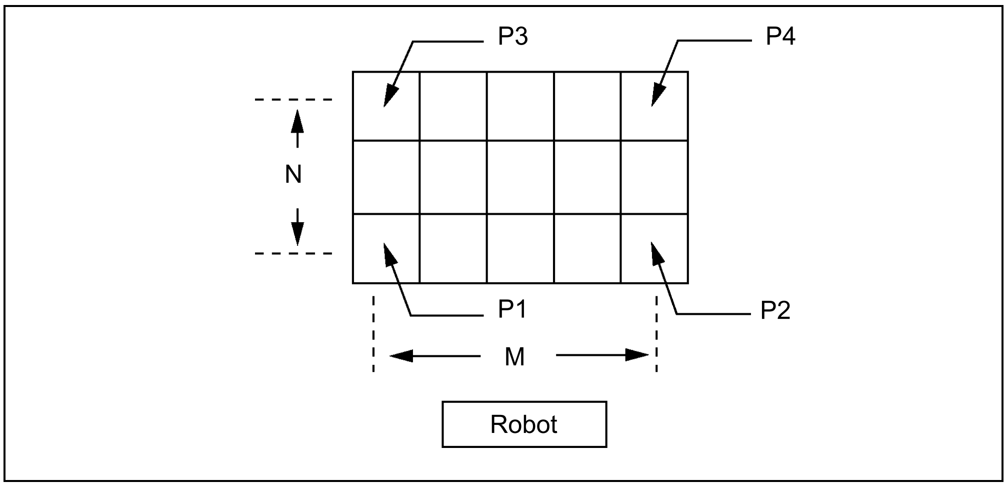


**Figure 16-3 Upper view of pallet**

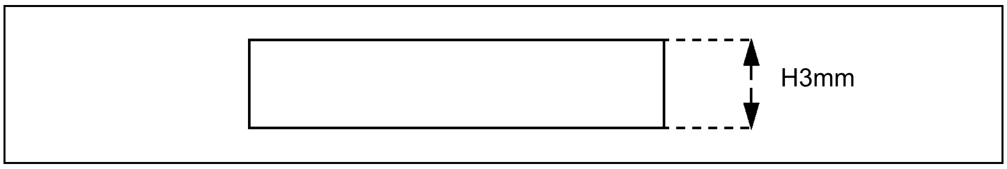


**Figure 16-4 Side view of pallet**

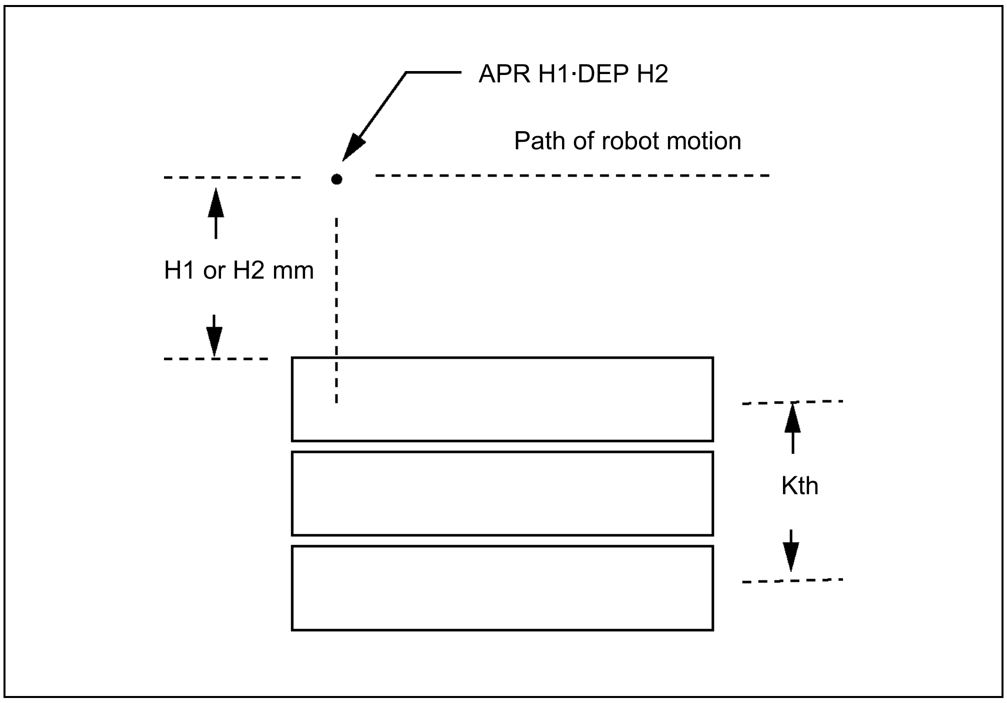


**Figure 16-5 Stacked pallets**

**Table 16-1  Parameters needed for palletizing**

| Symbol | Name | Description | Unit |
|---|---|---|---|
| | Palletizing number | Index of palletizing | None (Integer) |
| N | No. of row parts | Number of partitions from P1 to P3 | Count (Integer) |
| M | No. of column parts | Number of partitions from P1 to P2 | Count (Integer) |
| K | No. of stacked pallets | Number of stacked pallets | Count (Integer) |
| H1 | Approach clearance | Approach clearance where the robot approaches a pallet | mm (Single precision FPT) |
| H2 | Depart clearance | Departure clearance where the robot departs from a pallet | mm (Single precision FPT) |
| H3 | Height of a pallet | Height of a pallet | mm (Single precision FPT) |
| | Where H1 and H2 satisfy the conditions below.<br>H1 > {H3 x K-1)}+5<br>H2 > {H3 x K-1)}+5 | | |
| P1<br>P2<br>P3<br>P4 | Positions of the 4 corners of the pallet as shown in Figure 16-3.<br>It is not possible to exchange the relative positioning of any of the corners.<br>The robot maintains its orientation from where the position P1 was taught previously, for all points in the program. | | |

N  Number of partitions in row
Expresses the number of partitions in each row of the pallet.
If this is 3, it reflects 3 rows as in the example in Figure 16-3.

M  Number of partitions in column
This expresses the number of partitions in each column of the pallet.
If this is 5, it reflects 5 rows as in the example in Figure 16-4.

K  Number of stacked pallets
This expresses the number of pallets in the pallet stack.
If this is 3, it reflects 3 stacked pallets as in the example on Figure 16-5.

H1  Approach clearance
Expresses the length of the approach path as the robot approaches the pallets.
A program applies the single approach path length at every call of the same palletizing program.

H2  Departure path clearance
Expresses the length of the departure path as the robot departs from the pallets.
A program applies the single departure path length at every call of the same palletizing program.

H3  Pallet unit heights
Expresses height of each pallet.
For every pallet added to a stack, a plus unit value is added.
For every pallet removed from a stack, a minus unit value is added.
If the stack is not changed, 0 is added.

**Caution:** H1 and H2 shall satisfy the conditions below.

H1 > {H3 x (K-1)} + 5
H2 > {H3 x (K-1)} + 5

If not, an error will occur during initializing. These restrictions ensure the robot does not crush the pallet in operation by ensuring the robot approaches or departs from the stacked pallets at 5 mm higher than the topmost pallet in a stack.

As shown in Figure 16-6, changing stack height does not affect the approach/departure points of the robot in same palletizing program.

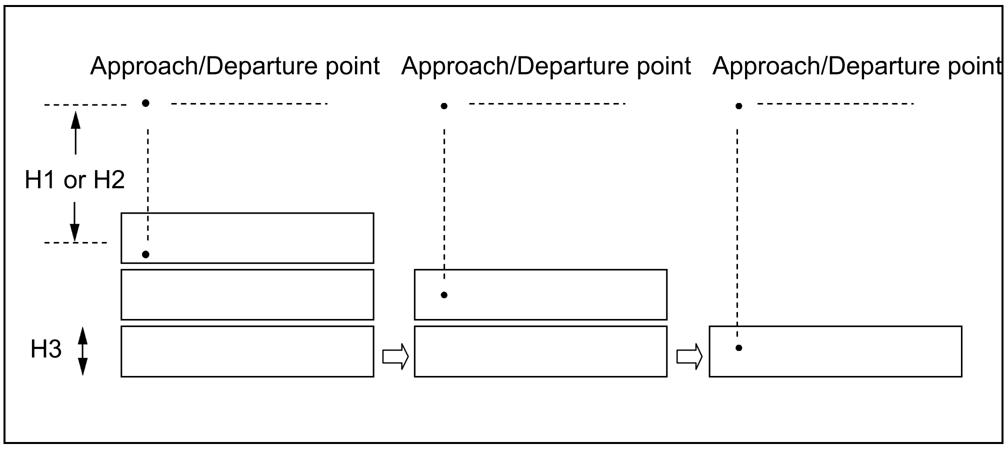


**Figure 16-6 Relationship between the stack height and approach/departure points**

### Four corner points P1, P2, P3 and P4

These points represent the parts position for each of the 4 corner partitions of the pallet. Figure 16-7 depicts in what order the robot palletizes these parts.

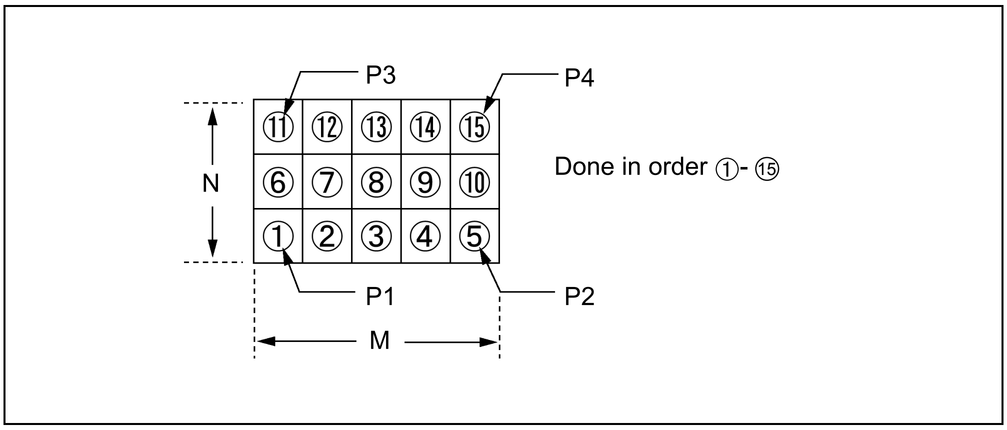


**Figure 16-7 Palletizing order**

### Setting palletizing parameters

To set the parameter values such as the number of row and column partitions and the number of pallets in a stack, first call the “pltInitialize” module from the program library.

If you select [1-palletizing] in [Device type] in the [New project] dialog box when you want to create "New system project," the System Manager will automatically register the program “PRO2” titled “Palletizing initialization template 1” in the library. Since this program is to call “pltInitialize” module, modify the values of the indexes in the CALL statement to the ones you want to use.

```
'!TITLE  “Palletizing initialization template 1”
' !AUTHOR   "DENSO CORPORTION"

PROGRAM PRO2
  CALL pltInitialize(0,4,3,1,50,50,50,52,53,54,55)          'Initializing palletizing No. 0.
END
```

**Figure 16-8 Library program "Palletizing initialization template 1"**

```
Descriptions on parameters of CALL pltInitialize (0, 3, 5, 3, 50, 50, 10, 52, 53, 54, 55)
  1st  . . .  Palletizing program number (0)
  2nd  . . .  Number of row partitions (3)
  3rd  . . .  Number of column partitions (5)
  4th  . . .  Number of stacked pallets (3)
  5th  . . .  Approach clearance (50mm)
  6th  . . .  Departure clearance (50mm)
  7th  . . .  Pallet height (10mm)
  8th  . . .  P1 position (P52) ┐
  9th  . . .  P2 position (P53) │ Designate only the indexes of the position
 10th  . . .  P3 position (P54) │ variables.
 11th  . . .  P4 position (P55) ┘
```

**Figure 16-9 Parameters of "pltInitialize"**

You can also find descriptions on the above parameters in the "Command builder" tool provided in the PAC Manager of WINCAPSII.

## Palletizing Counter

In palletizing, the robot counts the number of partitions as they change and retains the counts in the variables.

There are four types of counters; number of partitions in the row (N), number of partitions in the column (M), number of stacked pallets (K) and total (cnt).

These counters are defined in “pltKernl” which is the kernel program for controlling palletizing operation.

The library program “pltMove” adds 1 to the total counter every time a palletizing operation is completed and aligns the values of the other counters.

The library program “pltDecCnt” subtracts 1 from the total counter every time it is called and aligns each counter.

You can create up to 30 palletizing programs as the initial setting. Therefore, the system may provide 31 sets of the palletizing counters.

## Count Rules

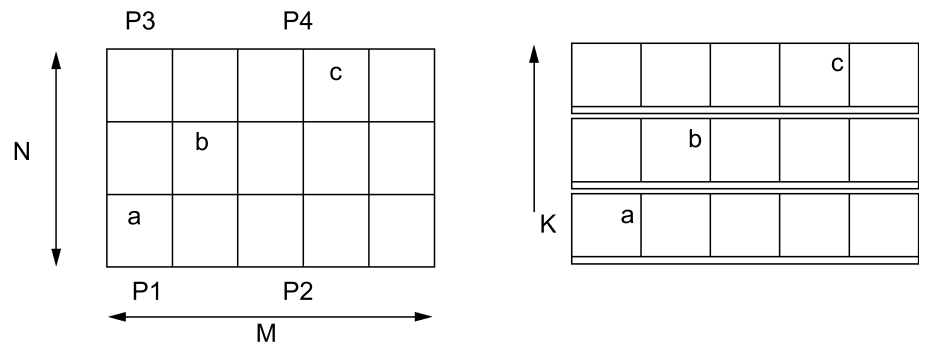
The palletizing counter adds 1 to the total counter every time “pltMove” is run and aligns the counts of the other counters so as to ensure the next pallet position.

If adding 1 to the total counter, the position of the pallet column indicated by the column counter (M) moves to the next column. If the pallet column position indicated by the column counter (M) reaches the end and becomes the maximum count, then the row counter (N) counts up by 1 to indicate the next row of the pallet and the column counter (M) becomes its minimum value. If the position of the row counter (N) reaches the end of the pallet row partition and becomes the maximum value, the stacked pallet counter (K) counts up by 1 and the row counter (N) becomes the set minimum value.

If you halt a palletizing program during operation and restart it, the robot moves to the next partition because the value of the counter variable is added to.

The system retains the contents of the palletizing counter even if the power is turned off. Unless you initialize the system after restarting, the robot will proceed to palletize from the previous counter value.

**Caution:** When you compile a new task program and load its run time module the system will automatically initialize the values of all variables.



If counts are N=3, M=5 and K=3,
position a is at (N=1, M=1 and K=1)
position b is at (N=2, M=2 and K=2)
position c is at (N=3, M=4 and K=3)

**Figure 16-10 Relationship between palletizing position and counters**

## Initializing the Counters

When you replace any pallets or do not want to use any partitions, you need to initialize all of the counters.

The systems substitute 1 into all of the counters to initialize them.

