MITSUBISHI ELECTRIC

Mitsubishi Industrial Robot

CR750/CR751 series controller

High Speed and Accuracy Tracking Function Instruction Manual

MELFA

BFP-A3382

⚠ Safety Precautions

Always read the following precautions and the separate "Safety Manual" before starting use of the robot to learn the required measures to be taken.

⚠CAUTION All teaching work must be carried out by an operator who has received special training.
(This also applies to maintenance work with the power source turned ON.)
→Enforcement of safety training

⚠CAUTION For teaching work, prepare a work plan related to the methods and procedures of operating the robot, and to the measures to be taken when an error occurs or when restarting. Carry out work following this plan.
(This also applies to maintenance work with the power source turned ON.)
→Preparation of work plan

⚠WARNING Prepare a device that allows operation to be stopped immediately during teaching work.
(This also applies to maintenance work with the power source turned ON.)
→Setting of emergency stop switch

⚠CAUTION During teaching work, place a sign indicating that teaching work is in progress on the start switch, etc.
(This also applies to maintenance work with the power source turned ON.)
→Indication of teaching work in progress

⚠DANGER Provide a fence or enclosure during operation to prevent contact of the operator and robot.
→Installation of safety fence

⚠CAUTION Establish a set signaling method to the related operators for starting work, and follow this method.
→Signaling of operation start

⚠CAUTION As a principle turn the power OFF during maintenance work. Place a sign indicating that maintenance work is in progress on the start switch, etc.
→Indication of maintenance work in progress

⚠CAUTION Before starting work, inspect the robot, emergency stop switch and other related devices, etc., and confirm that there are no errors.
→Inspection before starting work

The points of the precautions given in the separate "Safety Manual" are given below.
Refer to the actual "Safety Manual" for details.

⚠DANGER When automatic operation of the robot is performed using multiple control devices (GOT, programmable controller, push-button switch), the interlocking of operation rights of the devices, etc. must be designed by the customer.

⚠CAUTION Use the robot within the environment given in the specifications. Failure to do so could lead to faults or a drop of reliability.
(Temperature, humidity, atmosphere, noise environment, etc.)

⚠CAUTION Transport the robot with the designated transportation posture. Transporting the robot in a non-designated posture could lead to personal injuries or faults from dropping.

⚠CAUTION Always use the robot installed on a secure table. Use in an instable posture could lead to positional deviation and vibration.

⚠CAUTION Wire the cable as far away from noise sources as possible. If placed near a noise source, positional deviation or malfunction could occur.

⚠CAUTION Do not apply excessive force on the connector or excessively bend the cable. Failure to observe this could lead to contact defects or wire breakage.

⚠CAUTION Make sure that the workpiece weight, including the hand, does not exceed the rated load or tolerable torque. Exceeding these values could lead to alarms or faults.

⚠WARNING Securely install the hand and tool, and securely grasp the workpiece. Failure to observe this could lead to personal injuries or damage if the object comes off or flies off during operation.

⚠WARNING Securely ground the robot and controller. Failure to observe this could lead to malfunctioning by noise or to electric shock accidents.

⚠CAUTION Indicate the operation state during robot operation. Failure to indicate the state could lead to operators approaching the robot or to incorrect operation.

⚠WARNING When carrying out teaching work in the robot's movement range, always secure the priority right for the robot control. Failure to observe this could lead to personal injuries or damage if the robot is started with external commands.

⚠CAUTION Keep the jog speed as low as possible, and always watch the robot. Failure to do so could lead to interference with the workpiece or peripheral devices.

CAUTION After editing the program, always confirm the operation with step operation before starting automatic operation. Failure to do so could lead to interference with peripheral devices because of programming mistakes, etc.

CAUTION Make sure that if the safety fence entrance door is opened during automatic operation, the door is locked or that the robot will automatically stop. Failure to do so could lead to personal injuries.

CAUTION Never carry out modifications based on personal judgments, non-designated maintenance parts. Failure to observe this could lead to faults or failures.

WARNING When the robot arm has to be moved by hand from an external area, do not place hands or fingers in the openings. Failure to observe this could lead to hands or fingers catching depending on the posture.

CAUTION Do not stop the robot or apply emergency stop by turning the robot controller's main power OFF. If the robot controller main power is turned OFF during automatic operation, the robot accuracy could be adversely affected. Also a dropped or coasted robot arm could collide with peripheral devices.

CAUTION Do not turn OFF the robot controller's main power while rewriting the robot controller's internal information, such as a program and parameter. Turning OFF the robot controller's main power during automatic operation or program/parameter writing could break the internal information of the robot controller.

DANGER Do not connect the Handy GOT when using the GOT direct connection function of this product. Failure to observe this may result in property damage or bodily injury because the Handy GOT can automatically operate the robot regardless of whether the operation rights are enabled or not.

DANGER Do not connect the Handy GOT to a programmable controller when using an iQ Platform compatible product with the CR750-Q/CR751-Q controller. Failure to observe this may result in property damage or bodily injury because the Handy GOT can automatically operate the robot regardless of whether the operation rights are enabled or not.

DANGER Do not remove the SSCNET III cable while power is supplied to the multiple CPU system or the servo amplifier. Do not look directly at light emitted from the tip of SSCNET III connectors or SSCNET III cables of the Motion CPU or the servo amplifier. Eye discomfort may be felt if exposed to the light.
(Reference: SSCNET III employs a Class 1 or equivalent light source as specified in JIS C 6802 and IEC60825-1 (domestic standards in Japan).)

DANGER Do not remove the SSCNET III cable while power is supplied to the controller. Do not look directly at light emitted from the tip of SSCNET III connectors or SSCNET III cables. Eye discomfort may be felt if exposed to the light.
(Reference: SSCNET III employs a Class 1 or equivalent light source as specified in JIS C 6802 and IEC60825-1 (domestic standards in Japan).)

⚠DANGER Attach the cap to the SSCNET III connector after disconnecting the SSCNET III cable. If the cap is not attached, dirt or dust may adhere to the connector pins, resulting in deterioration connector properties, and leading to malfunction.

⚠CAUTION Make sure there are no mistakes in the wiring. Connecting differently to the way specified in the manual can result in errors, such as the emergency stop not being released. In order to prevent errors occurring, please be sure to check that all functions (such as the teaching box emergency stop, customer emergency stop, and door switch) are working properly after the wiring setup is completed.

⚠CAUTION Use the network equipments (personal computer, USB hub, LAN hub, etc) confirmed by manufacturer. The thing unsuitable for the FA environment (related with conformity, temperature or noise) exists in the equipments connected to USB. When using network equipment, measures against the noise, such as measures against EMI and the addition of the ferrite core, may be necessary. Please fully confirm the operation by customer. Guarantee and maintenance of the equipment on the market (usual office automation equipment) cannot be performed.

Revision history

Date of print	Specifications No.	Details of revisions
2015-03-20	BFP-A3382	First print

■Preface

Thank you very much for purchasing Mitsubishi Electric Industrial Robot.
High speed and accuracy tracking function allows robots to follow workpiece on a conveyer with high speed and accuracy, line up and process the workpieces without having to stop the conveyer.

Please be sure to read this manual carefully and understand the contents thoroughly before starting to use the equipment in order to make full use of high speed and accuracy tracking function.
Within this manual, we have tried to describe all ways in which the equipment can be handled, including non-standard operations, to the greatest extent possible. Please avoid handling the equipment in any way not described in this manual.

Note that this manual is written for the following software version.

CR750-Q/CR751-Q series : Ver. R6 or later
CR750-D/CR751-D series : Ver. S6 or later

◆ When robot controller before Ver. R6/S6 is used

Please refer to “Tracking Function Instruction Manual” (BFP-A8664).

◆ When not the straight conveyer, but the circular arc conveyer or turntable is used

Please refer to “Circular Arc Tracking Function Instruction Manual” (BFP-A3380).

[Contents]

1. Overview

1.1. What is high Speed and Accuracy Tracking Function?

High speed and accuracy tracking function allows a robot to follow workpieces moving on a conveyer with high speed and accuracy. With this function, it becomes possible to transport, line up and process workpieces without having to stop the conveyer. It also eliminates the need for mechanical fixtures and so forth required to fix workpiece positions.

The features of this function are described below.

1) It is possible to follow lined-up workpieces moving on a conveyer while working on them (conveyer tracking making use of photo electronic sensors).
2) It is possible to follow workpieces that are not in a line moving on a conveyer while working on them, even in the case of different types of workpieces (vision tracking combined with vision sensors).
3) It is possible to follow changes of movement speed due to automatic calculation of conveyer movement speed.
4) It increases in the conveyance ability about 15 % (The ratio of our company) compared with the conventional tracking function.
5) Tracking function can be easily achieved by using Mitsubishi's robot command MELFA-BASIC V.
6) System construction is made easy by use of sample programs.

1.2. System that can achieve

With high speed and accuracy tracking function of CR750-Q/CR751-Q series, CR750-D/CR751-D series, the example of the system that can be achieved is shown as following.

Table 1-1 Example of system that can be achieved by high speed and accuracy tracking function

No.	CR750-Q CR751-Q	CR750-D CR751-D	Example of the system
1	●	●	When a robot picks the workpieces moving on a conveyer, it is tracking.（transportation）
2	●	●	When a robot places workpieces which taken out from the pallet to a conveyer, it is tracking (transportation). It is also possible to hang workpieces on S character hook that moves the above of the robot.
3	●	●	A robot decorates (processing) the workpieces moving on a conveyer while tracking.
4	●	●	A robot attaches the parts (assembling) with the workpieces moving on a conveyer while tracking.
5	●	●	A robot has the vision sensor (hand eye) and it checks the workpieces moving on a conveyer. (inspection) It also can check and push the button while tracking, not the vision sensor.
6	●	●	When a robot picks the workpieces moving on a conveyer A, the tracking is done and a robot places the workpieces while tracking to marking on a conveyer B.
7	●	●	The tracking is done with an encoder of line driver (differential motion) output type.
8	●	(●) Note1)	The tracking is done with an encoder of voltage output/open collector type.
9	●	-	In case of multi CPU system, it makes possible to add max 9 pcs Q173DPX units（3 units per 1 CPU）. However, in each CPU, only the two channels can be used at the 3rd set of Q173DPX units.

Note1) This system requires the Encoder distribution unit. Please refer to the Encoder Distribution Unit Manual (BFP-A3300) for details.

Tracking is primarily intended for applications such as the following.

(1) Transfer of processed food pallets

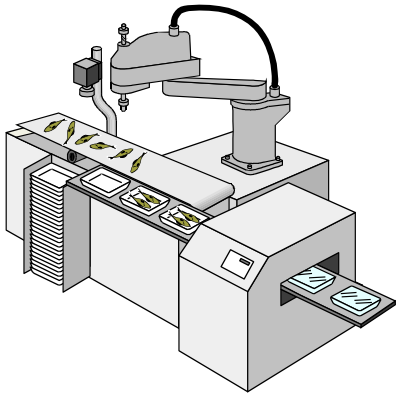


Figure 1-1 Example of Processed Food Pallet Transfer

(2) Lining up parts

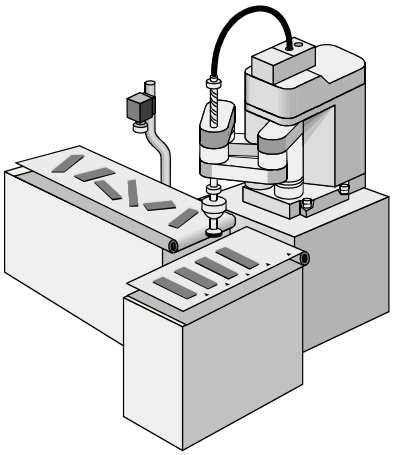


Figure 1-2 Example of Parts Lineup

(3) Assembly of small electrical products

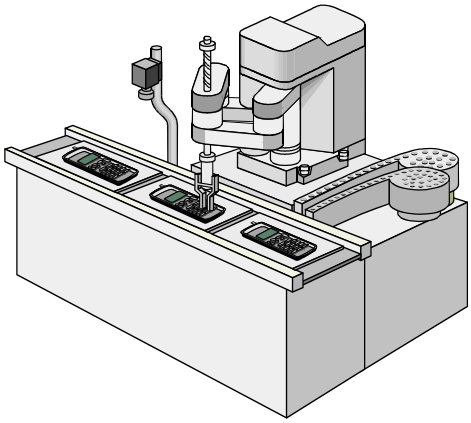


Figure 1-3 Example of Small Electrical Products Assembly

1.3. The terminology explanation

Table 1-2 The terminology explanation for high speed and accuracy tracking

Term	Explanation
Q type	CR750-Q/CR751-Q series robot controller.
D type	CR750-D/CR751-D series robot controller.
High speed and accuracy tracking function	The tracking function allows a robot to follow workpieces moving on a conveyer with high speed and accuracy. With this function, it becomes possible to transport line up and process workpieces without having to stop the conveyer.
Conveyer tracking	The conveyer tracking allows a robot to follow workpieces lining up on a conveyer. With this function, it becomes possible to transport, process workpieces.
Vision tracking	The vision tracking allows a robot to follow workpieces not lining up on a conveyer. With this function, it becomes possible to transport line up and process workpieces.
Q173DPX unit	Q173DRX unit is manual pulser input unit for motion controller. At Q series CPU, it is used as intelligent function unit (occupation 32 points） Each encoder figure can be got by connection with 1 pc the manual pulser machine（MR-HDP01）or 3pcs the incremental encoder.
Physical encoder number	Physical encoder numbers a number of the encoder physically allocated according to a certain rule. In the CR750-Q/CR751-Q series, the number is allocated by arranging the encoder connected with Q173DPX unit. The encoder which connected with CH1 of the Q173DPX unit specified for parameter “ENC UNIT1” is the first, the encoder which connected with CH2 is the second and with CH3 is the third. It becomes from 4 to 6 for the Q173DPX unit specified for parameter”ENCUNIT2”. It becomes from 7 to 8 for the Q173DPX unit specified for parameter”ENCUNIT3”. Note) The 3rd set of Q173DPX units can use only the two channels.
Logical encoder number	The physical encoder number change to the logical encoder number by parameter “EXTENC”. The purpose of this is to change freely number by the parameter for the encoder physically arranged. This logical encoder number is used with the instruction and the state variable of the robot program.
TREN signal	Tracking enable signal
SKIP input	SKIP input is an input for receiving signal from vision sensor.