If you use the library program "pltResetAll" you can initialize all the palletizing counters at once.

For example, if you want to initialize all counters for pallet number 1, write as follows.

```
CALL pltResetAll(1)
```

If you want to initialize each palletizing counter independently, use the library programs “pltLetN1,” “pltLetM1,” “pltLetK1” and “pltLetCnt.”

For example, if want to initialize the N counter for pallet number 1, write as follows.

```
CALL pltLetN1(1,1)
```

**Caution:** The second argument is the value to substitute into the N counter. You can also choose any number instead of 1.

## Ending palletizing program process

Upon finishing the palletizing for one of the stacked pallets or for the whole pallet stack, the palletizing program sets a stacked pallet end flag or whole pallet stack end flag, respectively.

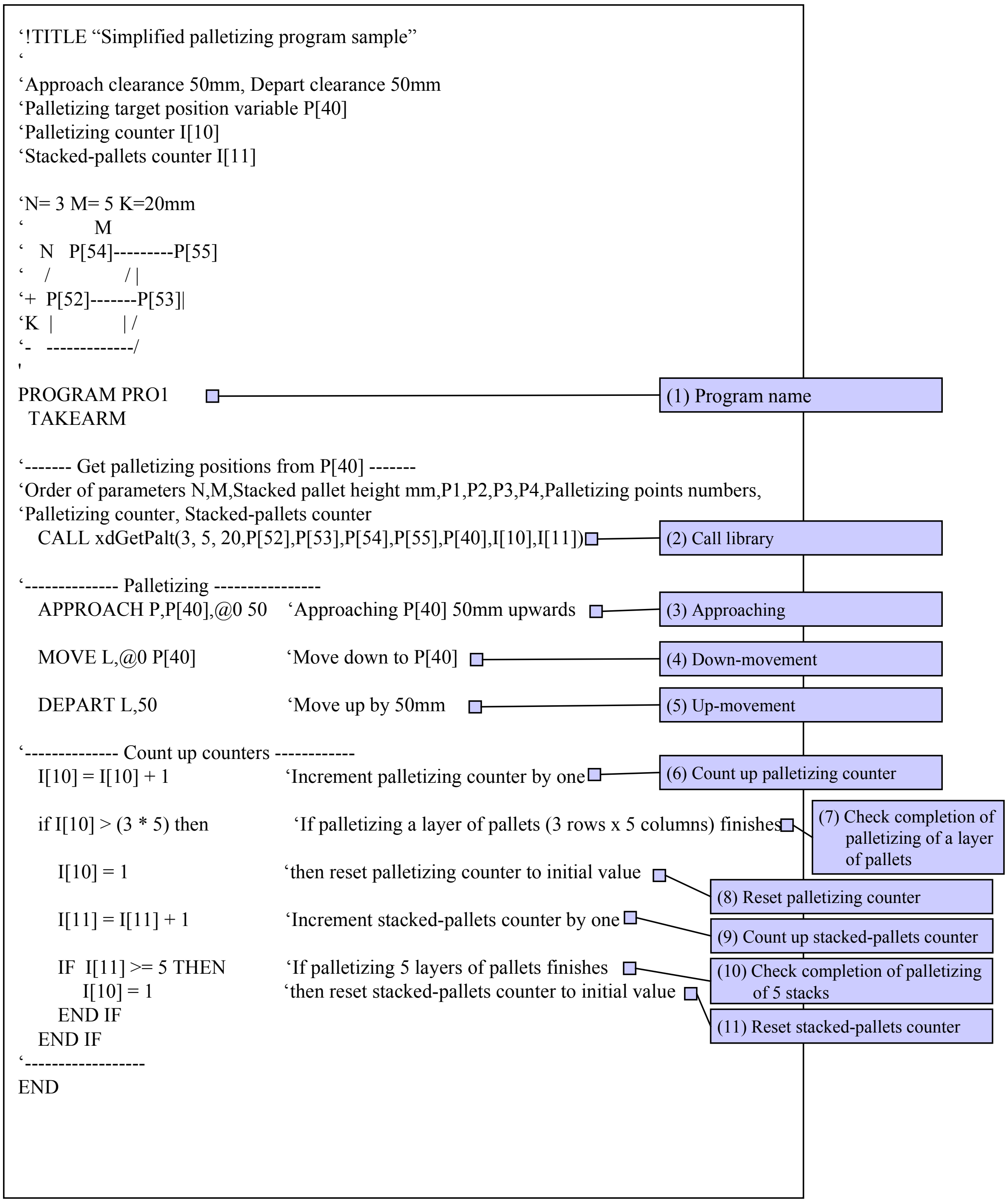
To obtain the one stacked pallet finish flag status, use the library program “pltGetPLT1END.” To obtain the whole pallet stack finish flag status, use the library program “pltGetPLTEND.”

To reset the one stacked pallet finish flag to (0), use the library program “pltResetPLT1END.” To reset the whole pallet stack finish flag to (0), use the library program “pltResetPLTEND.”

## 16.2 Simplified Palletizing

Palletizing is explained in the “16.1 What is palletizing?”. For simpler palletizing, this section provides you with a simplified palletizing template using a palletizing library.

**Simplified Palletizing Program "PRO1"**



```
‘!TITLE “Simplified palletizing program sample”
‘
‘Approach clearance 50mm, Depart clearance 50mm
‘Palletizing target position variable P[40]
‘Palletizing counter I[10]
‘Stacked-pallets counter I[11]

‘N= 3 M= 5 K=20mm
‘          M
‘  N  P[54]---------P[55]
‘   /            / |
‘+ P[52]-------P[53]|
‘K |           | /
‘-  -------------/
'
PROGRAM PRO1
  TAKEARM

‘------- Get palletizing positions from P[40] -------
‘Order of parameters N,M,Stacked pallet height mm,P1,P2,P3,P4,Palletizing points numbers,
‘Palletizing counter, Stacked-pallets counter
   CALL xdGetPalt(3, 5, 20,P[52],P[53],P[54],P[55],P[40],I[10],I[11])

‘-------------- Palletizing ----------------
   APPROACH P,P[40],@0 50    ‘Approaching P[40] 50mm upwards

   MOVE L,@0 P[40]           ‘Move down to P[40]

   DEPART L,50               ‘Move up by 50mm

‘-------------- Count up counters ------------
   I[10] = I[10] + 1         ‘Increment palletizing counter by one

   if I[10] > (3 * 5) then    ‘If palletizing a layer of pallets (3 rows x 5 columns) finishes

      I[10] = 1              ‘then reset palletizing counter to initial value

      I[11] = I[11] + 1      ‘Increment stacked-pallets counter by one

      IF  I[11] >= 5 THEN    ‘If palletizing 5 layers of pallets finishes
          I[10] = 1          ‘then reset stacked-pallets counter to initial value
      END IF
   END IF
‘------------------
END
```

(1) Program name
(2) Call library
(3) Approaching
(4) Down-movement
(5) Up-movement
(6) Count up palletizing counter
(7) Check completion of palletizing of a layer of pallets
(8) Reset palletizing counter
(9) Count up stacked-pallets counter
(10) Check completion of palletizing of 5 stacks
(11) Reset stacked-pallets counter

## ■ Simplified palletizing program "PRO1"

In palletizing explained in the “16.1 What is palletizing?”, you need to execute the pltInitialize library before starting palletizing.

This simplified palletizing program requires no execution of that library. Just executing PRO1 will start palletizing operation.

In simplified palletizing, you need to specify addition and resetting of the palletizing counter and stacked-pallets counter, while in conventional palletizing those counters are automatically controlled inside libraries.

Variables used in PRO1

- Palletizing target position variable (Position variable, P40 in this example)
- Palletizing counter variable (Integer variable, I10 in this example)
- Stacked-pallets counter (Integer variable, I11 in this example)
- Corner partition variables (Position variables, P52 to P55 in this example)

What to do before execution of PRO1

Before start of PRO1, you need to:

- Assign the initial value "1" to each of the palletizing counter I10 and stacked-pallets counter I11 and
- Teach the positions of four corner partitions in the pallet to corner partition variables P1 to P4.

On the following pages are detailed explanation of each part of the program PO1.

(1) Program name

```
‘
PROGRAM PRO1          Change the program name
  TAKEARM
```

(2) Call library

```
‘------- Get palletizing positions from P[40] --------
‘Order of parameters N,M,Stacked pallet height mm,P1,P2,P3,P4,Palletizing points numbers
‘Palletizing counter, Stacked-pallets counter
   CALL xdGetPalt (3, 5, 20,P[52],P[53],P[54],P[55],P[40],I[10],I[11])
```

Setting the following parameters to the called library will assign the target position to the palletizing target position variable specified by the 8th parameter.

| | |
|---|---|
| 1st parameter | No. of rows, which should be 1 or greater.<br>(3 rows in this example) |
| 2nd parameter | No. of columns, which should be 1 or greater.<br>(5 columns in this example) |
| 3rd parameter | Height of stacked pallets in mm.<br>Specify a positive value when increasing the layers of pallets; a negative value when decreasing them.<br>(20 mm specified in this example) |
| 4th to 7th parameters | Position variables to which four corner partition positions of the pallet are assigned.<br>(P52 to P55 in this example) |
| 8th parameter | Palletizing target position variable to which the target position will be assigned. This position may be calculated from the current counter values.<br>(P40 in this example) |
| 9th parameter | Palletizing counter, which should be 1 or greater and M*N or less. According to this value, the corner partition positions may be specified. |
| 10th parameter | Stacked-pallets counter, which should be 1 or greater.<br>According to this value, the layer number may be specified. |

(3) Approaching

(4) Down-movement

(5) Up-movement

```
‘------- Palletizing -------------------------
   APPROACH P,P[40],@0 50      ‘Approaching P[40] 50mm upwards
   MOVE L,@0 P[40]             ‘Move down to P[40]
   DEPART L,50                 ‘Move up by 50mm
```

As a result of execution of "(2) Call library," the palletizing target position is assigned to P40. Then some operations should be carried out to P40.

Usually, during those operation, chuck and unchuck processes will be inserted.

(6) Count up palletizing counter

(7) Check completion of palletizing of a layer of pallets

(8) Reset palletizing counter

(9) Count up stacked-pallets counter

(10) Check completion of palletizing of 5 layers of pallets

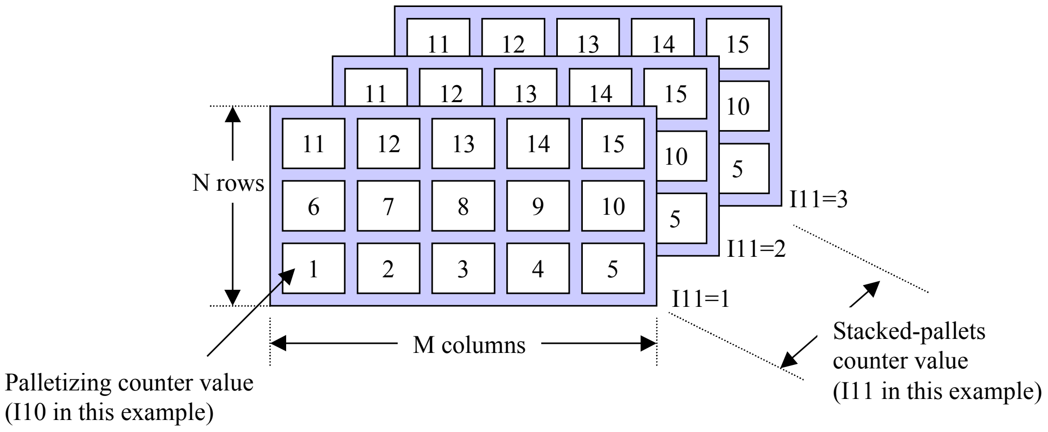
(11) Reset stacked-pallets counter

```
‘------- Count up counters-----------------
   I[10] = I[10] + 1              ‘Increment palletizing counter by one

   if I[10] > (3 * 5) then        ‘If palletizing a layer of pallets (3 rows x 5 columns) finishes
      I[10] = 1                   ‘then reset palletizing counter to initial value
      I[11] = I[11] + 1           ‘Increment stacked-pallets counter by one
      IF  I[11] >= 5 THEN         ‘If palletizing 5 layers of pallets finishes
         I[10] = 1                ‘then reset stacked-pallets counter to initial value
      END IF
   END IF
```

This part of the PRO1 counts up the palletizing counter and stacked-pallets counter and checks the completion of palletizing operation for a layer of pallets.

Unlike usual palletizing programs, the simplified palletizing program uses integer variables (I10 and I11 in this example) as a palletizing counter and stacked-pallets counter.

According to the values assigned to I10 and I11, the "(2) Call library" calculates the palletizing target position and assigns it to P40.

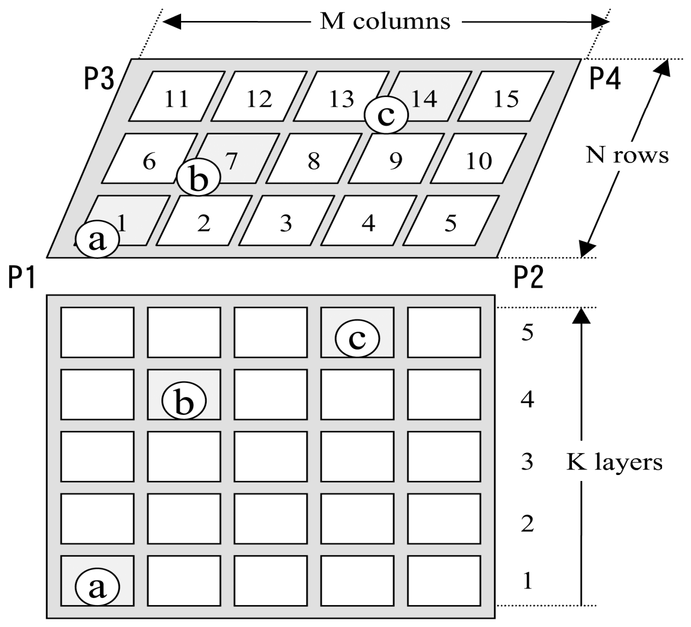


For a single layer of pallet, you may simplify the program further as shown below.

```
‘------- Count up counters-----------------
  I[10] = I[10] + 1              ‘Increment palletizing counter by one

  if I[10] > (3 * 5) then        ‘If palletizing a layer of pallets (3 rows x 5 columns) finishes
     I[10] = 1                   ‘then reset palletizing counter to initial value
     I[11] = I[11] + 1           ‘Increment stacked-pallets counter by one
     IF  I[11] >= 5 THEN        ‘If palletizing 5 layers of pallets finishes
        I[10] = 1                ‘then reset stacked-pallets counter to initial value
     END IF
  END IF
```

Delete these lines for a single layer of pallet.