2. System Configuration

2.1. Components

2.1.1. Robot controller enclosure products

The product structure of the tracking functional relation enclosed by the robot controller is shown in the Table 2-1 List of Configuration in the tracking functional-related product.

Table 2-1 List of Configuration in the tracking functional-related product

Product	Model name	Remark
High speed and accuracy Tracking Function INSTRUCTION MANUAL	BFP-A3382	It is downloades by Web.
Sample program	—	Please refer to “7 Installation of a sample robot program” for the sample robot program.

2.1.2. Devices Provided by Customers

When configuring the system, the customers must have certain other devices in addition to this product. The table below shows the minimum list of required devices. Note that different devices are required depending on whether conveyer tracking or vision tracking is used. Please refer to “Table 2-2 List of Devices Provided by Customers (Conveyer Tracking)” and “Table 2-3 List of Devices Provided by Customers (Vision Tracking)” for further details.

Table 2-2 List of Devices Provided by Customers (Conveyer Tracking)

Target type		Name of devices to be provided by customers	Model	Quantity	Remark
Q	D				
●	●	Hand	—	1	
●	●	Hand sensor	—	(1)	Used to confirm that workpieces are gripped correctly. Provide as necessary.
●	●	Solenoid valve set	See the Remark column	(1)	Different models are used depending on the robot used. Check the robot version and provide as necessary.
●	●	Hand input cable			
●	●	Calibration jig	—	(1)	This is a jig with a sharp tip that is attached to the mechanical interface of the robot arm and used for calibration tasks. It is recommended to use the jig if high precision is required.
●		Manual pulser input unit	Q173DPX	1	Manual pulser input unit for motion controller
	●	Parallel I/O interface	2D-TZ368/ 2D-TZ378	1	Used to confirm the input of the photoelectronic sensor. [*]In the case of CR750-Q/CR751-Q, This interface and unit are unnecessary to input to the TREN signal of the Q173DPX unit.
●	●	Conveyer	—	1	
●		Encoder	[Confirmed operation product] Omron encoder E6B2-CWZ1X -1000/ E6B2-CWZ1X -2000	1	Voltage output/open collector type Line driver output

Target type Q	Target type D	Name of devices to be provided by customers	Model	Quantity	Remark
	●	Encoder	[Confirmed operation product] Omron encoder E6B2-CWZ1X -1000/ E6B2-CWZ1X -2000	1	Line driver output
●		Encoder cable	2D-CBL05/ 2D-CBL15	1	
	●	Encoder cable	—	1	Shielded twisted pair cable
	●	5V power supply	—	1	+5V DC (±10%):For Encoder [*]In the case of Q type, the Q173DPX unit supplies 5V power supply to the encoder.
●	●	Photo electronic sensor	—	1	Used to detect a workpiece position
●	●	24V power supply	—	1	+24 VDC (±10%) : For the photo electronic sensor
●	●	RT ToolBox2	3D-11C-WINE 3D-12C-WINE	1	Please refer to the instruction manual of RT ToolBox2 for the details of the personal computer specifications.

Table 2-3 List of Devices Provided by Customers (Vision Tracking)

Target type Q	Target type D	Name of devices to be provided by customers	Model	Quantity	Remark
●	●	Hand	—	1	
●	●	Hand sensor	—	(1)	Used to confirm that workpieces are gripped correctly. Provide as necessary.
●	●	Solenoid valve set	See the Remark column	(1)	Different models are used depending on the robot used. Check the robot version and provide as necessary.
●	●	Hand input cable			
●	●	Calibration jig	—	(1)	This is a jig with a sharp tip that is attached to the mechanical interface of the robot arm and used for calibration tasks. It is recommended to use the jig if high precision is required.
●		Manual pulser input unit	Q173DPX	1	Manual pulser input unit for motion controller
	●	Parallel I/O interface	2D-TZ368/ 2D-TZ378	1	Used to confirm the input of the photoelectronic sensor. [*]In the case of CR750-Q/CR751-Q, This interface and unit are unnecessary to input to the TREN signal of the Q173DPX unit.
●	●	Conveyer	—	1	

Target type Q	Target type D	Name of devices to be provided by customers	Model	Quantity	Remark
●		Encoder	[Confirmed operation product] Omron encoder E6B2-CWZ1X -1000/ E6B2-CWZ1X -2000	1	Voltage output/open collector type Line driver output
	●	Encoder	[Confirmed operation product] Omron encoder E6B2-CWZ1X -1000/ E6B2-CWZ1X -2000	1	Line driver output
●		Encoder cable	2D-CBL05/ 2D-CBL15	1	
	●	Encoder cable	—	1	Shielded twisted pair cable
	●	5V power supply	—	1	+5V DC (±10%):For Encoder [*]In the case of Q type, the Q173DPX unit supplies 5V power supply to the encoder.
●		Encoder distribution unit	2F-YZ581	(1)	The Encoder distribution unit is required when two or more manual pulser input units are connected to the one encoder. Provide this unit as necessary. Refer to the Encoder Distribution Unit Manual (BFP-A3300) for details.
	●	Encoder distribution unit	2F-YZ581	(1)	The Encoder distribution unit is required when two or more robot controllers are connected to the one encoder. Provide this unit as necessary. If the Encoder distribution unit is used, a 5V power source for the encoder is not necessary. Refer to the Encoder Distribution Unit Manual (BFP-A3300) for details.
●	●	In-Sight 5000 series In-Sight Micro In-Sight EZ	—	1	COGNEX Vision sensor
●	●	Lens	—	1	C-mount lens
●	●	Breakout cable	—	1	This cable is used by high accuracy tracking.
●	●	Lighting installation	—	(1)	Provide as necessary
●	●	Hub	—	1	
●	●	24V power supply	—	1	+24 VDC (±10%) : For the Vision sensor
●	●	RT ToolBox2	3D-11C-WINE 3D-12C-WINE	1	Please refer to the instruction manual of RT ToolBox2 for the details of the personal computer specifications.

2. 2. Example of System Configuration

The following figure shows examples of conveyer tracking systems and vision tracking systems.

2. 2. 1. Configuration Example of Conveyer Tracking Systems [Q type]

The following figure shows a configuration example of a system that recognizes lined-up workpieces on a conveyer passing a photo electronic sensor and follows the workpieces.

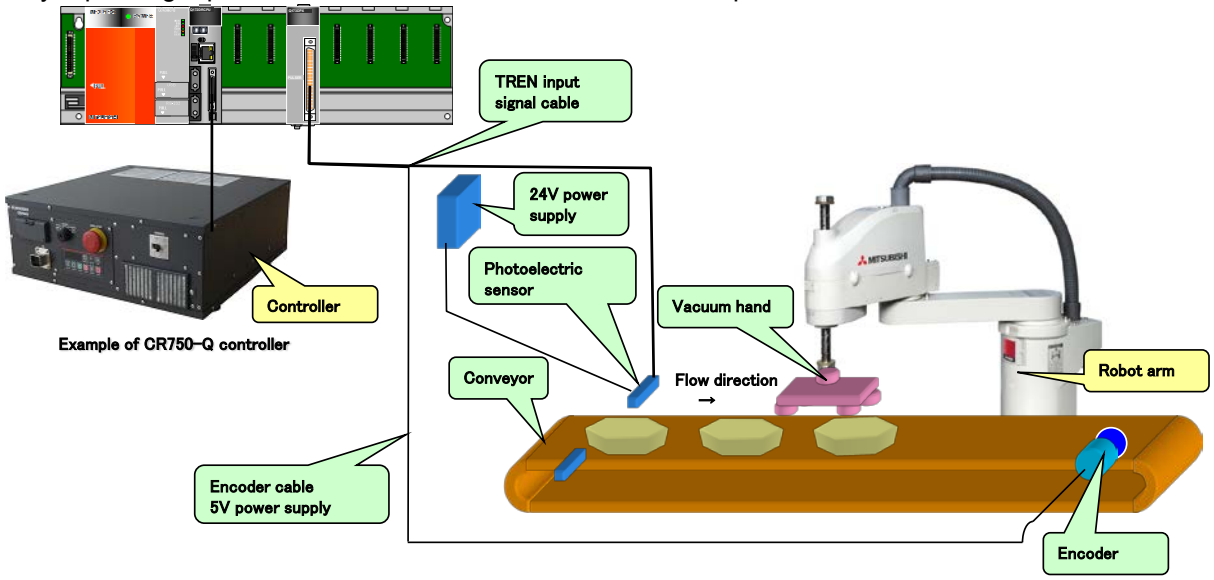


Figure2-1 Configuration Example of Conveyer Tracking Systems [Q type]

2. 2. 2. Configuration Example of Vision Tracking Systems [Q type]

The following figure shows a configuration example of a system that recognizes positions of workpieces that are not lined up on a conveyer with a vision sensor and follows the workpieces.

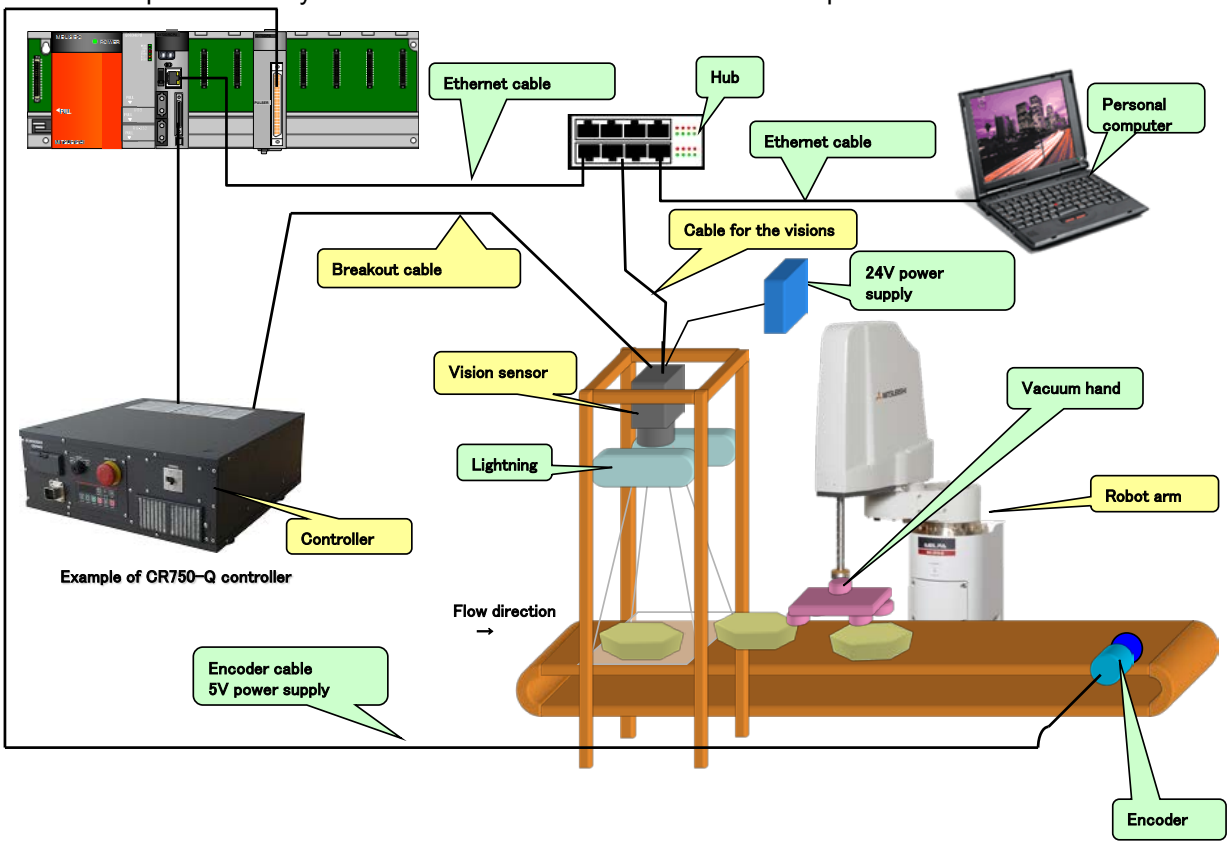


Figure2-2 Configuration Example of Vision Tracking Systems [Q type]

2. 2. 3. Configuration Example of Conveyer Tracking Systems [D type]

The following figure shows a configuration example of a system that recognizes lined-up workpieces on a conveyer passing a photo electronic sensor and follows the workpieces.