## ■ Relationship between the palletizing positions and counter values in the simplified palletizing program



If each pallet consists of 3 rows x 5 columns (N=3, M=5), palletizing counter is I10 and stacked-pallets counter is I11, then

Position ⓐ: I10=1, I11=1

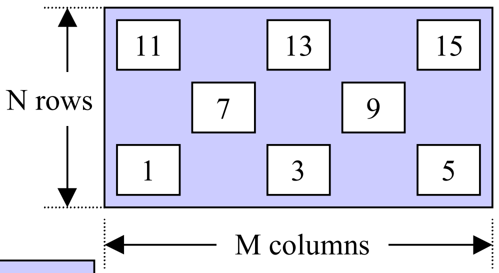
Position ⓑ: I10=7, I11=4

Position ⓒ: I10=14, I11=5

## ■ Applications of the simplified palletizing program --- Special-purpose palletizing examples ---

### (1) Alternate checker-pattern palletizing

Alternate checker-pattern palletizing refers to palletizing to every other partitions as illustrated below. Programming for this is very easy.

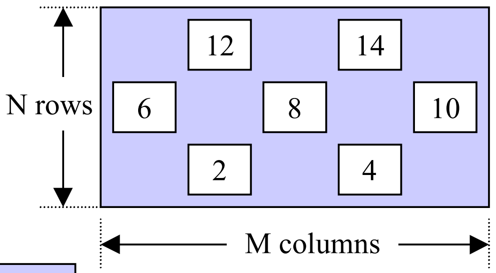


(6) Count up palletizing counter

```
'----------- Count up counters -----------------
  I[10] = I[10] + 1        'Increment palletizing counter by one
```

```
'--------- Count up counters -----------------
  I[10] = I[10] + 2        'Increment palletizing counter by 2
```

You need to assign "1" to palletizing counter I[10] with the teach pendant beforehand.



(6) Count up palletizing counter

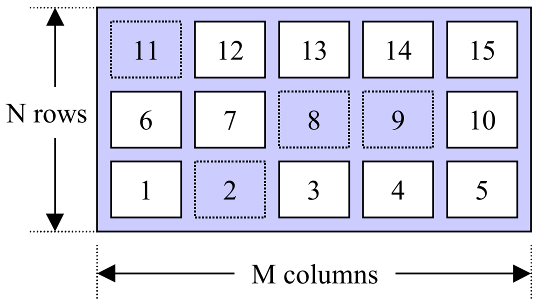
```
'------ Count up counters -------------
    I[10] = I[10] + 1        'Increment palletizing counter by one
  if I[10] > (3 * 5) then     'If palletizing a layer of pallets (3 rows x 5 columns) finishes
    I[10] = 1                 'then reset palletizing counter to initial value
```

```
'------Count up counters -----------
    I[10] = I[10] + 2        'Increment palletizing counter by 2
  if I[10] > (3 * 5) then     'If palletizing a layer of pallets (3 rows x 5 columns) finishes
    I[10] = 2                 'then reset palletizing counter to second value
```

You need to assign "2" to palletizing counter I[10] with the teach pendant beforehand.

### (2) Skipped palletizing

Skipped palletizing skips arbitrary partitions in palletizing.



No palletizing to 2nd, 8th, 9th or 11th partition.

The above palletizing operation seems complicated, but you may easily program such palletizing just by changing the palletizing counter value that will pass to the library.

```
PROGRAM PRO1
  TAKEARM

'------- Get palletizing positions from P[40] ------
'Order of parameters N,M,Stacked pallet height mm,P1,P2,P3,P4,Palletizing points numbers
'Palletizing counter, Stacked-pallets counter
    CALL xdGetPalt(3, 5, 20,P[52],P[53],P[54],P[55],P[40],I[10],I[11])
```

```
PROGRAM PRO1
  TAKEARM
  SELECT CASE I[10]
    CASE 2          'If palletizing counter I[10]=2
      I[10] = 3     'then set the counter to 3

    CASE 8,9        'If palletizing counter I[10]=8 or 9
      I[10] = 10    'then set the counter to 10

    CASE 11         'If palletizing counter I[10]=11
      I[10] = 12    'then set the counter to
  END SELECT

'------- Get palletizing positions from P[40] ------
'Order of parameters N,M,Stacked pallet height mm,P1,P2,P3,P4,Palletizing points numbers
'Palletizing counter, Stacked-pallets counter
    CALL xdGetPalt(3, 5, 20,P[52],P[53],P[54],P[55],P[40],I[10],I[11])
```

# Lesson 17 Using Libraries Effectively

You can assemble existing programs into a project in any of the following three ways.

- Program bank:

  You can add any programs registered in the program bank to your new project.

- Importing programs:

  You can register any programs into your new project folder by copying programs already created for another program project.

- Adding programs:

  You can register any programs already created for other program projects into your new project. In this case, the projects it is registered with share the program.

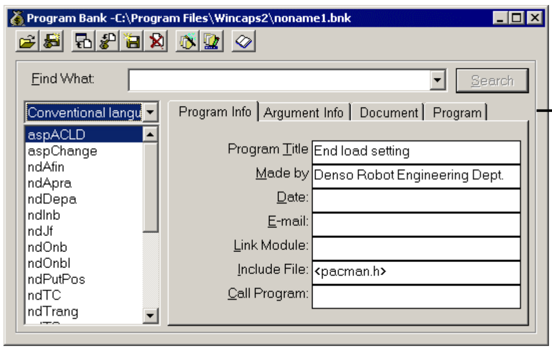
You can effectively apply existing programs to your new project using any of the methods above. Some examples to do so follow.

## 17.1 Program Bank

Let's add the program "dioSetAndWait" from the program bank to your new project.

**Step 1**

**Searching the program bank**



Program Bank -C:\Program Files\Wincaps2\noname1.bnk

Conventional langu

aspACLD
aspChange
ndAfin
ndApra
ndDepa
ndInb
ndJf
ndOnb
ndOnbl
ndPutPos
ndTC
ndTrang

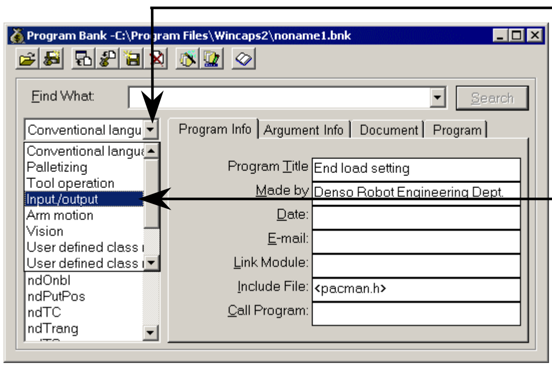
Program Title: End load setting
Made by: Denso Robot Engineering Dept.
Date:
E-mail:
Link Module:
Include File: <pacman.h>
Call Program:

① Select [Program Bank] from the Tool menu in the [PAC Manager window].

The [Program Bank] window is displayed.

**Step 2**

**Choosing the class**



Program Bank -C:\Program Files\Wincaps2\noname1.bnk

Conventional langu

Conventional langu
Palletizing
Tool operation
Input /output
Arm motion
Vision
User defined class
User defined class
ndOnbl
ndPutPos
ndTC
ndTrang

Program Title: End load setting
Made by: Denso Robot Engineering Dept.
Date:
E-mail:
Link Module:
Include File: <pacman.h>
Call Program:

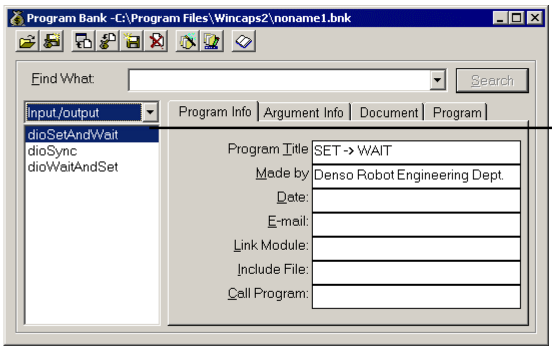
② Click the class selection box.

The class selection pull-down menu is displayed.

③ Select [Input/output].

**Step 3**

**Choosing a program to be added**



Program Bank -C:\Program Files\Wincaps2\noname1.bnk

Input/output

dioSetAndWait
dioSync
dioWaitAndSet

Program Title: SET -> WAIT
Made by: Denso Robot Engineering Dept.
Date:
E-mail:
Link Module:
Include File:
Call Program:

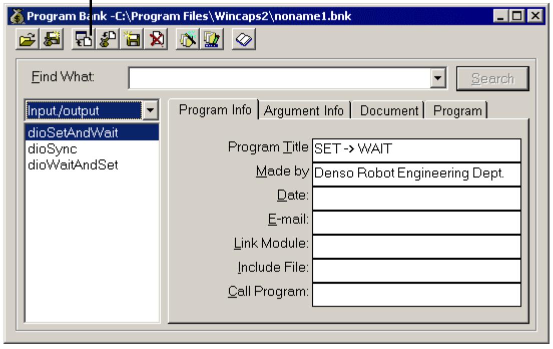
The program name list box shows the titles of programs registered in the [Input/output Class].

④ Select [dioSetAndWait] displayed in the list.

**Step 4**

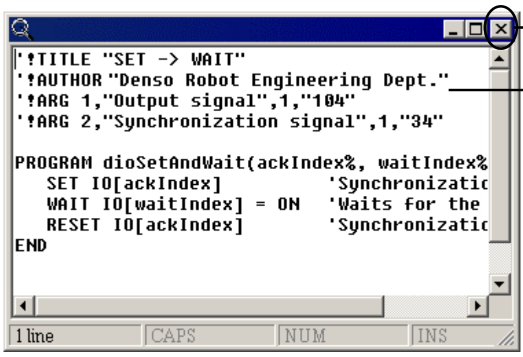
## Adding the program to a new project

⑤ Click [Add to Project].



dioSetAndWait
dioSync
dioWaitAndSet

Program Title: SET -> WAIT
Made by: Denso Robot Engineering Dept.
Date:
E-mail:
Link Module:
Include File:
Call Program:



```
'!TITLE "SET -> WAIT"
'!AUTHOR "Denso Robot Engineering Dept."
'!ARG 1,"Output signal",1,"104"
'!ARG 2,"Synchronization signal",1,"34"

PROGRAM dioSetAndWait(ackIndex%, waitIndex%
   SET IO[ackIndex]           'Synchronizatio
   WAIT IO[waitIndex] = ON    'Waits for the
   RESET IO[ackIndex]         'Synchronizatio
END
```

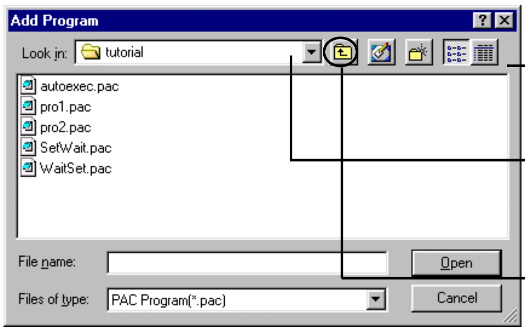
[dioSetAndWait] is registered in the new project and the [Edit] window is displayed.

⑥ Click here to close the [Program Bank] window.
The window will close.

## 17.2 Importing programs

Let's import two programs, "pro1" and "pro2" into your new project. If you have installed WINCAPSII in "Typical," these programs will be in the directory C:\ProgramFiles\Wincaps2\Sample\modelcase\.

### Step 1


Add Program

Look in: tutorial

autoexec.pac
pro1.pac
pro2.pac
SetWait.pac
WaitSet.pac

File name:

Files of type: PAC Program(*.pac)

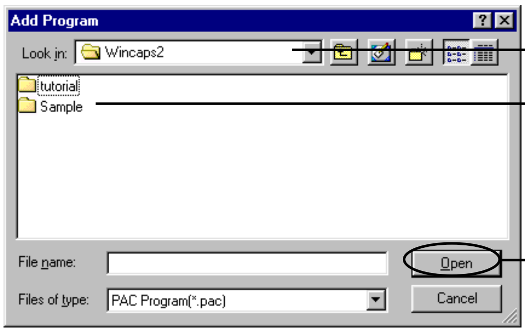
① Select [Import] from the Program menu, in the [PAC Manager] window.

The [Add Program] dialog window is displayed.

The [Tutorial] folder of the current system folder is displayed in the window.

② Click the [Previous] button .

### Step 2


Add Program

Look in: Wincaps2

tutorial
Sample

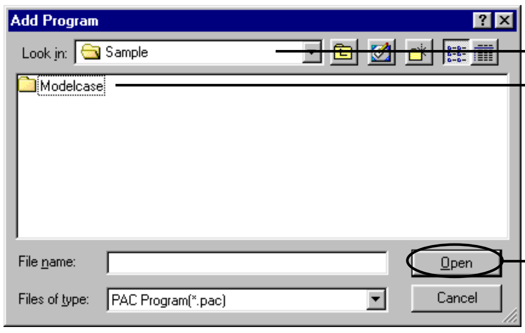
File name:

Files of type: PAC Program(*.pac)

The [Wincaps2] folder is displayed in the [Look in] window, showing the directory holding programs you are seeking

③ Select the [Sample] folder and click [Open].

### Step 3


Add Program

Look in: Sample

Modelcase

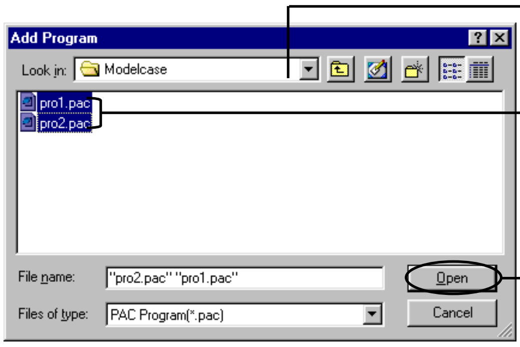
File name:

Files of type: PAC Program(*.pac)

The [Look in] window display changes to the [Sample] folder.

④ Select the [Modelcase] folder and click [Open].

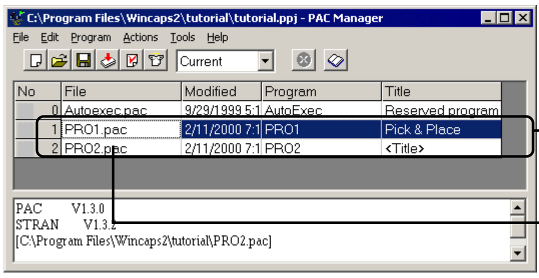
## Step 4


Add Program
Look in: Modelcase
pro1.pac
pro2.pac
File name: "pro2.pac" "pro1.pac"
Files of type: PAC Program(*.pac)
Open
Cancel

The [Look in] window display changes to the [Modelcase] folder.

⑤ Select the icons for the "PRO1.pa" and "PRO2.pac" files, and click [Open].