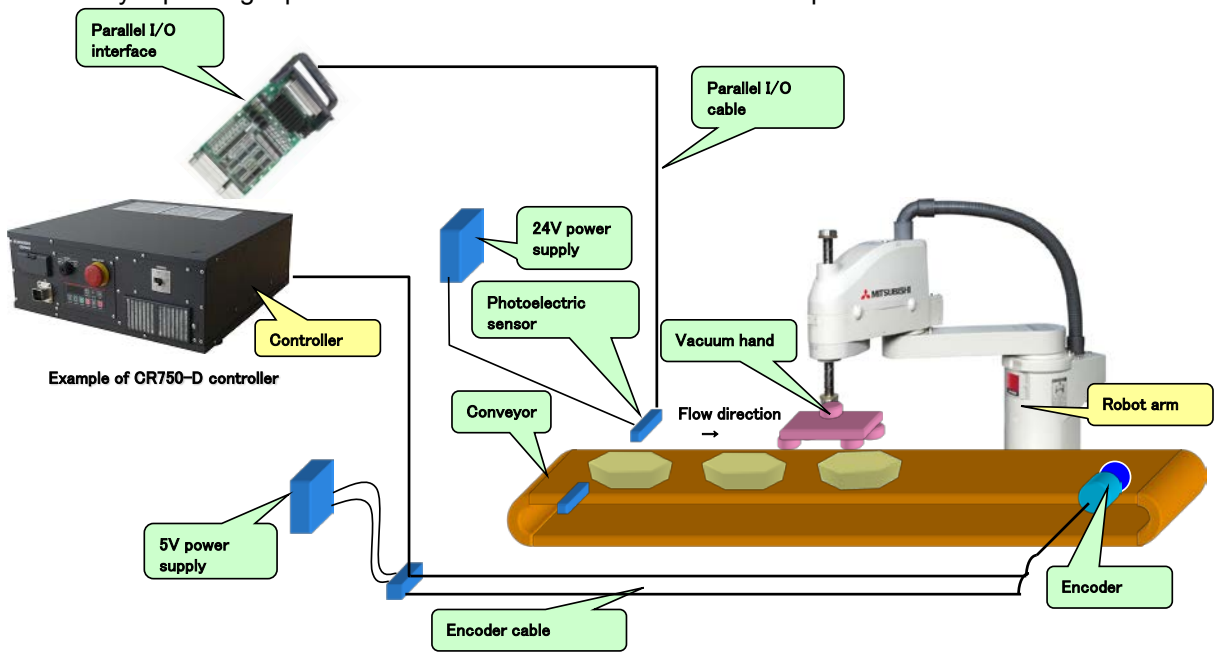


Figure2-3 Configuration Example of Conveyer Tracking Systems [D type]

2. 2. 4. Configuration Example of Vision Tracking Systems [D type]

The following figure shows a configuration example of a system that recognizes positions of workpieces that are not lined up on a conveyer with a vision sensor and follows the workpieces.

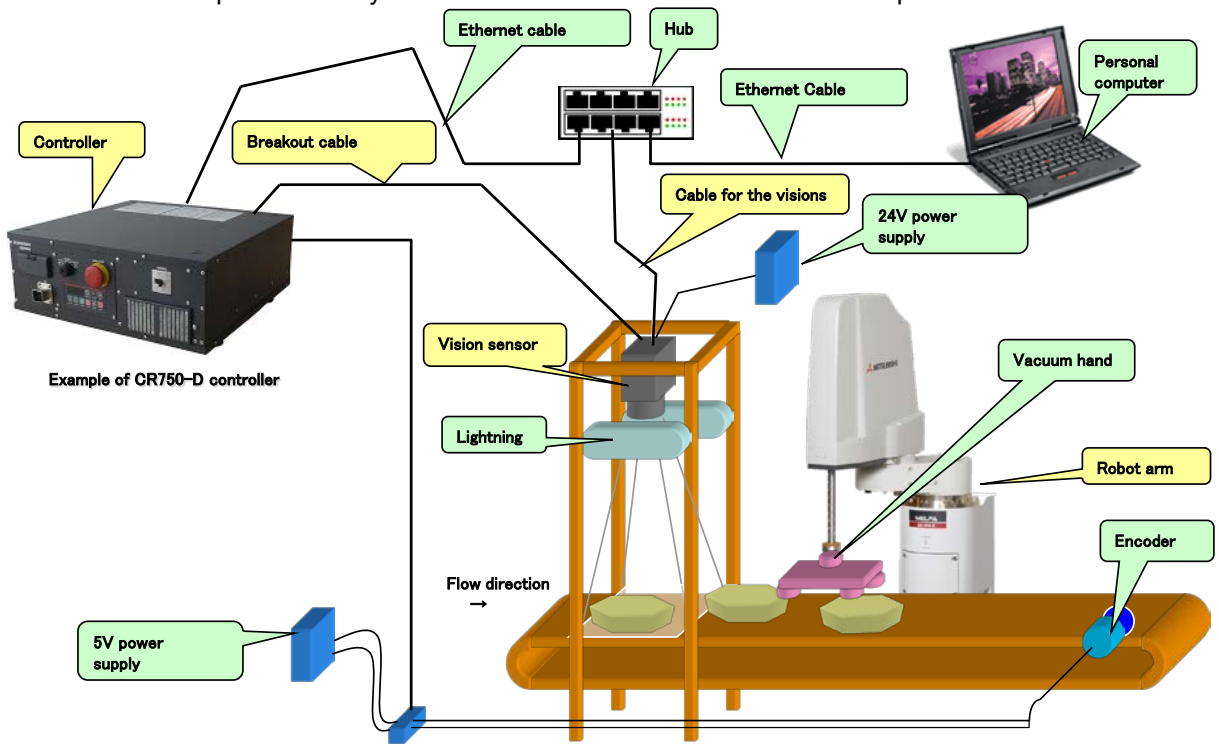


Figure2-4 Configuration Example of Vision Tracking Systems [D type]

3. Specification

3.1. High Speed and Accuracy Tracking Specifications

The table below shows the high speed and accuracy tracking specifications.
Please refer to “Standard Specifications Manual” for the specifications of the robot arm and controller to be used.

3.1.1. Q type

Table3-1 CR750-Q/CR751-Q Series Controller High Speed and Accuracy Tracking Function Specifications

Item		Specification
Supported robots (*1)		RH-FH-Q series / RV-F-Q series
Applicable robot controller		CR750-Q/CR751-Q series controller
Conveyer	Number of conveyer (*2)	Max 8pcs (in case 1pc encoder connect to 1 pc conveyer) Encoder 3 pcs / Q173DPX unit 1pc Q173DPX unit 3pcs / system
	Movement Speed(*3)	Possible to support up to 300mm/s (When the robot always transport the workpieces) Possible to support up to 500mm/s when the interval of workpiece is wide.
	Encoder	Voltage output/open collector type : A、B、Z (*4) Line driver output : A、$\overline{A}$、B、$\overline{B}$、Z、$\overline{Z}$ (*5) Resolution(pulse/rotation) : Up to 2000 (4000 and 8000 uncorrespond) Confirmed operation product : Omuron E6B2-CWZ1X-1000 E6B2-CWZ1X-2000
	Encoder cable	2D-CBL05(External I/O cable 5m) 2D-CBL15(External I/O cable 15m)
Encoder unit		Only Q173DPX unit
Photoelectronic Sensor(*6)		Used to detect workpieces positions in conveyer tracking. Output signal of sensor need to be connected to TREN terminal of Q173DPX unit. (Input signal number 810 to 817) And a momentary encoder value that the input enters is preserved in state variable "M_EncL".
Vision Sensor(*7)		COGNEX’s vision sensor. Output signal of sensor need to be connected to SKIP input terminal of CNUSR2.
Precision at handling position(*3)		Approximately ±1 mm (when the conveyer speed is approximately 300 mm/s) (Photoelectronic sensor recognition accuracy, robot repeatability accuracy and so on)

(*1) The sample program doesn't correspond to the RV-5 axis robot.
(*2) The encoder connected with the third channel of the Q173DPX unit specified for parameter "ENCUNIT3" cannot be used.
(*3) The specification values in the table should only be considered guidelines. The actual values depend on the specific operation environment, robot model, hand, Sensitivity of the sensor and other factors.
(*4) Voltage output/open collector type is an output circuit with two output transistors of NPN and PNP.
(*5) The line driver output is a data transmission circuit in accordance with RS-422A. It enables the long-distance transmission.
(*6) Please connect the output signal of a photoelectric sensor with the terminal TREN of the Q173DPX unit. This input can be confirmed, by the input signal 810th-817th.
(*7) Please connect the output signal of a breakout cable with SKIP input terminal of CNUSR2.

3.1.2. D type

Table3-2 CR750-D/CR751-D Series Controller High Speed and Accuracy Tracking Function Specifications

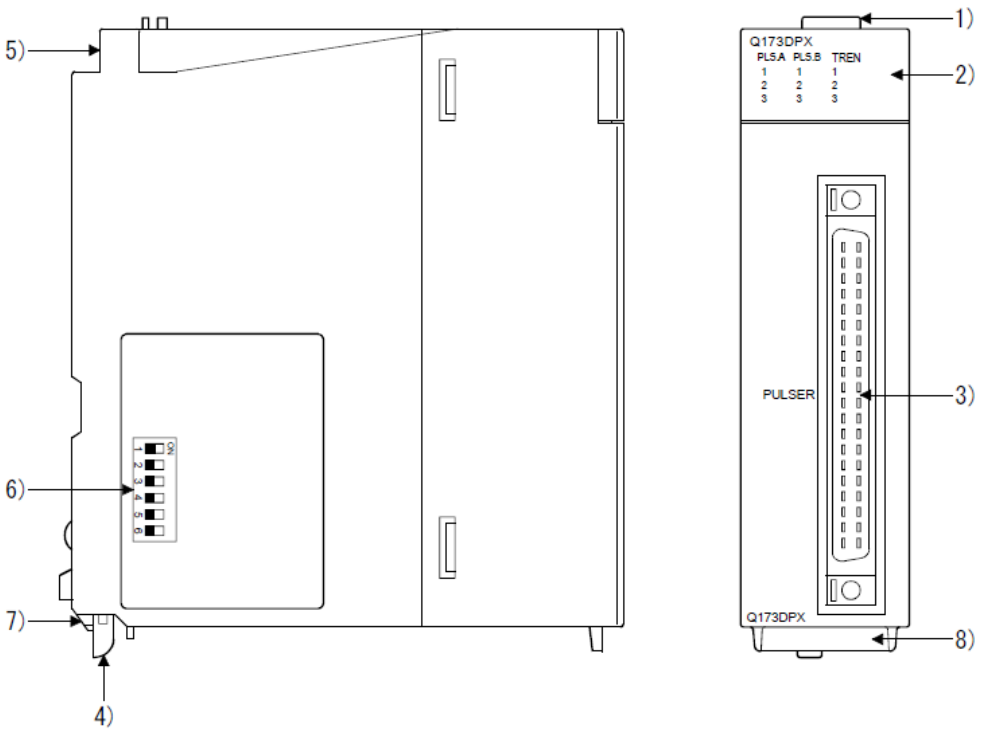
Item		Specification
Supported robots (*1)		RH-FH-D series / RV-F-D series
Applicable robot controller		CR750-D/CR751-D series controller
Conveyer	Number of conveyer	Max 2pcs (in case 1pc encoder connect to 1 pc conveyer) Encoder 2 pcs / system Possible to support up to two conveyers by robot controller standard constitution
	Movement Speed（*2）	Possible to support up to 300mm/s (When the robot always transport the workpieces) Possible to support up to 500mm/s when the interval of workpiece is wide.
	Encoder	Line driver output : A、$\overline{A}$、B、$\overline{B}$、Z、$\overline{Z}$ (*3) Resolution(pulse/rotation) : Up to 2000 (4000 and 8000 uncorrespond) Confirmed operation product : Omuron E6B2-CWZ1X-1000 E6B2-CWZ1X-2000 Maximum response frequency : 100 kHz
	Encoder cable	Shielded twisted-pair cable Outside dimension : Maximum phi6mm Conductor size: 24AWG (0.2 mm^2) Cable length: Up to 25 m
Encoder wiring		An encoder and the robot controller are accessible with one to one Encoder Distribution Unit
Photoelectronic Sensor(*4)		Used to detect workpieces positions in conveyer tracking. Output signal of sensor need to be connected to TREN terminal of Q173DPX unit. (Input signal number 810 to 817) And a momentary encoder value that the input enters is preserved in state variable "M_EncL".
Vision Sensor(*5)		COGNEX's vision sensor. Output signal of sensor need to be connected to SKIP input terminal of CNUSR2.
Precision at handling position（*3）		Approximately ±1 mm (when the conveyer speed is approximately 300 mm/s) (Photoelectronic sensor recognition accuracy, robot repeatability accuracy and so on)

(*1) The sample program doesn't correspond to the RV-5 axis robot.
(*2) The specification values in the table should only be considered guidelines. The actual values depend on the specific operation environment, robot model, hand, Sensitivity of the sensor and other factors.
(*3) The line driver output is a data transmission circuit in accordance with RS-422A. It enables the long-distance transmission.
(*4) Please input the output signal of the photoelectric sensor into the general-purpose input signal (voluntarily) of the robot controller.
(*5) Please connect the output signal of a breakout cable with SKIP input terminal of CNUSR2.

3. 2. Q173DPX(manual pulser input)unit specification

Add Q173DPX unit into PLC base unit (Q3□DB) when the customer use Q type high speed and accuracy tracking function.
Please refer to "Q173DCPU/Q172DCPU user's manual" about details of this unit.

(1) External and name of Q173DPX unit.



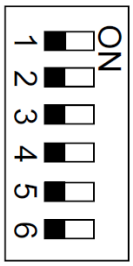
No.	Name	Application
1)	Module fixing hook	Hook used to fix the module to the base unit. (Single-motion installation)
2)	Mode judging LED	Display the input status from the external equipment. LED: PLS.A 1 to 3, PLS.B 1 to 3 — Display for input signal status of manual pulse generator/incremental synchronous encoder phases A, B TREN 1 to 3 — Display for signal status of tracking enable. The manual pulse generator/incremental synchronous encoder phases A, B and tracking enable signal does not turn ON without setting Q173DPX in the system setting.
3)	PULSER connector	Input connector of the Manual pulse generator/Incremental synchronous encoder.
4)	Module mounting lever	Used to install the module to the base unit.
5)	Module fixing screw hole	Hole for the screw used to fix to the base unit (M3×12 screw : Purchase from the other supplier)

Figure3-1 Externals of Q173DPX unit

(2) Dip switch

By setting the dip switch, the condition of the tracking enable signal is decided.

Table3-3 Item of dip switch

<table>
<tr><th>No.</th><th>Name</th><th colspan="2">Application</th></tr>
<tr><td rowspan="6">6)</td><td rowspan="6">Dip switches (Note-1)
 

(Factory default in OFF position)</td><td>Dip switch 1</td><td rowspan="2">Detection setting of TREN1 signal
SW1 SW2
OFF OFF
ON ON
ON OFF } TREN is detected at leading edge of TREN signal.
OFF ON TREN is detected at trailing edge of TREN signal.</td></tr>
<tr><td>Dip switch 2</td></tr>
<tr><td>Dip switch 3</td><td rowspan="2">Detection setting of TREN2 signal
SW3 SW4
OFF OFF
ON ON
ON OFF } TREN is detected at leading edge of TREN signal.
OFF ON TREN is detected at trailing edge of TREN signal.</td></tr>
<tr><td>Dip switch 4</td></tr>
<tr><td>Dip switch 5</td><td rowspan="2">Detection setting of TREN3 signal
SW5 SW6
OFF OFF
ON ON
ON OFF } TREN is detected at leading edge of TREN signal.
OFF ON TREN is detected at trailing edge of TREN signal.</td></tr>
<tr><td>Dip switch 6</td></tr>
<tr><td>7)</td><td>Module fixing projection</td><td colspan="2">Projection used to fix to the base unit.</td></tr>
<tr><td>8)</td><td>Serial number display</td><td colspan="2">Display the serial number described on the rating plate.</td></tr>
</table>

(Note-1) : The function is different according to the operating system software installed.

⚠CAUTION

- Before touching the DIP switches, always touch grounded metal, etc. to discharge static electricity from human body. Failure to do so may cause the module to fail or malfunction.
- Do not directly touch the module's conductive parts and electronic components. Touching them could cause an operation failure or give damage to the module.

(3) Specification of hardware

(a) Module specifications

Item	Specifications
Number of I/O occupying points	32 points(I/O allocation: Intelligent, 32 points)
Internal current consumption(5VDC)[A]	0.38
Exterior dimensions [mm(inch)]	98(H)×27.4(W)×90(D) (3.86(H)×1.08(W)×3.54(D))
Mass [kg]	0.15

(b) Tracking enable signal input

Item		Specifications
Number of input points		Tracking enable signal : 3 points
Input method		Sink/Source type
Isolation method		Photocoupler
Rated input voltage		12/24VDC
Rated input current		12VDC 2mA/24VDC 4mA
Operating voltage range		10.2 to 26.4VDC (12/24VDC +10/ -15%, ripple ratio 5% or less)
ON voltage/current		10VDC or more/2.0mA or more
OFF voltage/current		1.8VDC or less/0.18mA or less
Input resistance		Approx. 5.6kΩ
Response time	OFF to ON	0.4ms/0.6ms/1ms (CPU parameter setting, Default 0.4ms)
	ON to OFF	
Common terminal arrangement		1 point/common(Common contact: TREN.COM)
Indicates to display		ON indication(LED)

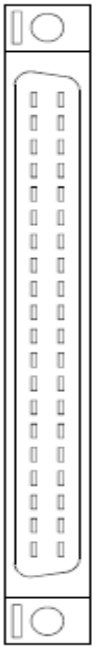
(Note): Functions are different depending on the operating system software installed.

(c) Manual pulse generator/Incremental synchronous encoder input

Item		Specifications
Number of modules		3/module
Voltage-output/ Open-collector type	High-voltage	3.0 to 5.25VDC
	Low-voltage	0 to 1.0VDC
Differential-output type (26LS31 or equivalent)	High-voltage	2.0 to 5.25VDC
	Low-voltage	0 to 0.8VDC
Input frequency		Up to 200kpps (After magnification by 4)
Applicable types		Voltage-output type/Open-collector type (5VDC), Recommended product: MR-HDP01, Differential-output type: (26LS31 or equivalent)
External connector type		40 pin connector
Applicable wire size		$0.3mm^2$
Applicable connector for the external connection		A6CON1 (Attachment) A6CON2, A6CON3, A6CON4 (Optional)
Cable length	Voltage-output/ Open-collector type	30m (98.43ft.) (Open-collector type: 10m (32.81ft.))
	Differential-output type	

(4) Wiring

The pin layout of the Q173DPX PULSER connecter viewed from the unit is shown below



PULSER connector

	Pin No.	Signal Name	Pin No.	Signal Name	
2)	B20	HB1	A20	HA1	2)
	B19	SG	A19	SG	
	B18	5V	A18	HPSEL1	1)
3)	B17	HA1N	A17	HA1P	3)
	B16	HB1N	A16	HB1P	
2)	B15	HB2	A15	HA2	2)
	B14	SG	A14	SG	
	B13	5V	A13	HPSEL2	1)
3)	B12	HA2N	A12	HA2P	3)
	B11	HB2N	A11	HB2P	
2)	B10	HB3	A10	HA3	2)
	B9	SG	A9	SG	
	B8	5V	A8	HPSEL3	1)
3)	B7	HA3N	A7	HA3P	3)
	B6	HB3N	A6	HB3P	
	B5	No connect	A5	No connect	
	B4	TREN1 −	A4	TREN1 +	
	B3	TREN2 −	A3	TREN2 +	
	B2	TREN3 −	A2	TREN3 +	
4)	B1	FG	A1	FG	4)

Applicable connector model name

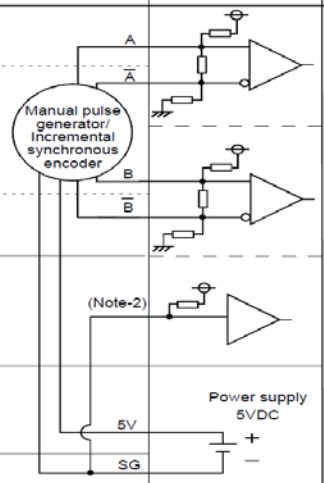
A6CON1 type soldering type connector
FCN-361J040-AU connector (FUJITSU COMPONENT LIMITED)
FCN-360C040-B connector cover
} (Attachment)

A6CON2 type Crimp-contact type connector
A6CON3 type Pressure-displacement type connector
A6CON4 type soldering type connector
} (Optional)

Figure3-2 Pin assignment of the PULSER connector

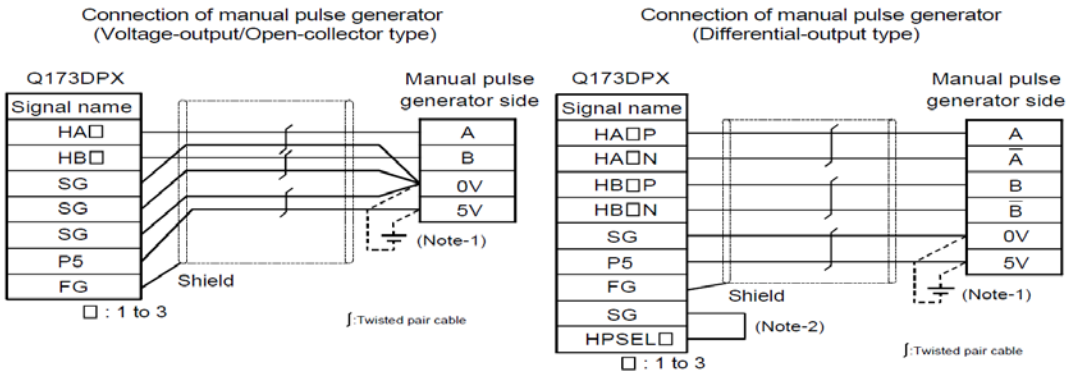
Interface between PULSER connecter and manual pulse generator (Differential-output type)/ Incremental synchronous encoder

Interface between Manual pulse generator (Differential-output type)/ Incremental synchronous encoder

Input or Output	Signal name		Pin No. PULSER connector Differential-output type 1	2	3	Wiring example	Internal circuit	Specification	Description
Input	Manual pulse generator, phase A	A+ HA□P	A17	A12	A7			• Rated input voltage 5.5VDC or less • HIGH level 2.0 to 5.25VDC • LOW level 0.8VDC or less • 26LS31 or equivalent	For connection manual pulse generator Phases A, B • Pulse width • Leading edge, Trailing edge time ••• 1μs or less. • Phase difference (1) Positioning address increases if Phase A leads Phase B. (2) Positioning address decreases if Phase B leads Phase A.
		A− HA□N	B17	B12	B7				
	Manual pulse generator, phase B	B+ HB□P	A16	A11	A6				
		B− HB□N	B16	B11	B6				
	Select type signal HPSEL□		A18	A13	A8				
Power supply	P5 (Note-1)		B18	B13	B8				
	SG		A19 B19	A14 B14	A9 B9				

(Note-1) : The 5V(P5)DC power supply from the Q173DPX must not be connected if a separated power supply is used as the Manual pulse generator/Incremental synchronous encoder power supply. Use a 5V stabilized power supply as a separated power supply. Any other power supply may cause a failure.

(Note-2) : Connect HPSEL□ to the SG terminal if the manual pulse generator (differential-output type) /incremental synchronous encoder is used.



(Note-1) : The 5V(P5)DC power supply from the Q173DPX must not be connected if a separated power supply is used as the Manual pulse generator/Incremental synchronous encoder power supply.
Use a 5V stabilized power supply as a separated power supply. Any other power supply may cause a failure.

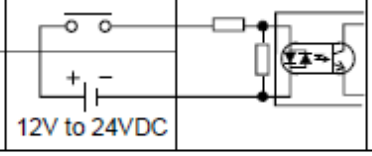
(Note-2) : Connect HPSEL□ to the SG terminal if the manual pulse generator (differential-output type)/incremental synchronous encoder is used.

Figure3-3 Wiring connection with rotary encoder

As above image, because DC5V voltage is output from Q173DPX unit, it makes possible to supply 5V from Q173DPX unit to rotary encoder. When 24V encoder type of power supply is used, it makes possible to use 24V output from PLC power unit.

The interface between tracking enable signal is shown follow.
This signal is used for input signal when the photoelectronic sensor is used to find workpieces so please connect output signal of photoelectronic sensor.

Interface between tracking enable signal

Input or Output	Signal name		Pin No. PULSER connector 1	2	3	Wiring example	Internal circuit	Specification	Description
Input	Tracking enable	TREN□+	A4	A3	A2				Tracking enable signal input.
		TREN□–	B4	B3	B2				

(Note) : As for the connection to tracking enable (TREN□+, TREN□–), both "+" and "–" are possible.

Figure3-4 Connected composition of tracking enable signal

⚠CAUTION

- If a separate power supply is used as the manual pulse generator/incremental synchronous encoder power supply, use a 5V stabilized power supply. Any other power supply may cause a failure.
- Always wire the cables when power is off. Not doing so may damage the circuit of modules.
- Wire the cable correctly. Wrong wiring may damage the internal circuit.

The connection robot system with Q173DPX unit is shown as follow.

Table3-4 Spec list of Q173DPX in robot system

Item	Spec and Remark
Encoder	Incremental synchronous encoder 3pcs
Tracking input points	3points Three points can be input to ± TREN1-3 in the pin assignment of the unit. When the input of a photoelectric sensor is put, this input is used.
Slot that can be connected	Connection with the base unit Possible to install I/O slot since 3 （Impossible to install CPU slot or I/O slot 0 to 2） Connection with additional base unit Possible to install all slots.
Robot CPU unit that can be managed	Q173DPX unit 3pcs
Robot CPU encoder that can be managed	Max 8pcs Impossible to use the third channel of the third Q173DPX unit. And impossible to use the encoder connected with the third channel of the unit specified for parameter「ENCUNIT3」.

4. Operation Procedure

This chapter explains the operation procedure for constructing a high speed and accuracy tracking system.

Start of operation

↓

1. Connection of Equipment ····· Refer to “Chapter 5.”
   [Q type]
   Chapter 5 explains Q173DPX (manual pulser input) unit preparation and the connection with the encoder.
   [D type]
   Chapter 5 explains setting of the option card and the connection with the encoder.

↓

2. Parameter Setting ····· Refer to “Chapter 6.”
   Chapter 6 explains assignment of signals for external equipment to control a robot and parameter about the tracking and parameter about movement such as the length of the tool.

↓

3. Installation of a sample robot program ····· Refer to “Chapter 7.”
   Chapter 7 explains functions related to sample programs.

↓

4. Calibration of Conveyer and Robot Coordinate Systems (“A1” program)····· Refer to “Chapter 8.”
   Chapter 8 explains how to calculate the amount of robot movement per encoder pulse.

↓

5. Calibration of Vision Coordinate and Robot Coordinate Systems (“B1” program)· Refer to “Chapter 9.”
   Chapter 9 explains how to display the position of a workpiece recognized by the vision sensor in the robot coordinate system.
   ∗Only Vision Tracking

↓

6. Workpiece Recognition and Teaching (“C1” program) ····· Refer to “Chapter 10.”
   [Conveyer Tracking]
   Chapter 10 explains how to calculate the relationship between the position of a workpiece recognized by the photoelectronic sensor and the position at which the robot grabs the workpiece.
   [Vision Tracking]
   Chapter 10 explains how to calculate the relationship between the position of a workpiece recognized by the vision sensor and the position at which the robot grabs the workpiece.

↓

7. Teaching and Setting of Adjustment Variables (“1” program) ····· Refer to “Chapter 11.”
   Chapter 11 explains adjustment of the conveyance route in the automatic driving and a change of the adsorption time.

↓

End of operation

8. Maintenance of robot program ····· Refer to “Chapter 13.”

9. Troubleshooting ····· Refer to “Chapter 14.”

5. Connection of Equipment

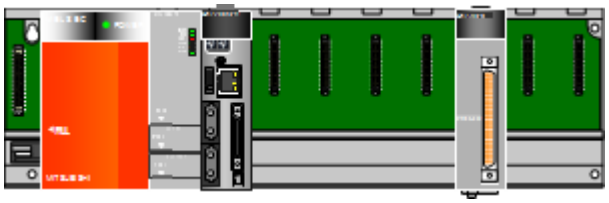
This section explains how to connect each of the prepared pieces of equipment.
Prepare equipment by referring to “Table 2-2” and “Table 2-3”.