## Step 5


C:\Program Files\Wincaps2\tutorial\tutorial.ppj - PAC Manager
File Edit Program Actions Tools Help
Current

| No | File | Modified | Program | Title |
|---|---|---|---|---|
| 0 | Autoexec.pac | 9/29/1999 5:1 | AutoExec | Reserved program |
| 1 | PRO1.pac | 2/11/2000 7:1 | PRO1 | Pick & Place |
| 2 | PRO2.pac | 2/11/2000 7:1 | PRO2 | <Title> |

```
PAC      V1.3.0
STRAN      V1.3.2
[C:\Program Files\Wincaps2\tutorial\PRO2.pac]
```

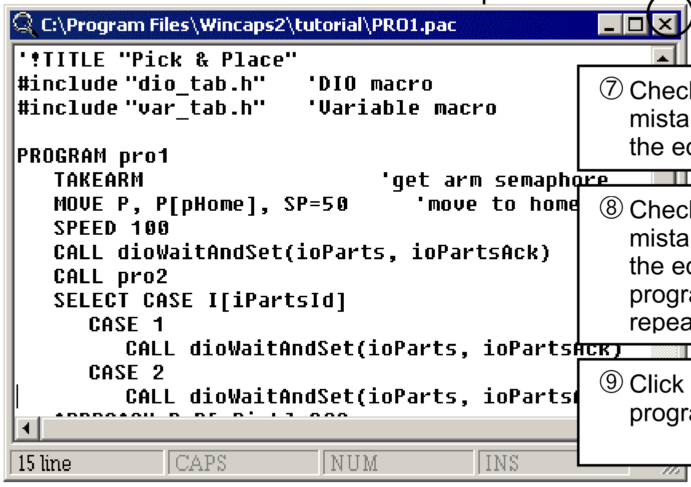
The two programs have been newly added to the list in the [PAC Manager] window.

⑥ Double-click [PRO1.pac] in the list in the [PAC Manager] window.

**★Tip★**

You can also open the program edit window by selecting [Show] in [Action] menu after clicking [PRO1.pac] to select it.

## Step 6



```
'!TITLE "Pick & Place"
#include "dio_tab.h"   'DIO macro
#include "var_tab.h"   'Variable macro

PROGRAM pro1
   TAKEARM                     'get arm semaphore
   MOVE P, P[pHome], SP=50      'move to home
   SPEED 100
   CALL dioWaitAndSet(ioParts, ioPartsAck)
   CALL pro2
   SELECT CASE I[iPartsId]
      CASE 1
         CALL dioWaitAndSet(ioParts, ioPartsAck)
      CASE 2
         CALL dioWaitAndSet(ioParts, ioParts
```

The program edit window for [PRO1.pac] is displayed.

⑦ Check that there are no mistakes in the contents of the edit window.

⑧ Check that there are no mistakes in the contents of the edit window for the other program, [PRO2.pac], by repeating steps 4 to 5.

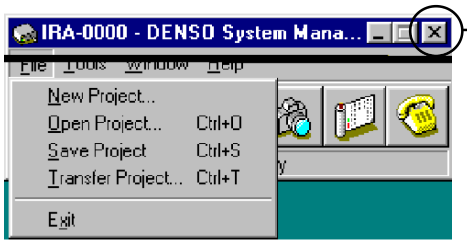
⑨ Click here to close the program edit window.

# Lesson 18 Terminating the Session

Let's close this session of Part 4 in both the PC and robot controller.

## 18.1 Terminating WINCAPSII and shutting down the personal computer

**Step 1**



① Click the closing box in the upper right corner of the [DENSO System Manager] window.

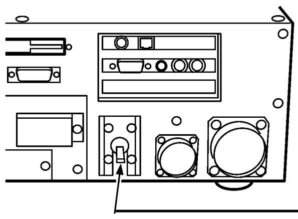
All WINCAPSII software on the PC will be exited.

**Step 2**

If needed, close the OS, such as Windows95, and shut down the PC.

## 18.2 Turning the robot controller OFF

**Step 1**



① Flip the power switch downward to turn the robot controller OFF.

# Part 5
# Features of Denso Robots

In Part 5, you will:

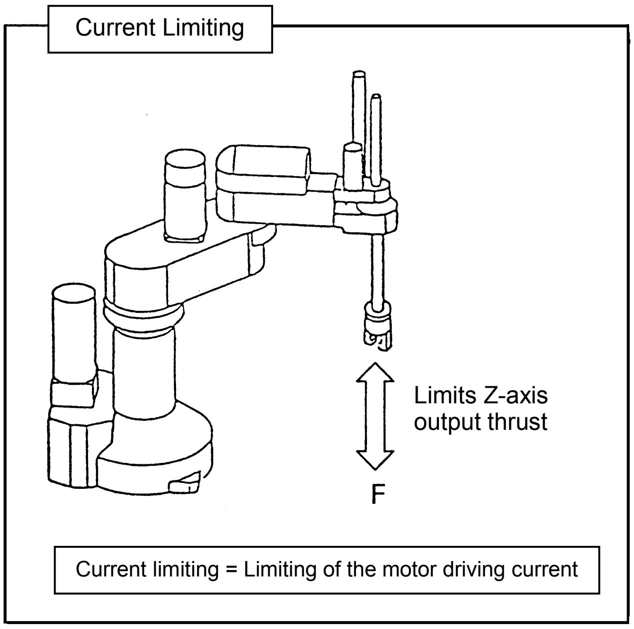
Learn the useful and advantageous features of Denso robots such as the current limiting, compliance control, and direct teaching. These features are effective for reducing installation costs and improving the efficiency of the preparation work and practical work.

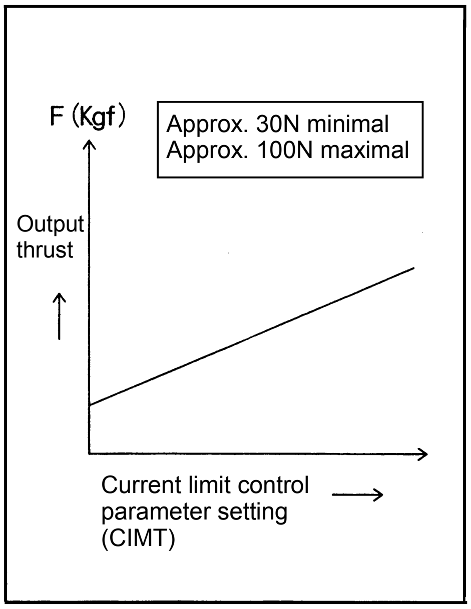
## Lesson 19  Current Limiting

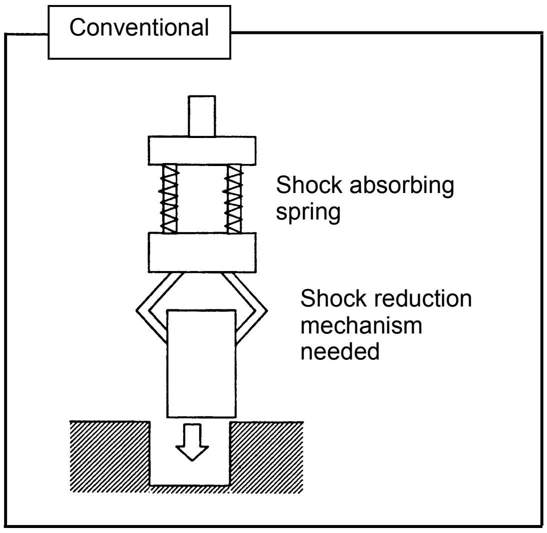
Current limiting restricts the normal operation current of the servomotor to below the set value. As the torque of the servomotor is proportional to the applied current, its torque may also be restricted by the limiting of the motor current.

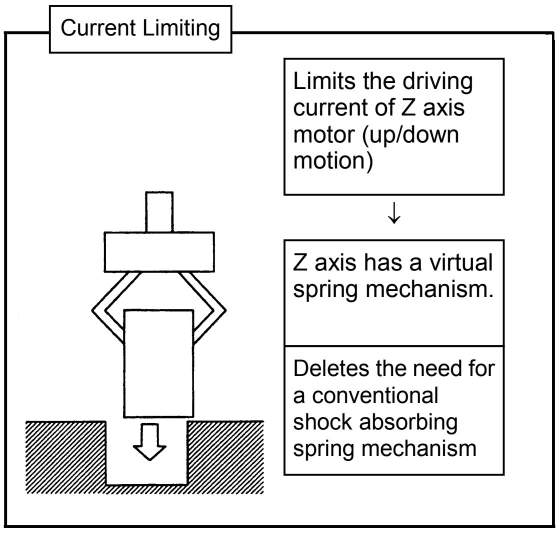
The output thrust of robots can be restricted on the direct drive axis such as the Z-axis (vertical axis) of the horizontal articulated robot. The ON/OFF or the set value of the current limit can be freely defined by the program.

If the current limit is applied to the vertical axis (Z-axis), damage to the workpiece through insertion failure and robot overcurrent error can be avoided as shown in the figures below without using the conventional shock absorbing spring mechanism.


Current Limiting
Limits Z-axis output thrust
F
Current limiting = Limiting of the motor driving current






Current Limiting
Limits the driving current of Z axis motor (up/down motion)
Z axis has a virtual spring mechanism.
Deletes the need for a conventional shock absorbing spring mechanism

## Current limit commands

Current limit commands are included in the PAC libraries and are roughly grouped into two libraries; the current limit function library and the deviation allowance value setting library. For details, refer to the Programmer's Manual.

Shown below are these libraries and the examples of application programs.

### ■ Current limit function library

| | |
|---|---|
| `SetCurlmt` | Set current limit |
| `ResetCurlmt` | Reset current limit |
| `SetForce_HM` | Set current limit for the output torque definition for Z-axis of HM/HS robots |
| `SetForce_HC` | Set current limit for the output torque definition for HC robots |

### ■ Deviation allowance value setting library

| | |
|---|---|
| `SetEralw` | Set deviation allowance value |
| `ResetEralw` | Reset deviation allowance value |

**Program examples for setting the current limit**

| | |
|---|---|
| `TAKEARM` | Acquire robot arm semaphore |
| `CALL SetEralw (3, 50)` | Set deviation allowance value of Z (3)-axis to 50mm |
| `CALL SetCurLmt (3, 30)` | Set current limit of Z(3)-axis to 30% |
| `(CALL SetForce_HM (50.0))` | Set force of Z-axis of HM robot to 50N |

**Program examples for resetting the current limit**

| | |
|---|---|
| `TAKEARM` | Acquire robot arm semaphore |
| `CALL ResetEralw (3)` | Reset deviation allowance value of Z(3)-axis to the initial value |
| `CALL ResetCurLmt (3)` | Reset current limit of Z(3)-axis (the deviation is also eliminated) |

**Note:** Remember at all times that the set value of the current limit for the robot output definition in the above examples is only for reference and not a guaranteed value.

# Lesson 20  Compliance Control (For 6-axis robot only)

Compliance control provides compliance for robots by software. This feature may absorb misalignment errors encountered when parts are mated during assembly operations or loaded into fixture and prevent robots or workpieces from undergoing excessive force.

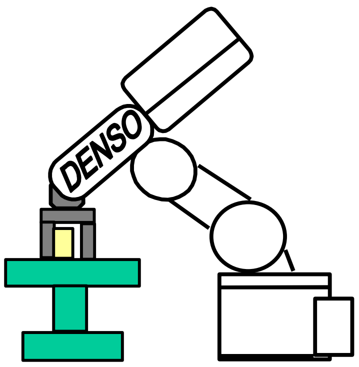
Two types of compliance control are available: one is a current limit function that sets compliance to individual joints, and the other is a tip compliance function that sets compliance to individual elements of the coordinates formed at the end of the robot flange (the mechanical interface coordinates).

**NOTE:** The current limit function is available for Ver. 1.2 or later. The tip compliance function is available for the V*-D/-E/-F/-G series, Ver. 1.4 or later.

## 20.1 Current limit function for individual axes

This function provides compliance for individual axes by limiting the drive torque (current) of each axis motor. It prevents excessive force from applying to robots or workpieces or avoids robot stops caused by an overload or overcurrent error.

**Absorbing misalignment errors in handling parts**



**Concept of current limit function**

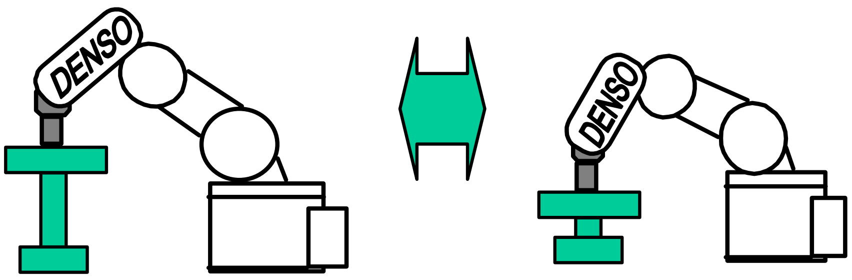
### ■ Enabling/disabling the current limit function

You may enable or disable the current limit function by executing the current limit library, SetCurLmt or ResetCurLmt, respectively. For details, refer to the PROGRAMMER'S MANUAL.

## 20.2 Tip compliance function

This function sets compliance to the individual elements of coordinates at the end of the robot flange (tip) by controlling the drive torque (current) of each axis motor based on the force limit at the tip. You can select the base coordinates, tool coordinates, or work coordinates to be applied. This function is used to make the robot follow an external force in a specified direction(s) or to make the robot touch an object for height check.

**Following an external force or touching an object for height check**



**Concept of tip compliance control**

### ■ Making the tip compliance function active from the teach pendant

This is one of the extended functions. You need to make extended functions active from the teach pendant. Once made active, the setting will be retained after the controller power is turned off. For details, refer to the PROGRAMMER'S MANUAL.

### ■ Enabling/disabling the tip compliance function

You may enable or disable the tip compliance function by executing the compliance control library, SetCompControl or ResetCompControl, respectively. For details, refer to the PROGRAMMER'S MANUAL.

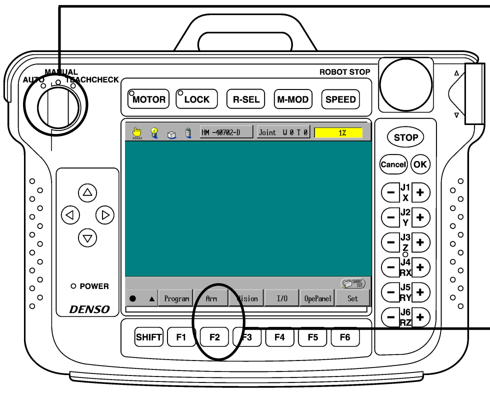
# Lesson 21 Direct Teaching (For 4-axis robot only)

Direct teaching refers to moving the robot arm directly by hand (without using the teach pendant) *with the motor OFF* and teaching the current position to a joint variable, position variable, or homogeneous transform matrix variable.

## 21.1 Entering the direct mode

Perform CAL beforehand and then proceed to the following procedure.

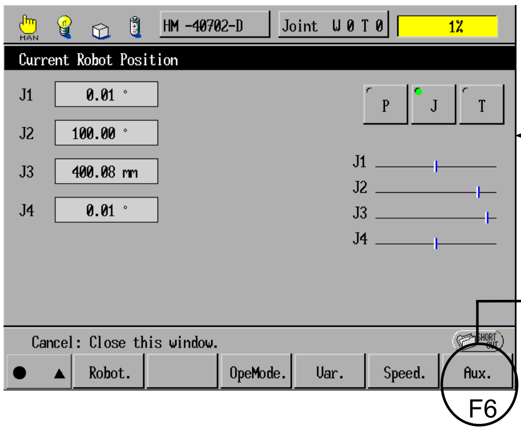
**Step 1**



① Set the mode selector switch to [MANUAL] to switch to the Manual mode.

② Press [F2 Arm] on the top screen.

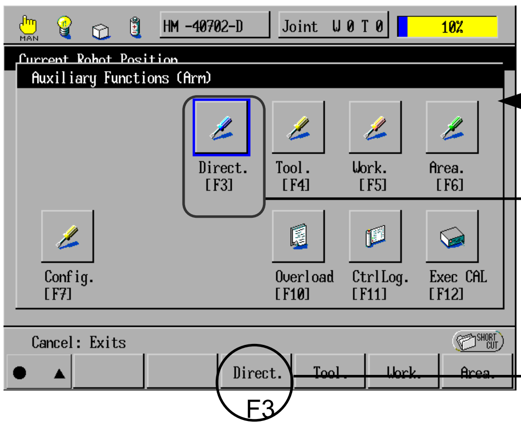
**Step 2**



The [Current Robot Position] window appears.

③ Press [F6 Aux.].

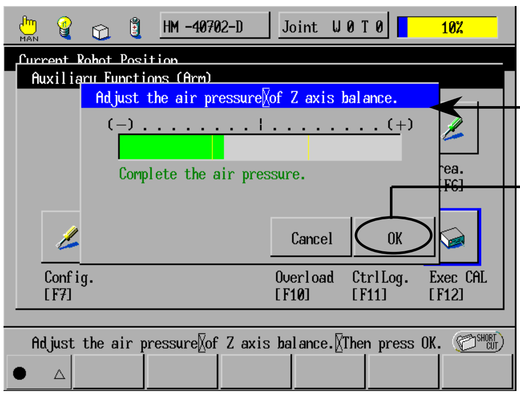
**Step 3**



The [Auxiliary Functions (Arm)] window appears.

④ Press [F3 Direct.].

**Step 4**



Adjust the air pressure of Z axis balance.

Complete the air pressure.

Adjust the air pressure of Z axis balance. Then press OK.

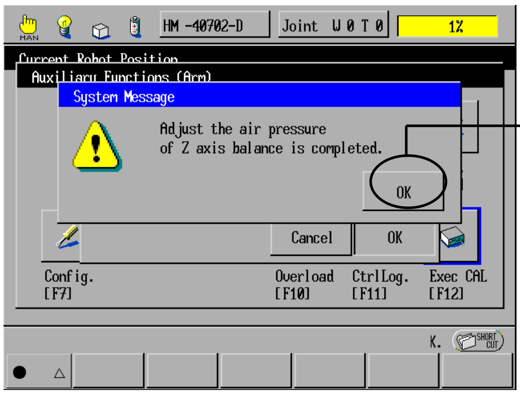
The [Adjust the air pressure of Z axis balance.] window appears.

⑤ According to the instructions given on the screen, adjust the air pressure.
If the "Complete the air pressure" is displayed as shown at left, press the OK button.

**★POINT★**

The air pressure adjustment for Z-axis balance is required only when you make the robot enter the direct teaching mode at the first time after turning the robot controller on.

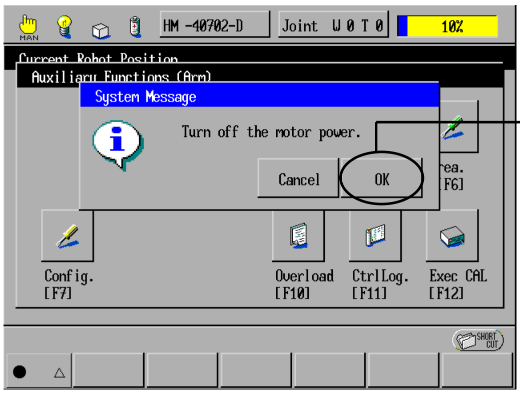
**Step 5**



Adjust the air pressure of Z axis balance is completed.

⑥ Press [OK] to finish the Z-axis air balance adjustment.

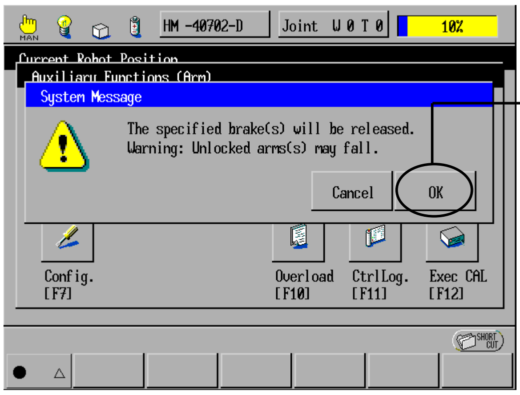
**Step 6**



Turn off the motor power.

⑦ Turn the motor off and then press [OK].

**Step 7**


System Message

The specified brake(s) will be released.
Warning: Unlocked arms(s) may fall.

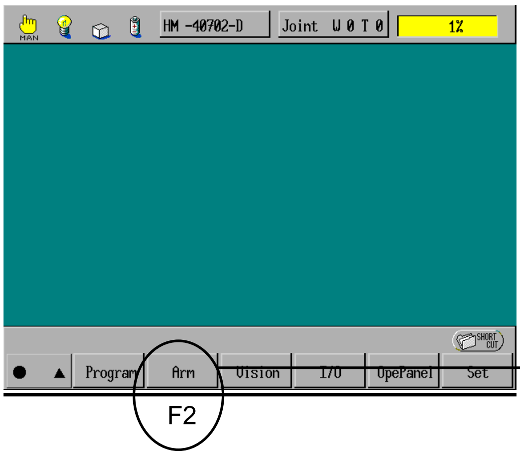
⑧ Check the message. If it is acceptable, then press [OK].

The robot is placed in the direct teaching mode.

## 21.2 Teaching the pose variables

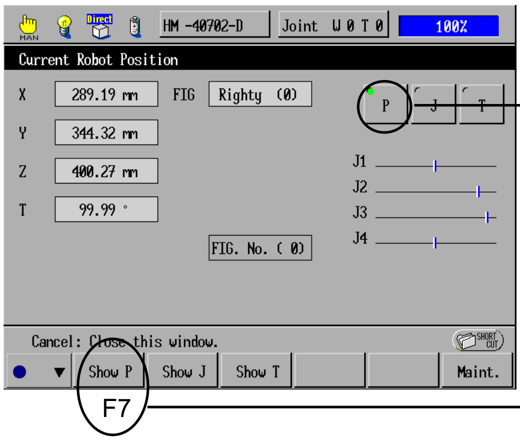
You can move the robot arm by hand easily as the Z-axis brake is released while the robot is in the direct mode. Let's move the top end of the robot by hand to the desired position and teach the position.

**Step 1**



① Press [F2 Arm] on the top screen.

**Step 2**



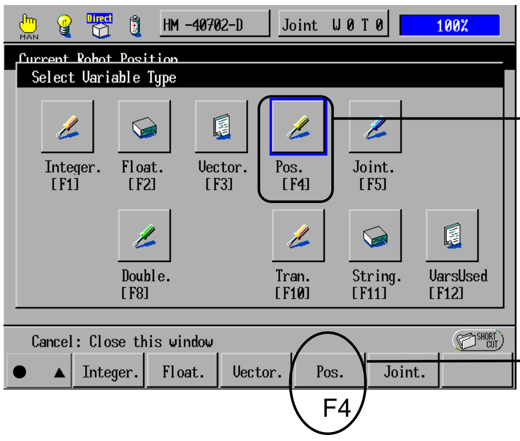
| | | | |
|---|---|---|---|
| X | 289.19 mm | FIG | Righty (0) |
| Y | 344.32 mm | | |
| Z | 400.27 mm | | |
| T | 99.99 ° | | |
| | | FIG. No. ( 0) | |

② Press [F7 Display] to display the position type. This can also be achieved by pressing [SHIFT] to display [F7 Show P] on the F1 key position and pressing F7.

**Step 3**

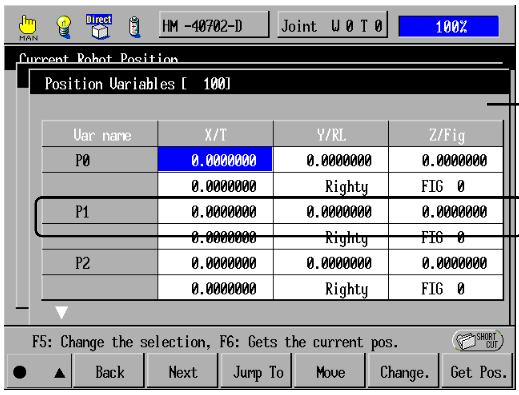
③ Move the top end of the robot arm by hand to the desired position.

**Step 4**



④ Press [F4 Pos.]

(Or press [P Position] in the [Select Variable Type] window.)



Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P0 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P1 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

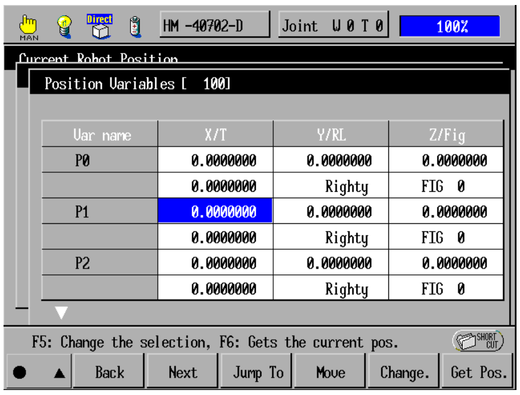
F5: Change the selection, F6: Gets the current pos.

The [Position Variables] window appears.

⑤ Select the row of the position variable number you want to teach.

This time, select variable P1. Select the row for variable P1 in the [Position Variables] window.

**Step 5**



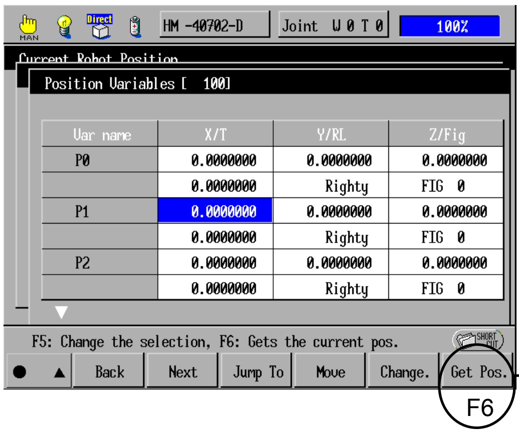
Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P0 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P1 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

F5: Change the selection, F6: Gets the current pos.

In the [Position Variables] window, six kinds of data are displayed for each variable. If any one of the six kinds of data is highlighted, it means that the variable name (P1 in this example) is selected.

**Step 6**



Position Variables [ 100]

| Var name | X/T | Y/RL | Z/Fig |
|---|---|---|---|
| P0 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P1 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |
| P2 | 0.0000000 | 0.0000000 | 0.0000000 |
| | 0.0000000 | Righty | FIG 0 |

F5: Change the selection, F6: Gets the current pos.

⑥ Press [F6 Get Pos.].

A system message appears.

**Step 7**

Check the system message and press [OK] if it is satisfactory.
The current position is read in as the value of the variable P1.

# Lesson 22 Other Features

In addition to current limiting and direct teaching, Denso robots have the following features. Refer to the Programmer's Manual for more details.

| Name | Outline of the features and commands | Related commands |
|---|---|---|
| Interference area check | An output signal is issued from the specified I/O port when the robot arm end is within the designated area.<br>These commands are useful to confirm the original position and the working position. | AREA<br>SETAREA<br>RESETAREA |
| Interrupt stop feature | When any interrupt signal of the system I/O port is ON, the current operation command is terminated and the next operation command is processed.<br>This command is useful when you want to change the robot stop position through external input.<br>However, the distance until the robot stops depends upon the speed setting of the operation command and the actual speed of the robot. | INTERRUPT ON/OFF |
| Parallel execution function of operation/non-operation command | The non-operation command can be executed during the execution of an operation command.<br>This command is useful to shorten the cycle time. | IOBLOCK ON/OFF |
| Multi-task feature | More that one program can run concurrently.<br>It is possible to shorten cycle time by checking the external interlock and executing vision commands in other programs. | RUN<br>KILL<br>etc. |
| Program debugging facilities | It is possible to display messages and sound a beeper. This is useful for debugging user programs. | PRINTMSG<br>PRINTDBG<br>BUZZER |
| Optimal payload setting feature | This makes it possible to set acceleration speed appropriate to the mass of the workpiece and end-effector.<br>This makes it possible to shorten cycle time. | aspACLD<br>aspChange |

# Appendices

# Appendix-1 Glossary

## A

**ABOVE**

One of the elbow figures of 6-axis robot. (⇔ BELOW)

**ABSOLUTE MOTION**

The motion to move to the motion target position set by teaching. (⇔ relative motion)

**ADDRESS SETTING (IP address)**

To set the controller IP address. It is required in Ethernet communication.

**APPROACH VECTOR**

Positive directional vector of Z-axis on the mechanical interface coordinates.

**AREA**

The number of white and black pixels in a window when an image data is binarized. (Vision terms)

**ARM CONFIFURATION MACRO DEFINITION FILE**

The file which contains the macro definition information of the arm setting data.

**ARM FIGURE**

The figure determined by the value of the 1st through the 3rd axes of 6-axis robot. There are two kinds of figures; RIGHTY and LEFTY.

**ARM FILE**

The file in which the information peculiar to the robot is recorded. The arm manager uses the file.

**ARM MANAGER**

The software which simulates the robot movement.

**ARM SEMAFORE**

The privilege of robot control. The task which has the privilege can operate the robot.

**AUTOMATIC ROBOT RUN**

To run the robot by executing a program.

## B

**BASE**

The portion to install the 1st axis of the robot.

**BASE COORDINATES**

The three dimensional orthogonal coordinate system which has the origin on the robot base.

**BASE MOUNTING SURFACE**

The junction surface of the base and the installation frame.

**BELOW**

One of the elbow figures of 6-axis robot. (⇔ ABOVE)

**BINARIZATION**

To change the brightness of each pixel to either white (0) or black (1) by the threshold value (binarization level).

**BINARIZATION LEVEL**

The threshold value of binarization. (Vision terms)

**BRAKE-OFF (releasing brakes)**

To release the brake of each axis.

**BRAKE-ON (locking brakes)**

To apply the brake of each axis.

**BRIGHTNESS**

The numerical value (0-255) which shows the brightness of each pixel. (Vision terms)

**BRIGHTNESS INTEGRAL VALUE**

The value which is the sum of the brightness of all the pixels in the window. (Vision terms)

## C

**CAL**

Slight movement of all axes of the robot to make the robot confirm the current position after the robot controller power on.

**CALSET**

Calibration of the relation between the actual robot position and the positional information of the controller.

**CALSET OF A SINGLE AXIS**

To perform CALSET on the specified axis only.

**CENTER OF GRAVITY**

The balance point on which the object weight balances on a plane. (Vision terms)

**COMMAND AREA**

A group of I/O ports which specify the I/O command type.