5.1. Connection of Equipment [Q type]

The connection with each equipments is explained as follow.

5.1.1. Connection of Unit

Q173DPX unit is connected to base unit（Q3□DB）or Q6□B increase base unit.
For example, attach Q173DPX unit to I/O5 slot as follows.



5.1.2. Connection with encoder for conveyer and encoder cable

E6B2-CWZ1X (made by Omron) is used, and the wiring for the encoder and the encoder cable for the conveyer is shown in “Figure5-2 The encoder for the conveyer and the wiring diagram of the encoder cable [Q type]”.
The encoder for the conveyer up to 3 pcs can be connected per Q173DP unit 1pc. The signal cables needed in case of the connection are power supply (+,-) and encoder A,B,Z each +,-, total 8 cables. Please refer to the manual of the encoder, please connect signal cable correctly. Also please ground shield line (SLD).

The wiring example by the thing is shown below.
(Please note that the connector shape is different depending on the controller)

Figure5-1 Wiring example from an encoder to aunit [Q type]

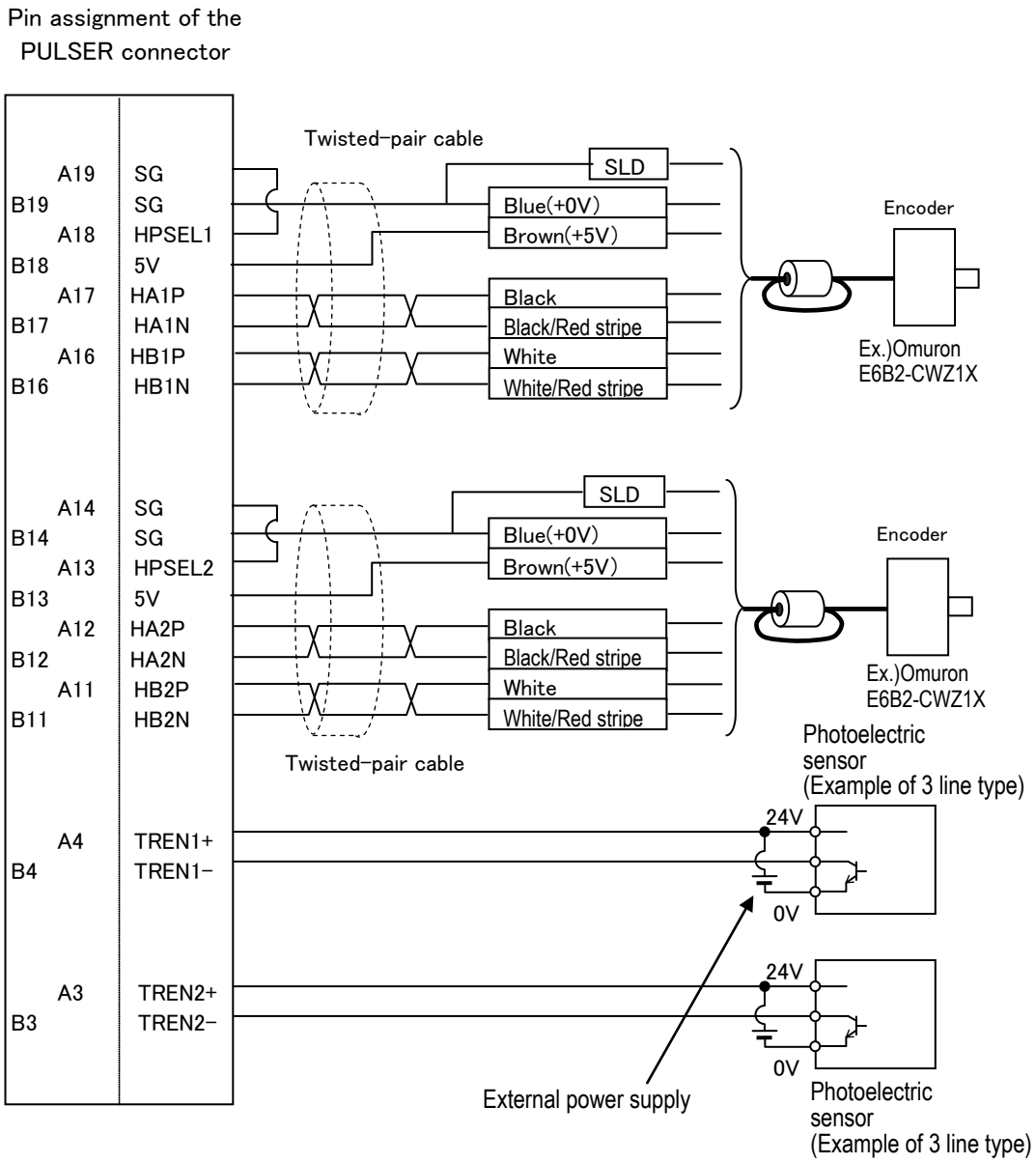


Figure5-2 The encoder for the conveyer and the wiring diagram of the encoder cable [Q type]

*Please refer to "Figure3-2 Pin assignment of the PULSER connector" with the pin crack of the PULSER connector that arrives at the unit.

5.1.3. Connection of Photoelectronic Sensor

If a photoelectronic sensor is used for detection of workpieces, connect the output signal of the photoelectronic sensor to a tracking enable signal of the Q173DPX unit.
In this section, the connection example to 1 channel (A4, B4) is shown below.

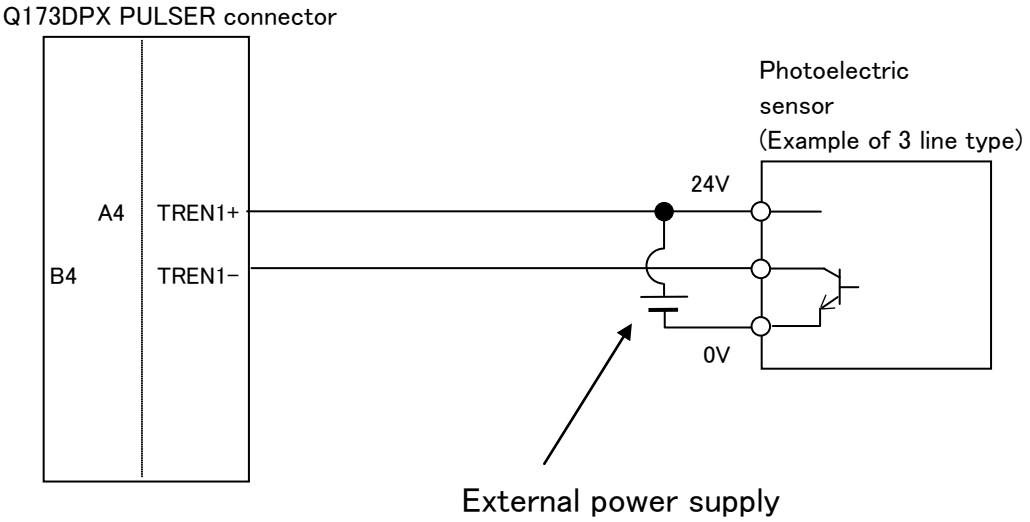


Figure5-3 Photoelectronic Sensor Connection Example (6th General Input Signal is Used) [Q type]

The tracking enable signal is connected to the robot input signal as follows.

Table5-1 List with signal crack of tracking enable signal (TREN)

Encoder physics number	Connection channel Q type	Robot Input signal number
1	1st channel of Parameter ENCUNIT1	810
2	2nd channel	811
3	3rd channel	812
4	1st channel of Parameter ENCUNIT2	813
5	2nd channel	814
6	3rd channel	815
7	1st channel of Parameter ENCUNIT3	816
8	2nd channel	817

5.1.4. Connection of Vision Sensor

If a vision sensor is used for detection of workpieces, connect “HS OUT 0” and “GROUND (Micro series: HS COMMON)” of the vision sensor to SKIP input terminal of CNUSR connector.
In this section, the connection example to 2 channel (10, 35) is shown below.
Refer to a manual of the vision sensor which you use about specification of a breakout cable.

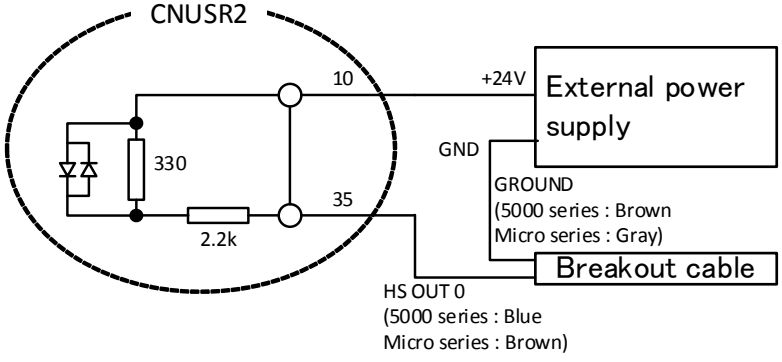


Figure5-4 Vision Sensor Connection Example (SKIP “2” Input Signal is Used) [Q type]

5. 2. Connection of Equipment [D type]

The connection with each equipments is explained as follow.

5. 2. 1. Connection with encoder for conveyer and encoder cable

E6B2-CWZ1X (made by Omron) is used, and the wiring for the encoder and the encoder cable for the conveyer is shown in “Figure5-6 The encoder and the wiring diagram of the encoder cable (CR750-D series controller)” and “Figure5-8 The encoder and the wiring diagram of the encoder cable (CR751-D series controller)”.
The a maximum of two encoders for the conveyors are connectable as standard specification. A total of 8 signal wires are required for the connection for the power supply (+ and - terminals) and the + and - terminals of the differential encoders' A, B and Z phases. Refer to the instruction manual of the encoders to be used and connect the signal wires correctly. Note that shielded wires (SLD) should be connected to the ground of the controller and system.

CAUTION

Be sure to mount ferrite cores on all encoder cables.

Be sure to mount the ferrite cores on the encoder cables at a position near the robot controller. If ferrite cores are not mounted, the robot may malfunction due to the influence of noise.

CAUTION

There is one robot controller connectable with the one encoder.

If two or more robot controllers are connected to the one encoder, the waveform of the encoder falls and the exact encoder value may be unable to be acquired. If you want to connect two or more robot controller to the one encoder, the Encoder distribution unit (model: 2F-YZ581) is required. Refer to the Encoder Distribution Unit Manual (BFP-A3300) for details.

(1)CR750-D series controller

The wiring example by the thing is shown below.
(Please note that the connector shape is different depending on the controller)

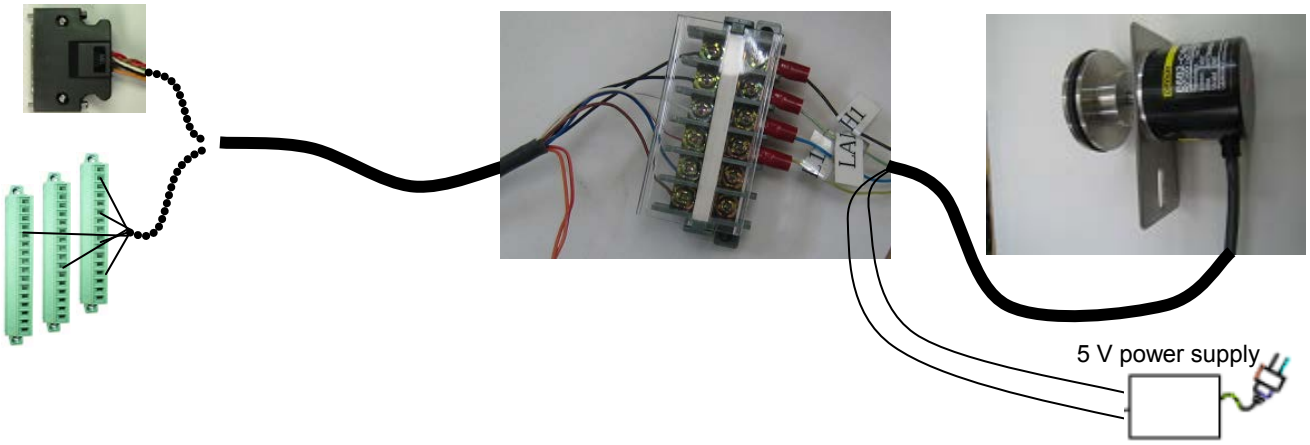


Figure5-5 Wiring example (CR570-D series controller)

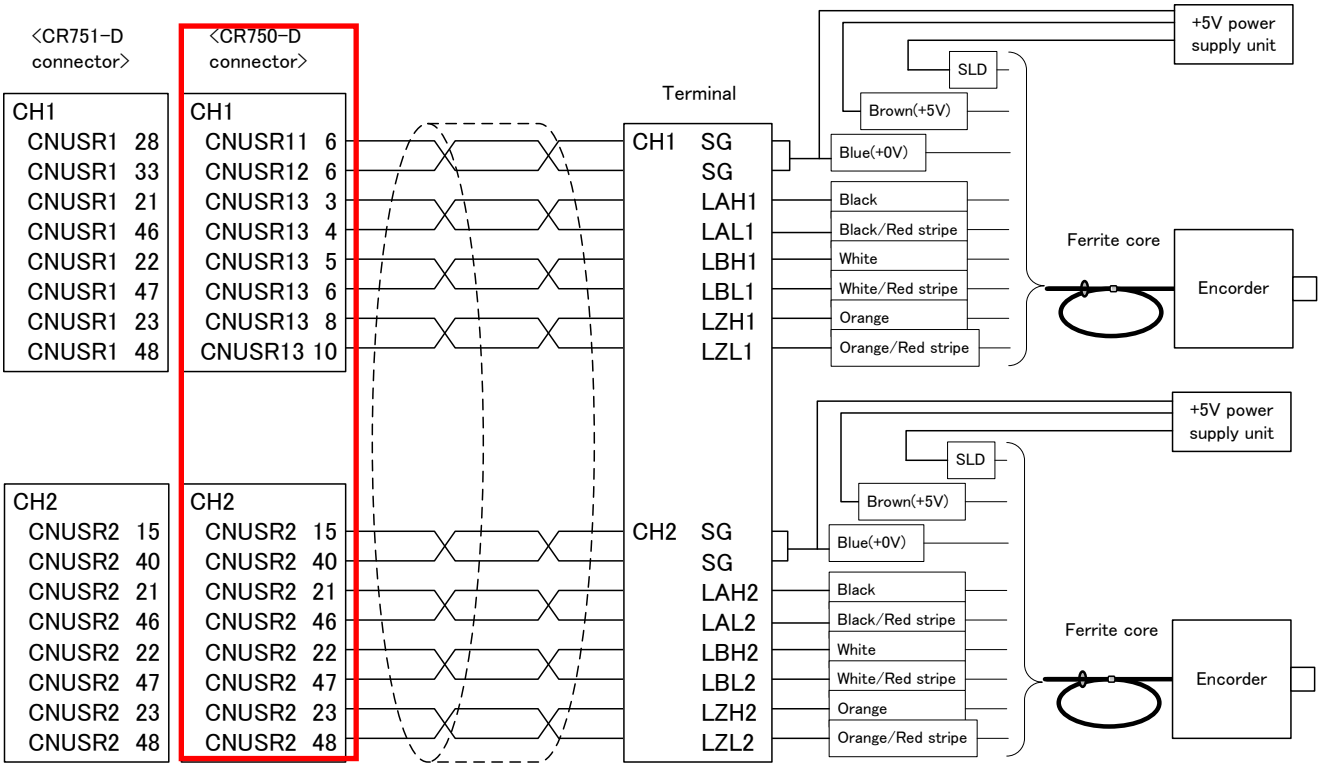


Figure5-6 The encoder and the wiring diagram of the encoder cable (CR750-D series controller)

∗Refer to "Table 15-4 Connectors: CNENC/CNUSR Pin Assignment" with pin assignment of connector CNUSR.

(2) CR751-D series controller

The wiring example by the thing is shown below.
(Please note that the connector shape is different depending on the controller)

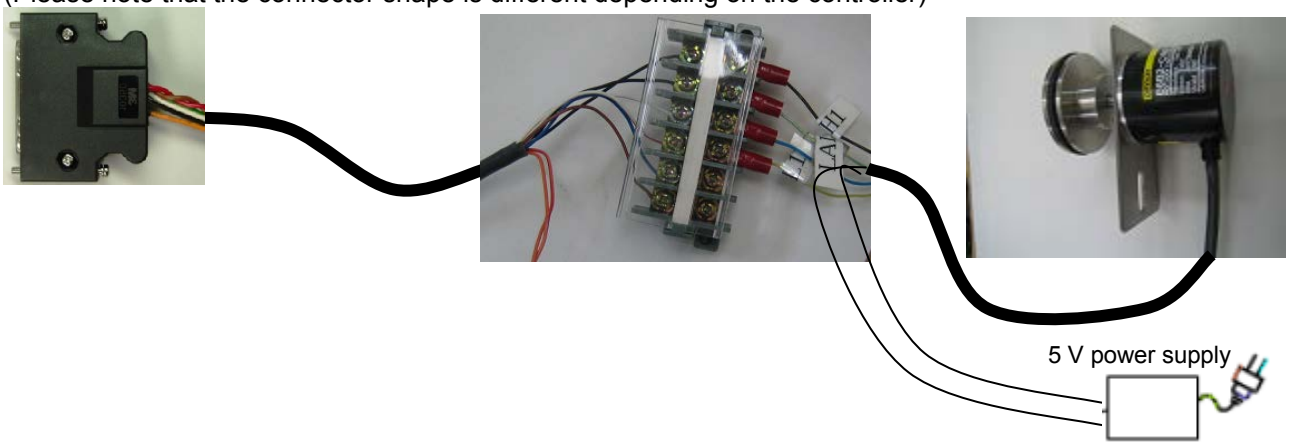


Figure5-7 Wiring example (CR751-D series controller)

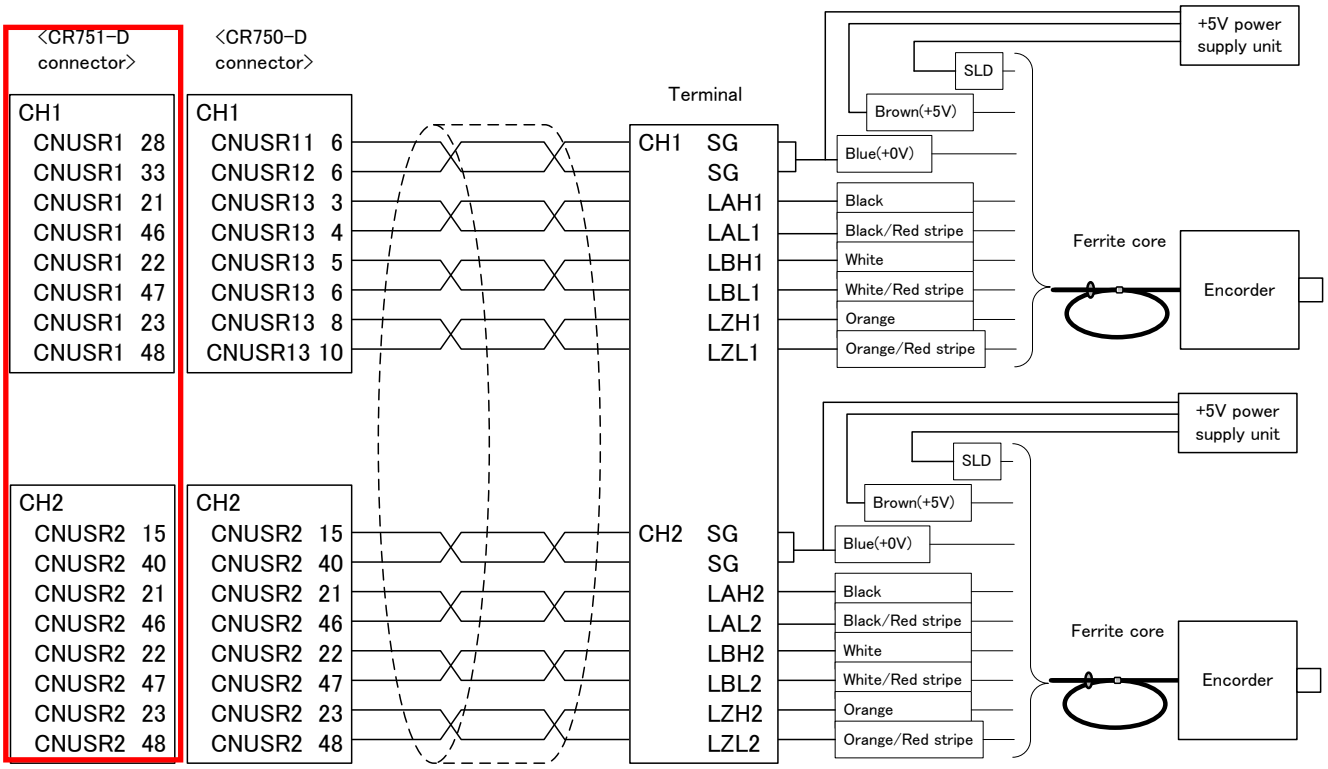


Figure5-8 The encoder and the wiring diagram of the encoder cable (CR751-D series controller)

*Refer to "Table 15-4 Connectors: CNENC/CNUSR Pin Assignment" with pin assignment of connector CNUSR.

5. 2. 2. Installation of encoder cable

The installation method of the encoder cable is shown by controller to be used.
*CR750-D series: “Figure5-9 Installation of encoder cable (CR750-D series)”
*CR751-D series: “Figure5-10 Installation of encoder cable (CR751-D series)”
And, the description about the measures against the noise is shown in the figure “5.3 Measures against the noize”.

(1) CR750-D series controller

<CR750-D series controller (rear)>

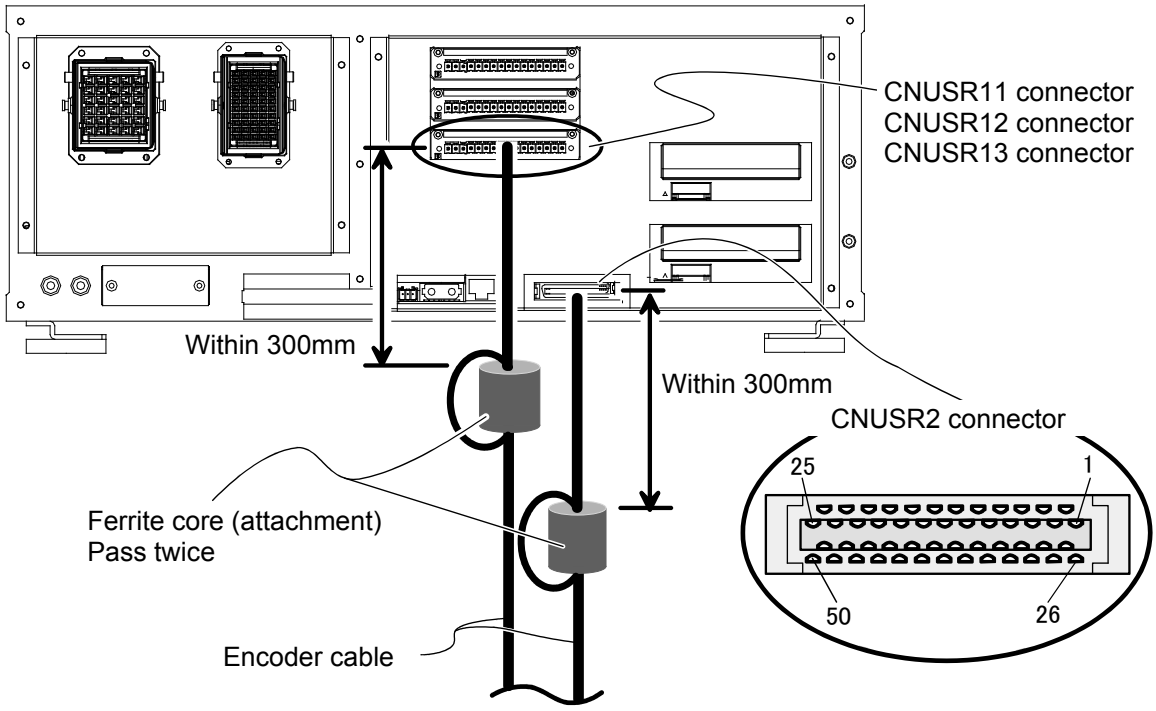


Figure5-9 Installation of encoder cable (CR750-D series)

(2) CR751-D series controller

<CR750-D series controller (front)>

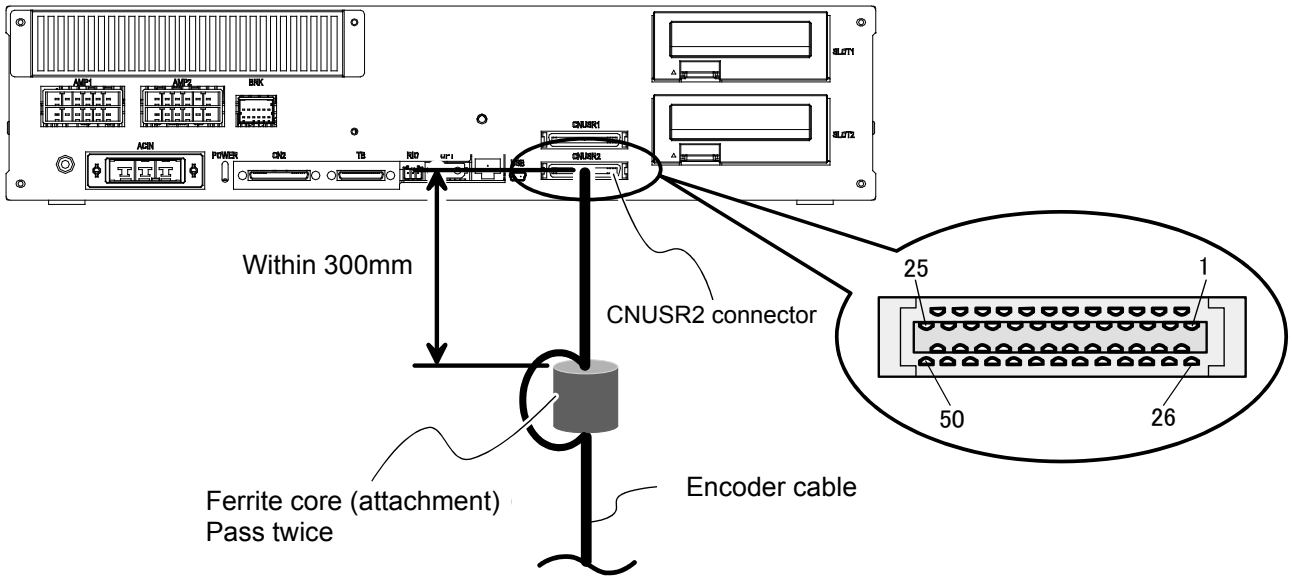
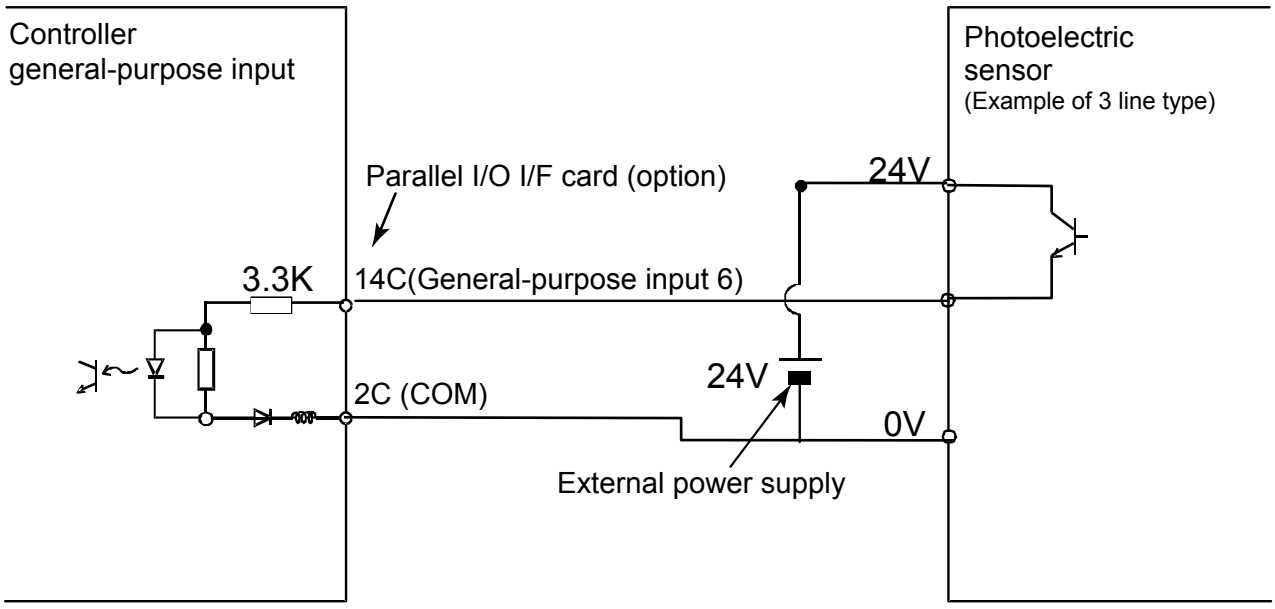