**COMMAND EXECUTION I/O SIGNAL**

The input/output signal fixed to the system in order to inform the execution of I/O command and the execution status to the outside.

**COMMAND PROCESSING COMPLETE**

The output signal to inform the completion of I/O command processing to the outside.

**COMMAND**

The instruction written in a program. The controller reads commands in the sequence written in a program, interprets commands and executes.

**COMMENT**

Explanatory notes in a program to make the program easy to understand. The controller does not execute comment.

**COMMUNICATION LOG**

The record of the communication condition between the PC and the robot.

**COMPATIBLE MODE**

The mode in which the I/O allocation is set to be compatible with the conventional series of robots. It is switched by software.

**CONTINUOUS START**

The start method to execute a program in iteration. The operation continues until it is forced to stop.

**CONTROL LOG**

The record of the specified value, the encoder value, the current value and the load ratio. They are recorded by each motion axis.

**CONVENTIONAL LANGUAGE**

The robot language used in Denso robot conventionally.

**CP CONTROL**

Compensation control to make the path from the current position to the motion target position a straight line or a circle. (⇔ PTP control)

**CURRENT POSITION**

The current position of the origin of the tool coordinates.

**CYCLE STOP**

The stop method to stop a program after one cycle execution.

## D

**D VARIABLE (Double-precision variable)**

The variable which has a value of double precision real number (15 digits of effective precision).

**DAILY INSPECTION**

The inspection before the daily work.

**DATA AREA**

A group of I/O ports to specify the necessary data for I/O command.

**DEADMAN SWITCH**

The switch which moves robot as long as any of the arm traverse keys is pressed simultaneously for safety. The robot stops immediately when either the arm traverse key or the deadman switch is released. The switch is also called "enable switch."

**DEFINING INTERFERENCE AREA**

To define the interference area. It is set either with the teach pendant, in WINCAPSII or with the program command.

**DEFINING TOOL COORDINATES**

To define tool coordinates. Origin offset amount and rotational angle amount around each axis are defined in reference to the mechanical interface coordinates. TOOL1 through TOOL63 can be defined.

**DIO MANAGER**

The software which monitors I/O condition and manages I/O allocation.

**DISCRIMINATION ANALYSIS METHOD**

The method to set the binarization level from the histogram using statistical method. (Vision terms).

**DOUBLE**

One of the 6th axis figures of 6-axis robot. (⇔ SINGLE)

**DOUBLE4**

One of the 4th axis figures of 6-axis robot. (⇔ SINGLE4)

## E

**EDGE**

Transition point of brightness. (Vision terms)

**ELBOW FIGURE**

The figure determined by the 2nd and the 3rd axis value of 6-axis robot. There are two kinds of elbow figures; ABOVE and BELOW.

**ENABLE AUTO**

The signal to enable auto mode in ON condition. Manual mode and teach check mode are possible in OFF condition.

**ENCODER VALUE CHECK MOTION**

The motion which judges that the target position is reached when the encoder value becomes within the specified pulse range toward the motion target position set by teaching.

**END MOTION**

The motion which judges that the target position is reached when the specified position of the servo coincides with the motion target position set by teaching.

**ERROR CODE**

Four digits hexadecimal code which describes error causes/conditions occurred in the robot or in WINCAPII. Refer to the error code table for the meaning of each error code.

**ERROR LOG**

Record of the error content and occurred time.

**ETERNET BOARD**

One of the controller optional boards. It is used to communicate with WINCAPSII through TCP/IP protocol.

**EXECUTION PROGRAM**

The program converted to the data format intelligible to the robot.

**EXTERNAL ACCELERATION**

The acceleration value set with the teach pendant. Percentage value to the maximum acceleration is inputted.

**EXTERNAL AUTOMATIC RUN**

To execute a program from the external device.

**EXTERNAL DECELERATION**

The deceleration value set with the teach pendant. Percentage value to the maximum acceleration is inputted.

**EXTERNAL MODE**

The mode in which robot operation is possible from the external device.

**EXTERNAL SPEED**

The speed set with the teach pendant. Percentage value to the maximum speed is inputted.

## F

**F VARIABLE (Floating-point variable)**

The variable which has a value of single precision real number (7 digits of effective precision).

**FIG**

The number which denotes the robot figure.

**FIGURE**

The possible status of each axis (joint) of the robot. Multiple figures are possible for the same position and posture.

**FIGURE COMPONENT**

The component which determine figure. There are five components in 6-axis robot; arm, elbow, wrist, the 6th axis and the 4th axis.

**FIRST ARM**

The robot arm nearest to the base.

**FLIP**

One of the wrist figures of 6-axis robot. (⇔ NONFLIP)

**FUNCTION KEYS**

The buttons provided under the pendant screen. Function names are displayed on the lower part of the screen and executes the function upon pressing the button.

## G

**GLOBAL VARIABLE**

The variable available for any task.

## H

**HALT**

The stop method to stop the program immediately. The motor power is not turned off.

**HAND (end-effector)**

The portion to hold the work. The same as tool.

**HISTOGRAM**

The occurrence ratio of the brightness value in a window. (Vision terms)

## I

**I VARIABLE (Integer variable)**

The variable which has an integer value.

**I/O**

The input and/or output signal.

**I/O COMMAND**

The process command given by the external device through the I/O port. The robot controller processes according to this command.

**INITALIZATION FLOPPY DISK**

The disk in which the initial setting of the robot at the factory shipment is recorded. It is used to recover to the initial condition when an error occurs in the controller memory.

**INSTALLATION FRAME**

The platform to install the robot.

**INTERFERENCE AREA**

The area provided by the user to watch if the tool interferes with the installation. If the origin of the tool coordinates enters into this area, output signal is issued from the specified I/O port.

**INTERNAL ACCELERATION**

The acceleration set in a program.

**INTERNAL AUTOMATIC RUN**

To execute a program from the operating panel or the teach pendant.

**INTERNAL DECELERATION**

The deceleration set in a program.

**INTERNAL MODE**

The mode in which robot run and teaching are possible using the operating panel or the teach pendant.

**INTERNAL SPEED**

The speed set in a program.

**INTERRUPT SKIP**

The input signal which halts the operation of the current step when it is ON during the execution of a robot command and starts the execution of the next step.

## J

**J VARIABLE (Joint variable)**

The variable denoted by the value of each axis.

**JOG DIAL**

The dial on the pendant which is used to move cursor or to select a path on the input screen.

**JOINT MODE**

The mode in which the robot is manually operated on each axis.

## L

**LABELING**

To number the binarized white and black area. (Vision terms)

**LEFTY**

One of the arm figures of 6-axis robot. (⇔ RIGHTY)

**LIBRARY**

The collection of programs for reuse. They are registered and utilized using the program bank of WINCAPSII.

**LOAD**

To read programs, arm data, etc. from the floppy disk into the robot controller.

**LOAD CAPACITY**

The mass of the sum of the tool and the work which the robot can hold.

**LOCAL VARIABLE**

The variable which is utilized within a task.

**LOG**

The record about operations, motions, etc. of the robot. There are four kinds of logs; error log, operation log, control log and communication log.

**LOG MANAGER**

The software which copies and manages the record of operations, errors, etc. of the robot in personal computer.

## M

**MACHINE LOCK**

The state of simulating motion by the robot controller without actual robot motion.

**MACRO**

The definition of names with 12 characters in regard to variable numbers and port numbers. Names are replaced with numbers in program execution.

**MACRO DEFINITION FILE**

The file which defines macro.

**MANUAL ROBOT OPERATION**

Robot operation by the user using the operation keys of the teach pendant or the operating panel.

**MECHANICAL END**

The mechanical motion limit set by the mechanical stopper. (⇔ Software limit)

**MECHANICAL INTERFACE**

The junction surface of the flange and the tool. Mechanical interface (JIS)

**MECHANICAL INTERFACE COORDINATES**

Three dimensional orthogonal coordinate system which has the origin on the center of the flange.

**MECHANICAL STOPPER**

The mechanism to restrict the motion of the robot axes physically.

**MENU TREE**

The description of the functional menu of function keys in tree form. It is listed on the operational guide.

**MODE METHOD**

The method to set binarization level in the valley when the histogram is two hills distribution.

**MODE SWITCH**

The switch on the pendant. It can switch the robot run mode.

**MONITOR**

To display the current status of the robot.

**MOTION SPACE**

The range in which the robot can operate.

**MULTITASKING**

The state in which multiple programs are executed virtually simultaneously. It is realized in the way that CPU of the robot controller executes each program in a short interval by turns.

## N

**NLIM**

The negative directional end value of the software limit. (⇔ PLIM)

**NONFLIP**

One of the wrist figures of 6-axis robot. (⇔ FLIP)

**NORMAL MODE**

The standard allocation mode of I/O.

**NORMAL VECTOR**

Positive directional vector of X-axis on the mechanical interface coordinates.

## O

**OERATING MODE**

The mode in which the robot is operated manually. Three are three modes; each axis mode, X-Y mode and TOOL mode.

**OERATION LOG**

The record of operations about the use of the teach pendant or the operating panel.

**OPERATING PANEL**

The fixed operating panel connected to the controller. It has no teaching function.

**OPTIMAL LOAD CAPACITY SETTING FUNCTION**

The function which sets the optimal speed and acceleration in response to the load condition or the posture of the robot.

**ORIENT VECTOR**

Positive directional vector of Y-axis on the mechanical interface coordinates.

**OVERHEAD VERSION**

The robot specified to install as it hangs from the ceiling setting the base above and the arm below. As the installation space is not needed on the working platform, working space could be wider.

**Operator**

One of the user levels of WINCAPSII. Important parameters cannot be changed. Password input is not necessary.

## P

**P TYLE METHOD**

The binarization level setting method to make the area of the object and the area of the black (or white) portion to be the same. (Vision terms)

**P VARIABLE (Position variable)**

The variable denoted by the position, the posture and the figure.

**PAC (PAC)**

New robot language used in Denso robot. It is upward compatible from SLIM. (Industrial robot language of JIS)

**PAC PROGRAM MANAGER**

The software to support PAC program development. Editor, command builder and program bank functions are included.

**PALLETIZING**

To put in or take out parts, etc. to/from the pallet with partition.

**PANEL OPERATION**

To make ON/OFF operation of the internal I/O from the teach pendant screen.

**PASS MOTION**

The motion to pass near the motion target position set by teaching.

**PENDANTLESS STATE**

To run the robot from the external device when the operating panel or the teach pendant is not connected.

**PITCH ANGLE**

The rotational angle around Y-axis.

**PIXEL**

The point which forms the screen. ( visual terms)

**PLATE MECHANICAL INTERFACE**

The portion to install tools located on the top end of the robot arm.

**PLIM**

The positive directional end value of the software limit. (⇔ NLIM)

**POSITION DATA**

The data of the base coordinates which describes the position of the robot flange center (the tool top end when the tool definition is effective) and the robot posture at the time.

**POSTURE**

The inclination of the tool determined by the roll, pitch and yaw angles in case of 6-axis robot. The tool direction determined by the angle around Z-axis in case of 4-axes robot.

**POWERING OFF THE MOTER**

To turn off the motor power of the robot.

**POWERING OFF THE ROBOT CONTROLLER**

To turn off the power of the robot controller.

**POWERING ON THE MOTER**

To turn on the motor power of the robot.

**POWERING ON THE ROBOT CONTROLLER**

To turn on the power of the robot controller.

**PRINCIPAL AXIS**

The axis which gives the minimum moment of inertia in case of rotating the object on a plane. (Vision terms)

**PRINCIPAL AXIS ANGLE**

The angle formed by the horizontal axis and the principal axis. (Vision terms)

**PRIORITY**

The sequence of task execution in order of importance. The program with higher priority is executed first.

**PROGRAM RESET**

The input signal to force program execution from the top of the program.

**PROGRAM START**

The input signal to start a program. When it is a step stop, execution begins from the next step and when it is a halt, execution begins from the following of the same step.

**PROGRAM TRANSFER**

To send/receive robot programs between the robot controller and WINCAPSII (PC).

**PTP CONTROL**

The control which moves the robot arm to the target position without compensation. The path may not necessarily be a straight line. (⇔ CP control)

**Programmer**

One of the user levels of WINCAPSII. All the common operations are possible. Password input is necessary to enter into this mode.

## R

**RANG**

The angle which determines the relation of the robot standard position and the mechanical end.

**RELATIVE MOTION**

The motion to move from the current position for the motion amount set by teaching.

**REMOTE OPERATION**

To operate the robot arm which is displayed on the arm manager.

**RIGHTY (RIGHTY)**

One of the arm figures of 6-axis robot. (⇔ LEFTY)

**ROBOT ERROR**

The output signal which informs that an error condition occurred in the robot such as servo error, program error, etc.

**ROBOT STOP**

The stop method to stop programs immediately and power off the motor.

**ROBOT WARNING**

The output signal which informs that a slight error occurred during I/O command or servo processing.

**ROLL ANGLE**

The rotational angle around Z-axis.

**RX COMPONENT**

The amount of rotational angle around the X coordinate axis.

**RY COMPONENT**

The amount of rotational angle around the Y coordinate axis.

**RZ COMPONENT**

The amount of rotational angle around the Z coordinate axis.

## S

**SAVE**

To save programs, arm data, etc. onto the floppy disk from the robot controller.

**SEARCH**

To search the space which coincides with a standardized image data (search model). (Vision terms)

**SECOND ARM**

The farther arm of the robot arms measured from the base.

**SEMAPHORE**

The task execution privilege which is used to synchronize among tasks or to do exclusive control among the tasks that must not be executed simultaneously.

**SERVO ON**

The signal to inform to the outside that the motor power is on.

**SET COMMUNICATION**

To set the usage conditions (communication speed, etc.) of each communication port of the robot controller.

**SET COMMUNICATION PERMISSION**

To set the usage permission of each communication port of the robot controller.

**SINGLE**

One of the 6th axis figures of 6-axis robot. (⇔ DOUBLE)

**SINGLE-CYCLE START**

The start method to make a program execute one cycle. The program stops after one cycle execution (to the last step of the program).

**SINGLE-STEP START**

The start method to make a program execute one step. The program stops after one step execution.

**SINGLE4**

One of the 4th axis figures of 6-axis robot. (⇔ DOUBLE4)

**SINGULAR POINT**

The position on the boundary of the two figures.

**SNAPSHOT**

The function to record the current status of the robot.

**SOFTWARE LIMIT**

The limit of the robot motion range determined by the software. (⇔ mechanical end)

**STATUS AREA**

A group of output signals to inform the result of I/O command processing. The status corresponding to the I/O command is set.

**STEP CHECK**

One step execution of a program in teach check mode.

**STEP STOP**

The stop method to stop a program after one step execution.

**STOP KEY**

One of the pendant buttons. Pressing the button makes all programs halt immediately.

**STROBE SIGNAL**

The input signal to instruct the start of I/O command processing.

**SUBROUTINE**

The program which describes a specific motion and is called from a portion of a main program.

**SYSTEM I/O SIGNALS**

The input/output signals fixed to the system in order to inform the run control or run condition to the outside.

**SYSTEM MANAGER**

The software which generally manages all the information of WINCAPSII.