Figure5-10 Installation of encoder cable (CR751-D series)

5. 2. 3. Connection of Photoelectronic Sensor

If a photoelectronic sensor is used for detection of workpieces, connect the output signal of the photoelectronic sensor to a general input signal of the robot controller. Any general input signal number of the robot controller can be selected.
In this section, a connection example where the photoelectronic sensor signal is connected to the 6th general input signal is shown in “Figure5-11 Photoelectronic Sensor Connection Example (6th General Input Signal is Used) [D type]”.



Note) The external power supply and photoelectric sensor must be prepared by the customer.

Note) This connection example shows the connection of the source type.

Figure5-11 Photoelectronic Sensor Connection Example (6th General Input Signal is Used) [D type]

5. 2. 4. Connection of Vision Sensor

If a vision sensor is used for detection of workpieces, connect “HS OUT 0” and “GROUND (Micro series: HS COMMON)” of the vision sensor to SKIP input terminal of CNUSR connector.
In this section, the connection example to 2 channel (10, 35) is shown below.
Refer to a manual of the vision sensor which you use about specification of a breakout cable.

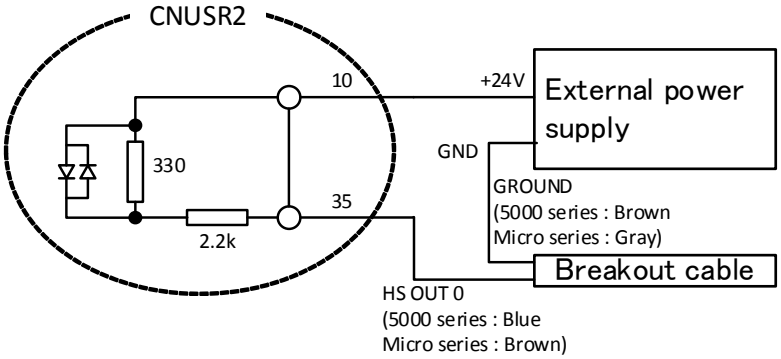


Figure5-12 Vision Sensor Connection Example (SKIP “2” Input Signal is Used) [D type]

5. 3. Measures against the noize

The example of noise measures of the tracking system is shown in the following.
Please implement the measures against the noise if needed in the power supply periphery section for the encoders which prepared of the customer.

1) Please insert AC line filter (recommendation: MXB-1210-33 * Densei-Lambda) in the AC input side cable of the power supply for the encoders.
2) Please insert the ferrite core (recommendation: E04SR301334 * SEIWA ELECTRIC MFG.) in the DC output side cable of the power supply for the encoders.
3) Please connect the power supply case for the encoders to the installation operator control panel, connect the earth wire to grounding or the case, and insert the ferrite core (recommendation: E04SR301334 * SEIWA ELECTRIC MFG.).

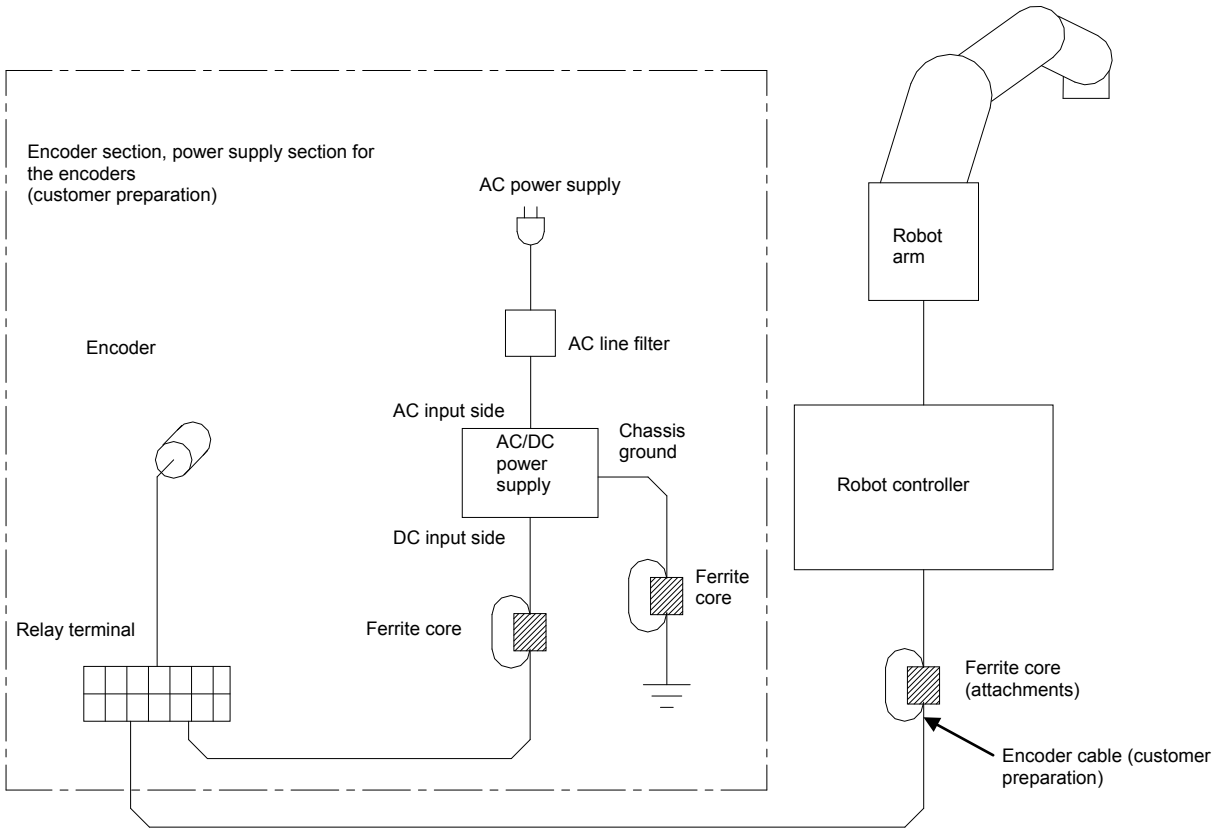


Figure5-13 Example of noise measures of tracking system

6. Parameter Setting

This chapter explains how to set dedicated input/output signals that play the role of interface between a robot and an external device (e.g., a Programmable Logic Controller) and parameters related to the tracking function.
Please refer to “Detailed Explanations of Functions and Operations” for how to set the parameters.

6.1. Tracking Parameter Setting

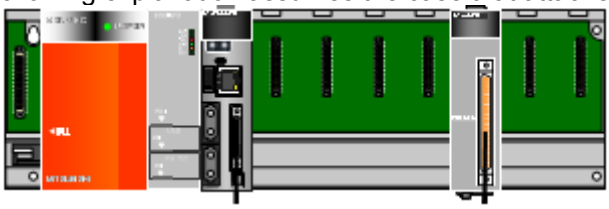
Specify to which channel of the encoder connector an encoder of conveyer is connected.
The parameter to set is shown below, make settings as required.

6.1.1. Sequencer CPU Parameter Setting [Q type]

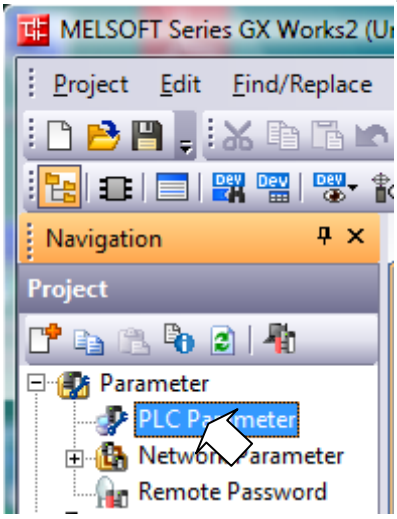
In the case of Q type, it is necessary to set multi CPU related parameters for both the sequencer CPU and robot CPU In order to use the sequencer link function.

a) Multiple CPU setting : Set the number of CPU units.
b) I/O assignment : Select I/O units and/or Intelligent units.
c) Control PLC setting : Set the CPU Unit numbers which control the Q173DPX unit.

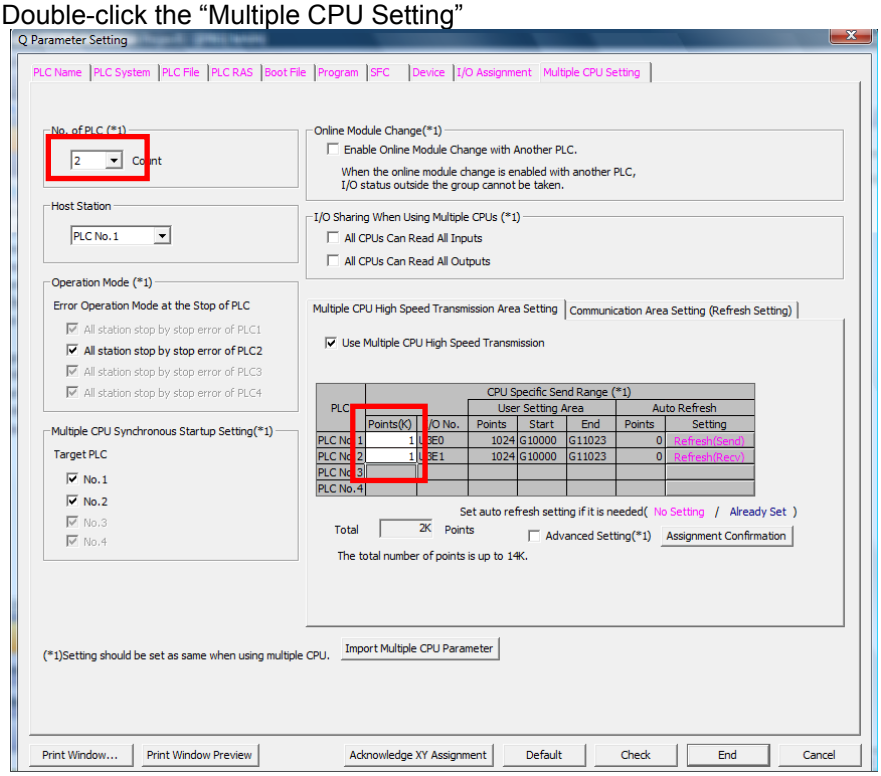
The setting procedure of the parameter is as below.
The following explanation assumes the case that attached Q173DPX unit to the fifth slot of baseboard.



(1) Execute the GX Works2 and select the project file.
(2) Double-click the “PLC Parameter”, then the “Q Parameter Setting” is displayed.