**SYSTEM PROJECT**

Programs and related data groups which are managed by the system manager.

**SYSTEM VARIABLE**

The variable to check the system condition in a program.

## T

**T VARIABLE (Homogeneous transform matrix variable)**

The variable denoted by the position vector, the orient vector, the approach vector and the figure.

**TASK**

The motion process formed by each program when multiple programs are managed their simultaneous execution.

**TEACH CHECK**

To check the motion by the program.

**TEACHING**

To input the necessary information for operation into the robot using the teach pendant.

**TOOL**

The portion of the robot which affects the work immediately. It is a synonym of end-effector (JIS).

**TOOL COORDINATES**

The coordinate system which sets the origin on the tool and offsets the origin of the mechanical interface coordinates to any point and rotates around each axis.

**TOOL MODE**

The manual operation mode on the tool coordinates.

**TOOL0**

A special form of tool definition that has origin offset zero, i.e. it implies the mechanical interface coordinates.

**TYPE DECLARATION**

To declare the type of variable in a program.

## U

**USER COORDINATES**

The coordinate system which users can define.

**USER I/O SIGNALS**

The input/output signals controllable by the user program.

**USER LEVEL**

The class provided for users to keep data management security. Access to information or operation is restricted by each class.

## V

**VARIABLE TABLE**

A group of data which are the pair of each port number and value retained by the controller.

**VISUAL DEVICE**

The device to provide the robot with necessary data by processing the images inputted from the camera.

**VISUAL FUNCTION**

The function to provide the robot control function with necessary data by processing the images inputted from the camera.

## W

**WINDOW**

The space to process images. (Vision terms)

**WORK COORDINATES**

The three dimensional orthogonal coordinate system which sets the origin on the work to be processed by the robot.

**WRIST FIGURE**

The figure determined by the value of the 4th and the 5th axis of the 6-axis robot. There are two kinds of wrist figures; FLIP and NONFLIP.

## X

**X-Y MODE**

The manual operation mode on the base coordinates.

## Y

**YAW ANGLE**

The rotational angle around X-axis.

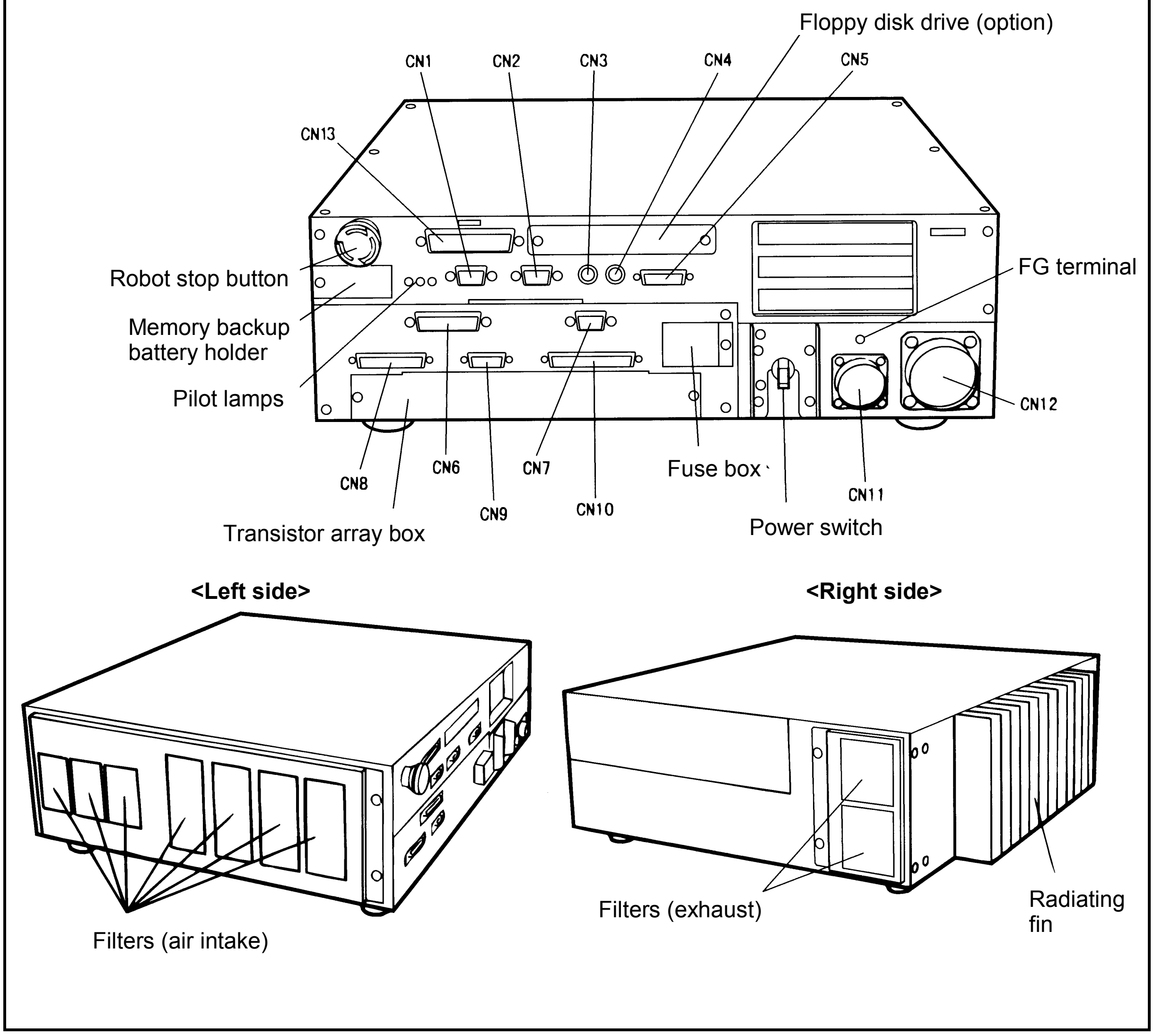
## SYMBOLS

**µVision**

Visual device manufactured by Denso.

## Appendix-2 Names of the robot controller parts (RC5)

The figure and table given below show the names of the robot controller parts.



**Names of Robot Controller Parts**

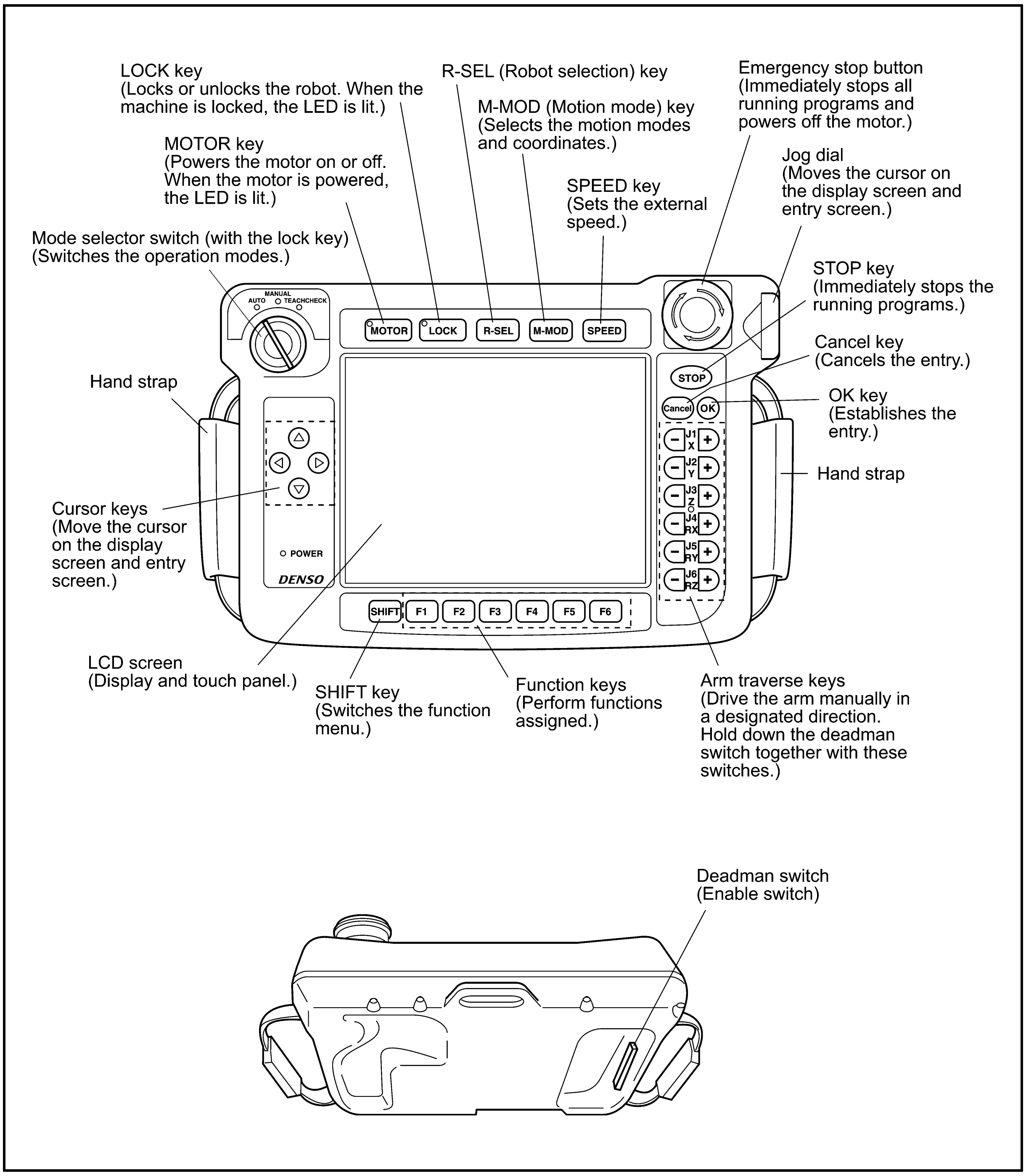
**Connector Names**

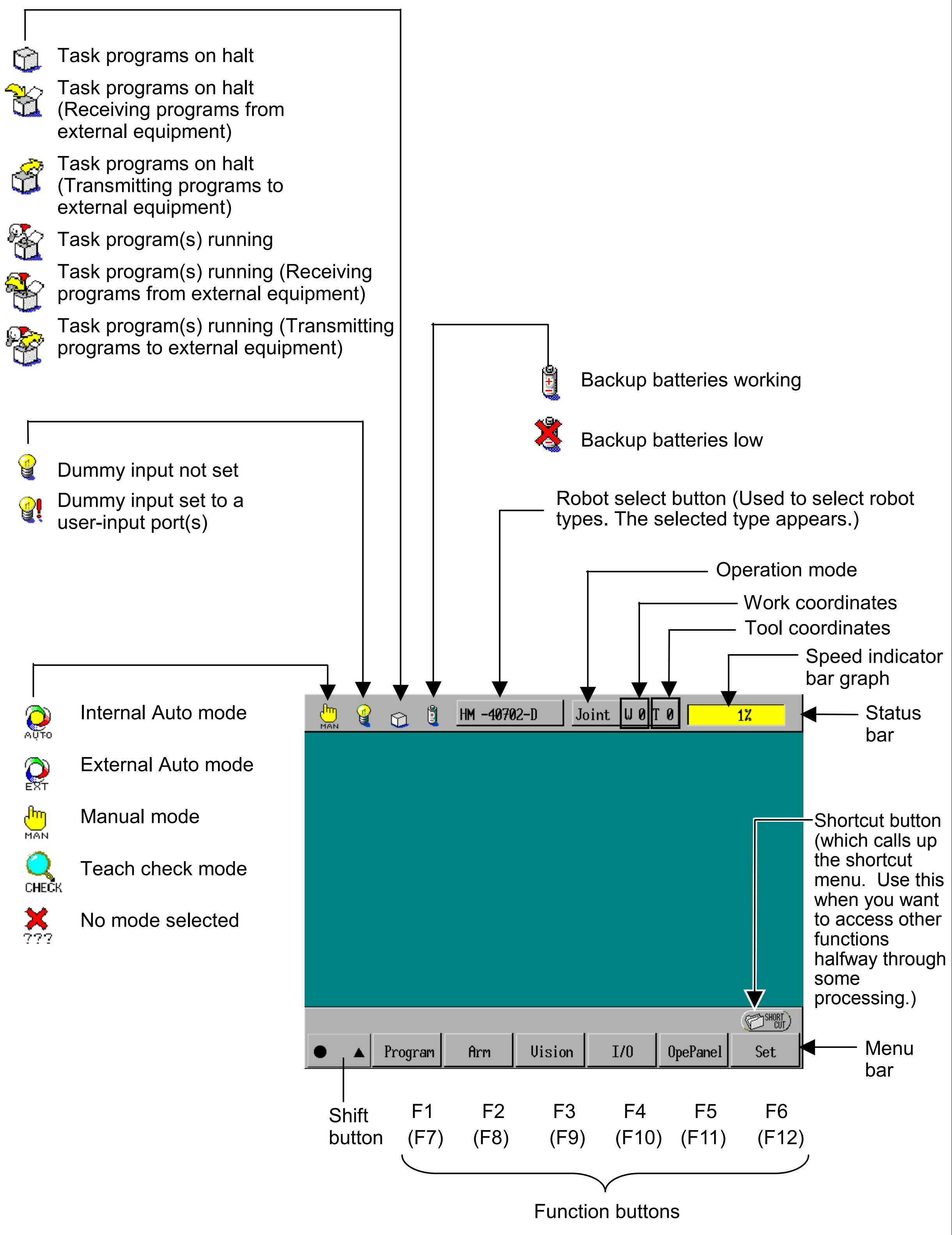
| Connector No. | Marking | Name | Connector No. | Marking | Name |
|---|---|---|---|---|---|
| CN1 | RS232C | Serial interface connector | CN8 | INPUT | Connector for user input or system input |
| CN2 | CRT | Connector for CRT | CN9 | HAND I/O | Connector for end-effector I/O |
| CN3 | KEYBD | Connector for keyboard | CN10 | OUTPUT | Connector for user output or system output |
| CN4 | MOUSE | Connector for PS/2 mouse | CN11 | INPUT AC | Power connector |
| CN5 | PENDANT | Connector for pendant | CN12 | MOTOR | Connector for motor |
| CN6 | PRINTER | Connector for printer | CN13 | ENCODER | Connector for encoder |
| CN7 | I/O POWER | Power connector for I/O | | | |

⚠Caution: The robot controller connectors are of a screw-lock type or ring-lock type. Lock the connectors securely. If even one of the connectors is not locked, incomplete contact may result thereby causing an error.

Connecting or disconnecting the power connector or motor connector when the robot controller power switch is ON may cause damage to the internal circuits of the robot controller. Turn OFF the power switch before connecting or disconnecting these connectors.

# Appendix-3  Names of the teach pendant parts


LOCK key
(Locks or unlocks the robot. When the machine is locked, the LED is lit.)
R-SEL (Robot selection) key
Emergency stop button
(Immediately stops all running programs and powers off the motor.)
MOTOR key
(Powers the motor on or off. When the motor is powered, the LED is lit.)
M-MOD (Motion mode) key
(Selects the motion modes and coordinates.)
Jog dial
(Moves the cursor on the display screen and entry screen.)
SPEED key
(Sets the external speed.)
Mode selector switch (with the lock key)
(Switches the operation modes.)
STOP key
(Immediately stops the running programs.)
MANUAL
AUTO
TEACHCHECK
MOTOR
LOCK
R-SEL
M-MOD
SPEED
Cancel key
(Cancels the entry.)
Hand strap
STOP
Cancel
OK
OK key
(Establishes the entry.)
J1 X
J2 Y
J3 Z
J4 RX
J5 RY
J6 RZ
Hand strap
Cursor keys
(Move the cursor on the display screen and entry screen.)
POWER
DENSO
SHIFT
F1
F2
F3
F4
F5
F6
LCD screen
(Display and touch panel.)
SHIFT key
(Switches the function menu.)
Function keys
(Perform functions assigned.)
Arm traverse keys
(Drive the arm manually in a designated direction. Hold down the deadman switch together with these switches.)
Deadman switch
(Enable switch)


Task programs on halt
Task programs on halt (Receiving programs from external equipment)
Task programs on halt (Transmitting programs to external equipment)
Task program(s) running
Task program(s) running (Receiving programs from external equipment)
Task program(s) running (Transmitting programs to external equipment)
Backup batteries working
Backup batteries low
Dummy input not set
Dummy input set to a user-input port(s)
Robot select button (Used to select robot types. The selected type appears.)
Operation mode
Work coordinates
Tool coordinates
Speed indicator bar graph
AUTO
Internal Auto mode
EXT
External Auto mode
MAN
Manual mode
CHECK
Teach check mode
???
No mode selected
MAN
HM -40702-D
Joint
W 0
T 0
1%
Status bar
Shortcut button (which calls up the shortcut menu. Use this when you want to access other functions halfway through some processing.)
SHORT CUT
●
▲
Program
Arm
Vision
I/O
OpePanel
Set
Menu bar
Shift button
F1 (F7)
F2 (F8)
F3 (F9)
F4 (F10)
F5 (F11)
F6 (F12)
Function buttons

**Top screen**

# Appendix-4 Menu tree of the teach pendant

Top Screen

- [F1 Program] in Manual Mode
  - [F1 NewProg.]
  - [F2 Delete]
  - [F3 Copy]
  - [F4 Var.]

    (The following menus have some submenus of F1 to F12.)
    - [F1 Integer.]**
    - [F2 Float.] · · ·
    - [F3 Vector.] · · ·
    - [F4 Pos.]* · · ·
    - [F5 Joint.]* · · ·
    - [F6 RegVar.]*** · ·
    - [F8 Double.] · · ·
    - [F10 Tran.]* · · ·
    - [F11 String.] · · ·
    - [F12 VarsUsed]. · ·
      - [F1 Back]
      - [F2 Next]
      - [F3 Jump To], [F3 Search]***
      - [F4 Move]*, [F4 Switch]**, [F4 Delete]***
      - [F5 Change.], [F5 Display]***
      - [F6 Get Pos.]*
      - [F7 Copy Var]
      - [F10 Del All]***
      - [F12 Register]
  - [F5 Edit.]
    - [F1 NewLine.]
      - [F1 User.]
      - [F2 Flow.]
      - [F3 Robot.]
      - [F4 Category]
      - [F5 Recent.]
      - [F6 Clr All]
    - [F2 Del Line]
    - [F3 CopyLine]
    - [F4 Paste]
    - [F5 EditLine]
      - [F1 User.]
      - [F2 Flow.]
      - [F3 Robot.]
      - [F4 Category]
      - [F5 Recent.]
      - [F6 Clr All]
    - [F6 Save.]
    - [F10 SyntxErr]
    - [F11 SetBP]
  - [F6 Aux.]
    - [F1 Set PRJ.]
    - [F3 Options.]
    - [F5 BPSettng]
    - [F7 Continue]
    - [F8 SS Mode.]
    - [F9 StpBack.]
    - [F10 LoadMode]
    - [F12 Compile]
  - [F7 New PRJ]
  - [F10 SyntxErr]
  - [F12 Config.]

Next page

- [F1 Program]
  in Teach Check
  Mode
  - [F1 Halt]
  - [F2 StepStop]
  - [F4 CycStart]
  - [F5 StepBack]
  - [F6 StpStart]
  - [F7 ProgRst.]
  - [F9 Priorty.]
  - [F11 Display.]
  - [F12 PrintDbg]
    - [F1 Back]
    - [F2 Next]
    - [F6 ClrDisp]
- [F1 Program]
  in Auto Mode
  - [F1 Halt]
  - [F2 StepStop]
  - [F3 CycStop]
  - [F4 Strart.]
  - [F6 StpStart]
  - [F7 ProgRst.]
  - [F9 Priorty.]
  - [F10 Continue]
  - [F11 Display.]
  - [F12 PrintDbg]
    - [F1 Back]
    - [F2 Next]
    - [F6 ClrDisp]

Next page

```
[F2  Arm] ─┐
           ├[F1  Robot.]
           ├[F3  OpeMode.] ─┐
           │                ├ [F1  Back]
           │                ├ [F2  Next]
           │                ├ [F5  Work No.]
           │                └ [F6  Tool No.]
           │
           │                  (The following menus have some submenus of F1 to F12.)
           ├[F4  Var.] ───────┬[F1  Integer.]** ──┐
           │                  ├[F2  Float.]    ···├ [F1  Back]
           │                  ├[F3  Vector.]   ···├ [F2  Next]
           │                  ├[F4  Pos.]*     ···├ [F3  Jump To], [F3 Search]***
           │                  ├[F5  Joint.]    ···├ [F4  Move]*, [F4 Switch]**, [F4 Delete]***
           │                  ├[F6  RegVar.]***···├ [F5  Change.]
           │                  ├[F8  Double.]   ···├ [F6  Get Pos.]*
           │                  ├[F10 Tran.]     ···├ [F7  Copy Var]
           │                  ├[F11 String.]   ···├ [F10 Del All]***
           │                  └[F12 VarsUsed]  ···└ [F12 Register]
           ├[F5  Speed.] ─────┐
           │                  ├ [F1  1%]
           │                  ├ [F2  10%]
           │                  ├ [F3  50%]
           │                  ├ [F4  100%]
           │                  └ [F5  Change.]
           │
           │                  (F4 to F7 have submenus F1 to F6.
           │                  Submenu marked with * is only for [F5 Work.] and [F6 Area.].
           │                  Submenu marked with ** is only for [F6 Area.].)
           ├[F6  Aux.] ───────┐
           │                  ├ [F3  Direct.] (For 4-axis robot)
           │                  ├ [F4  Tool.] ──────┐
           │                  ├ [F5  Work.]   ····├[F1  Back]
           │                  ├ [F6  Area.]   ····├[F2  Next]
           │                  ├ [F7  Config.] ····├[F3  Jump To]
           │                  ├ [F10 Overload]    ├[F4  AutoCalc]*
           │                  │                   ├[F5  Change.]
           │                  │                   └[F6  Activate]**
           │                  │
           │                  ├ [F11 CtrlLog.] ───┐
           │                  │                   ├[F1  StrtLog]
           │                  │                   ├[F2  StopLog]
           │                  │                   ├[F6  ClrLog]
           │                  │                   ├[F7  SaveLog.]
           │                  │                   └[F12 DelLog.]
           │                  │
           │                  └ [F12 Exec CAL]
           ├[F7  Show P]
           ├[F8  Show J]
           ├[F9  Show T]      (F1 and F2 have submenus F1 to F5 except F4.)
           └[F12 Maint.] ─────┐
                              ├ [F1  M Space.] ───┐
                              ├ [F2  RANG.]   ····├[F1  Back]
                              │                   ├[F2  Next]
                              │                   ├[F3  Jump To]
                              │                   └[F5  Change.]
                              │
                              ├ [F3  Brake.] ─────┬[F4  CanclAll]
                              ├ [F4  Adj.Z.Bal]   ├[F5  ON/OFF]
                              │ (For 4-axis robot)└[F6  SelctAll]
                              ├ [F6  CALSET.] ─┘
                              ├ [F10 ENC inf.]
                              ├ [F11 ENC rst]
                              └ [F12 ENC set]
```

Next page

```
├─ [F3  Vision] ─┬─ [F1  Camera] ──────┬─ [F1  Update]
│                │                     ├─ [F2  Live]
│                │                     └─ [F5  Change]
│                ├─ [F2  Display] ─────┬─ [F1  Update]
│                │                     └─ [F5  Change]
│                ├─ [F3  CLS] ─────────┬─ [F3  Change]
│                │                     ├─ [F4  CanclAll]
│                │                     ├─ [F5  Sel/Canc]
│                │                     └─ [F6  SelctAll]
│                ├─ [F4  Drawing]
│                ├─ [F6  Monitor] ─────┬─ [F1  Update]
│                │                     ├─ [F2  Continue]
│                │                     └─ [F3  Stop]
│                ├─ [F7  Window] ──────┬─ [F1  New] ──────┬─ [F1  1 pixel]
│                │  (Ver. 1.5 or later)│                  ├─ [F2  10 pixel]
│                │                     │                  ├─ [F3  50 pixel]
│                │                     │                  └─ [F5  Change]
│                │                     ├─ [F2  Edit] ─────┬─ [F1  1 pixel]
│                │                     ├─ [F3  Del]       ├─ [F2  10 pixel]
│                │                     ├─ [F5  Change]    ├─ [F3  50 pixel]
│                │                     ├─ [F7  Capture]   └─ [F5  Change]
│                │                     └─ [F8  Live]
│                ├─ [F8  Model] ───────┬─ [F1  Display]
│                │                     ├─ [F2  New] ──────┬─ [F1  1 pixel]
│                │                     │                  ├─ [F2  10 pixel]
│                │                     ├─ [F3  Delete]    ├─ [F3  50 pixel]
│                │                     ├─ [F5  Change]    └─ [F5  Change]
│                │                     ├─ [F7  Capture]
│                │                     └─ [F8  Live]
│                ├─ [F9  Analysis] ────┬─ [F1  Capture]
│                │  (Ver. 1.5 or later)├─ [F2  Live]
│                │                     ├─ [F3  Window] ───┬─ [F1  1 pixel]
│                │                     ├─ [F4  Model]     ├─ [F2  10 pixel]
│                │                     ├─ [F5  Change]    ├─ [F3  50 pixel]
│                │                     ├─ [F7  Normal]    ├─ [F5  Change]
│                │                     └─ [F8  Binary]    ├─ [F6  Number]
│                │                                        ├─ [F7  Capture]
│                │                                        └─ [F8  Live]
│                ├─ [F11  Options] ────┬─ [F1  Back]
│                │                     ├─ [F2  Next]
│                │                     ├─ [F3  Jump To]
│                │                     └─ [F5  Change.]
│                └─ [F12  Init.]
│
```

Next page

```
├─[F4  I/O.]
│   ├─ [F1  Back]
│   ├─ [F2  Next]
│   ├─ [F3  Jump To]
│   ├─ [F4  Dummy IN]
│   ├─ [F5  ON/OFF]
│   ├─ [F6  Aux.]
│   │   ├─ [F1  Set H/W]
│   │   ├─ [F2  AlocMode]
│   │   ├─ [F3  Sw Disp]
│   │   ├─ [F7  I/O Lock]
│   │   ├─ [F8  MasterPrm]
│   │   │      ([F1 Set H/W], [F8 MasterPrm])
│   │   │       ├─ [F1  Back]
│   │   │       ├─ [F2  Next]
│   │   │       ├─ [F3  Jump To]
│   │   │       └─ [F5  Change.]
│   │   ├─ [F9  SlaveMap]
│   │   │   ├─ [F1  InArea]
│   │   │   ├─ [F2  OutArea]
│   │   │   ├─ [F4  Scan]
│   │   │   └─ [F5  Change]
│   │   ├─ [F11 DevAssign]
│   │   └─ [F12 MastrStat]
│   │       ├─ [F1  Back]
│   │       ├─ [F2  Next]
│   │       └─ [F3  Jump To]
│   └─ [F10 ClrDummy]
│
├─[F5  OpePanel]
│   ├─ [F1  Back]
│   └─ [F2  Next]
│
├─[F6  Set]
│   ├─ [F1  Load!]
│   ├─ [F2  Log.]
│   │   ├─ [F1  ErrLog]
│   │   └─ [F2  OpeLog]
│   ├─ [F3  FD.]
│   │  [Ver. 1.99 or earlier]
│   │   ├─ [F1  Read.]  ─┬─ [F4  CanclAll]
│   │   ├─ [F2  Write]  ─┤  [F5  Sel/Canc]
│   │   │                └─ [F6  SelctAll]
│   │   ├─ [F5  Format.]
│   │   └─ [F12  Aux.] ──── [F11 CtrlLog.]
│   ├─ [F3  USB.]
│   │  [Ver. 2.2 or later]
│   │   ├─ [F1  Read.]  ─┬─ [F4  CanclAll]
│   │   ├─ [F2  Write]  ─┤  [F5  Sel/Canc]
│   │   │                └─ [F6  SelctAll]
│   │   └─ [F12  Aux.] ──── [F11  CtrlLog.]
│   ├─ [F4  MemoInfo]
│   ├─ [F5  Set Com.]
│   │   ├─ [F1  Permit.] ──── [F5  Change.]
│   │   ├─ [F2  SerialIF] ─┬─ [F4  Default.]
│   │   ├─ [F3  Modem]     └─ [F5  Change.]
│   │   ├─ [F4  Address] ──── [F5  Change.]
│   │   ├─ [F5  Gateway] ──── [F5  Change.]
│   │   ├─ [F7  Hispeed!]
│   │   ├─ [F10  Client]
│   │   └─ [F11  Server]
│   ├─ [F6  Maint.]
│   │   ├─ [F1  Total h]
│   │   │   ├─ [F4  Cumu. o]
│   │   │   └─ [F5  Cumu. r]
│   │   ├─ [F2  Version]
│   │   ├─ [F3  Date.] ──── [F5  Change]
│   │   ├─ [F4  Battery] ── [F5  Change]
│   │   └─ [F5  Odometer] ─ [F6  Reset]
│   ├─ [F7  Options.]
│   │   ├─ [F3  Protect.]
│   │   ├─ [F6  Language]
│   │   ├─ [F8  Extnsion]
│   │   │   ├─ [F4  Remove]
│   │   │   └─ [F5  Input ID]
│   │   ├─ [F11 ROBTYPE]
│   │   └─ [F12 Update.]
│   ├─ [F8  Save!]
│   └─ [F9  SaveFile]
│
├─[F9  Panel]
├─[F10 Int/Ext]
└─[F11 Unplug]
```

**Vertical Articulated V＊-D/-E/-F/-G Series**
**Horizontal Articulated H＊-D/-E/-G Series**
**Cartesian Coordinate XYC-4D Series**

**BEGINNER'S GUIDE**

First Edition March 2000
Fifth Edition July 2005

DENSO WAVE INCORPORATED
Factory Automation Division

7G＊＊C