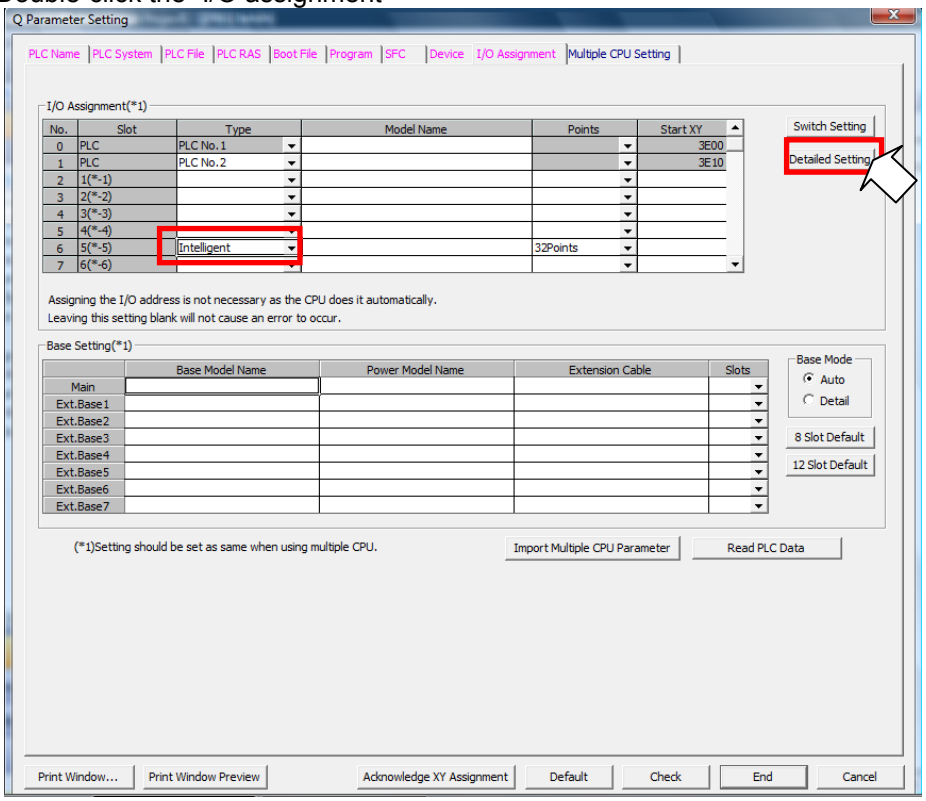


(3) Double-click the “Multiple CPU Setting”



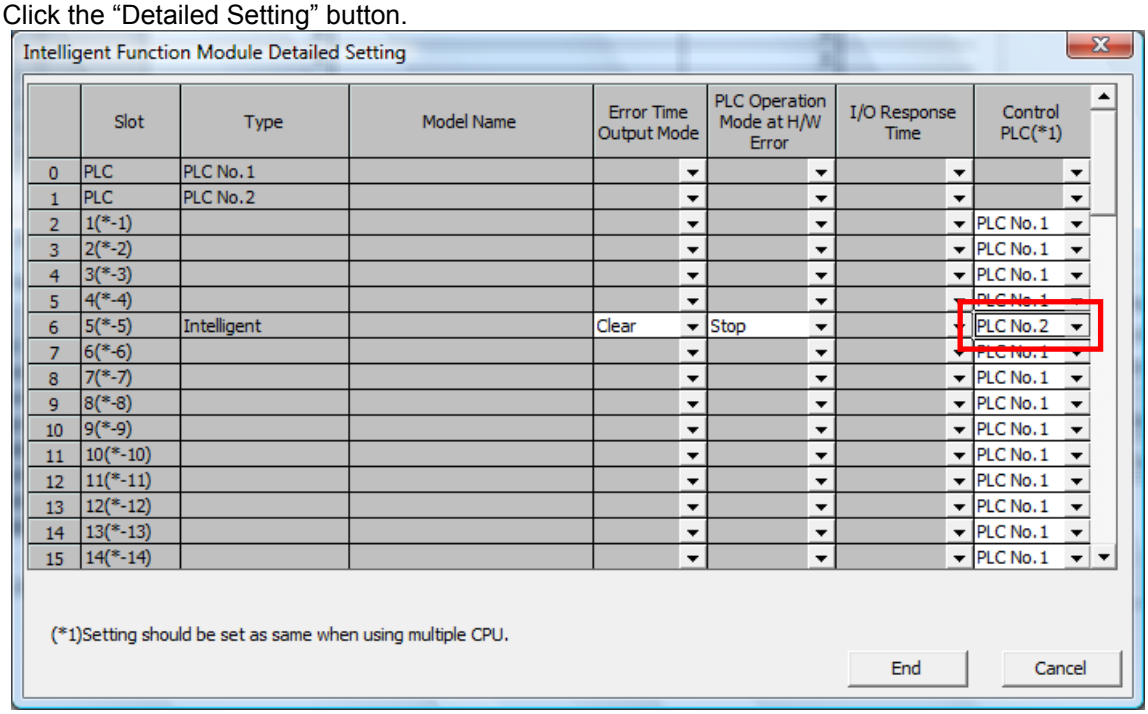
Set the number of CPU and this system area size (K Points)

(4) Double-click the “I/O assignment”



When Q173DPX unit is attached to fifth slot, change the type of slot 5 to the “Intelligent”.

(5) Click the “Detailed Setting” button.



Because the robot CPU manages the Q173DPX unit, change the Control PLC of slot 5 to the “PLC No.2” (Robot CPU).

(6) Click the “END” button. The Parameters are memorized into the sequencer CPU.
(7) A power supply of a sequencer is reset.
(8) Close GX Works2.

6.1.2. Robot Parameter Setting

After the installation of Q173DPX module and connection with the encoder are complete, use the following steps to establish robot CPU parameters.

(1) Set a parameter TRMODE to 1, validate a function of tracking.
(2) Specify the channel to which the encoder is connected using a parameter EXTENC.
(3) In the case of Q type, Using parameter ENCUNT* (*=1 to 3), designate the slot in which Q173DPX module under the control of robot CPU is installed.
(4) Reset a power supply and reflect a parameter.

Table 6-1 Tracking Parameter Setting

<table>
<tr><th>Parameter</th><th>Parameter name</th><th>Number of elements</th><th>Explanation</th><th>Value set at factory shipping</th></tr>
<tr><td>Tracking mode</td><td>TRMODE</td><td>1 integer</td><td>Enable the tracking function
Please set it to “1" when you use the tracking function.
0: Disable/1: Enable</td><td>0 → 1</td></tr>
<tr><td>Encoder number allocation

(*1)</td><td>EXTENC</td><td>8 integers</td><td>Set connection destinations on the connector for encoder numbers 1 to 8.
Parameter elements correspond to encoder number 1, encoder number 2 … encoder number 8 of a state variable "M_Enc" from the left.
Setting value is input encoder physics number from below list.
<table>
<tr><th rowspan="2">Encoder physics number</th><th colspan="2">Connection channel</th></tr>
<tr><th>Q</th><th>D</th></tr>
<tr><td>1</td><td>1st channel of Parameter ENCUNIT1</td><td>Standard CH1</td></tr>
<tr><td>2</td><td>2nd channel</td><td>Standard CH2</td></tr>
<tr><td>3</td><td>3rd channel</td><td>-</td></tr>
<tr><td>4</td><td>1st channel of Parameter ENCUNIT2</td><td>-</td></tr>
<tr><td>5</td><td>2nd channel</td><td>-</td></tr>
<tr><td>6</td><td>3rd channel</td><td>-</td></tr>
<tr><td>7</td><td>1st channel of Parameter ENCUNIT3</td><td>-</td></tr>
<tr><td>8</td><td>2nd channel</td><td>-</td></tr>
</table>
In the case of Q type, it is convenient to check the status variable “M_Enc” when determining the setting value of the “EXTENC” parameter.
In the case of D type, The value of the encoder which wired the channel 1 in case of the standard encoder input connector [CNENC] for the robot controller is equipped with the encoder cable with initial setting, The value of the encoder which wired the channel 2 by the status variable "M_Enc (1)", "M_Enc (3)", "M_Enc (5)", and "M_Enc (7)",It can confirm by the status variable "M_Enc (2)", "M_Enc (4)", "M_Enc (6)", and "M_Enc (8)."

Please refer to "13.1.2 List of Robot Status Variables” for the explanation of state variable “M_Enc".
Please refer to “Detailed Explanations of Functions and Operations” for how to check the status variable.</td><td>[Q type]
1,2,3,4,5,6,7,8

[D type]
1,2,1,2,1,2,1,2

↓

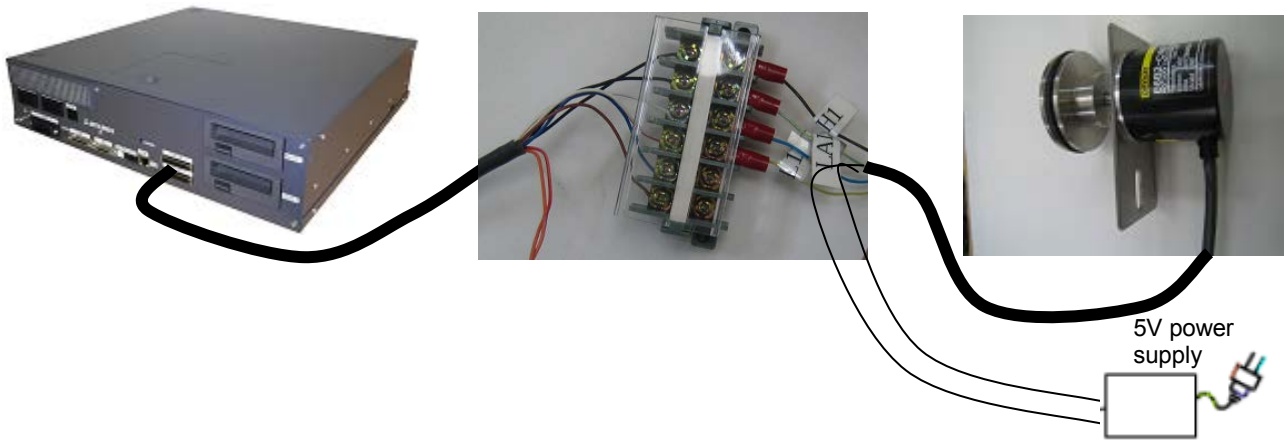
Change the set value according to the situation.</td></tr>
</table>

Parameter	Parameter name	Number of elements	Explanation	Value set at factory shipping
Tracking Workpiece judgment distance	TRCWDST	1 integer	Distance to judge that the same workpiece is being tracked (mm) The sensor reacts many times when the workpiece with the ruggedness passes the sensor. Then, the robot controller judged that one workpiece is two or more pieces. The sensor between values [mm] set to this parameter does not react after turning on the sensor.	5.00 ↓ Size of the workpiece
first Q173DPX	ENCUNIT1	2 integers	The base unit-number of the first Q173DPX unit (element 1) that robot CPU uses and slot number (element 2) are set. [Element 1] -1 : No connection 0 : Basic base unit 1 - 7 : Increase base unit [Element 2] 0 - 11 : I/O Slot number	[Q type] -1,0 ↓ Installation place of Q173DPX
Second Q173DPX	ENCUNIT2	2 integers	The base unit-number of the second Q173DPX unit (element 1) that robot CPU uses and slot number (element 2) are set. [Element 1] -1 : No connection 0 : Basic base unit 1 - 7 : Increase base unit [Element 2] 0 - 11 : I/O Slot number	[Q type] -1,0

(*1) The example of a setting of a parameter EXTENC is shown as follow.

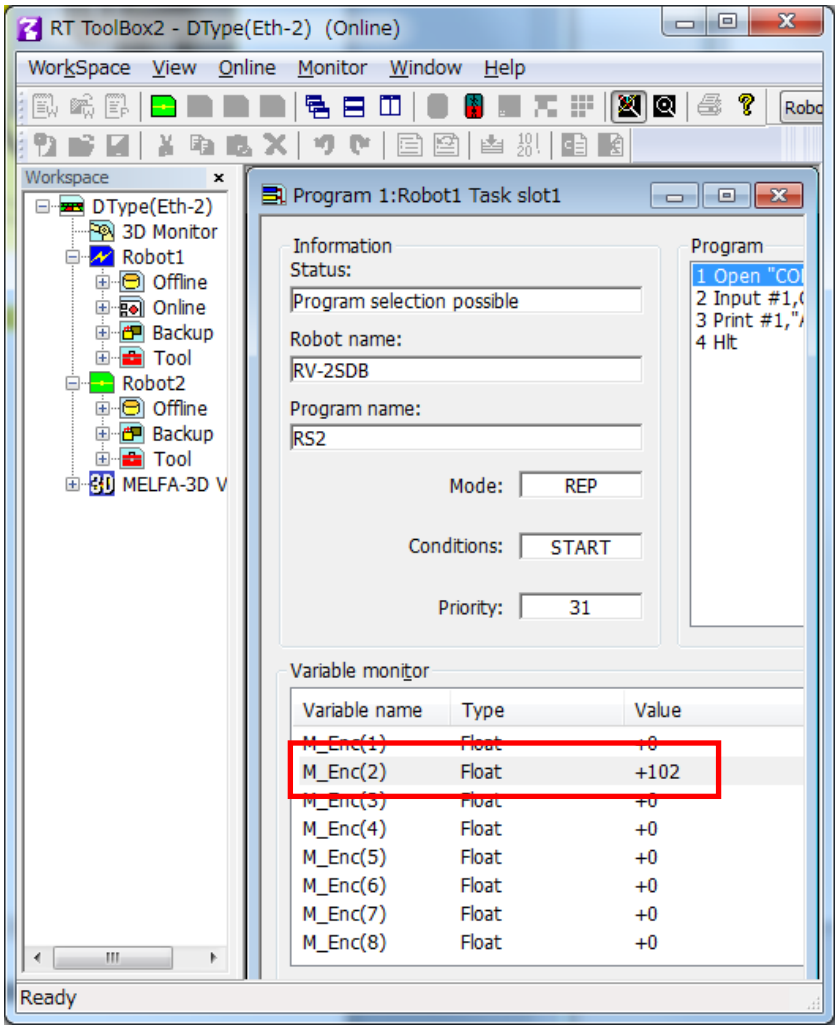
Hardware configuration

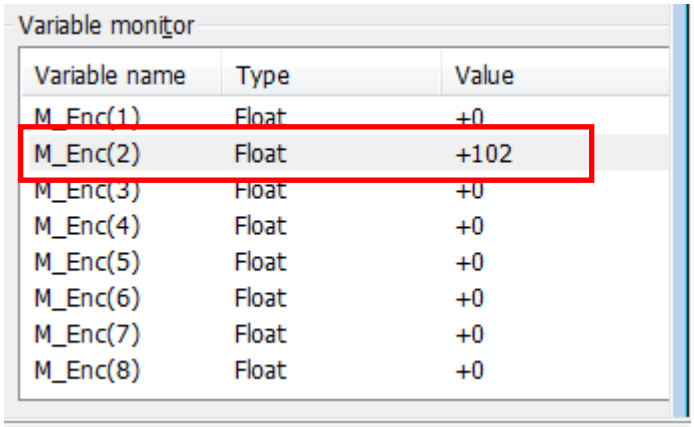
In CR750-D and a CR751-D controller, when using a common encoder cable, it is convenient to use CNUSR2 connector.
In this case, in order to connect with the channel 2 of an encoder, an encoder value will be checked using a state variable "M_Enc (2)."



Monitoring the encoder value

When the encoder value is showed by variable monitor of “Program monitor”, the encoder value changes as follows.



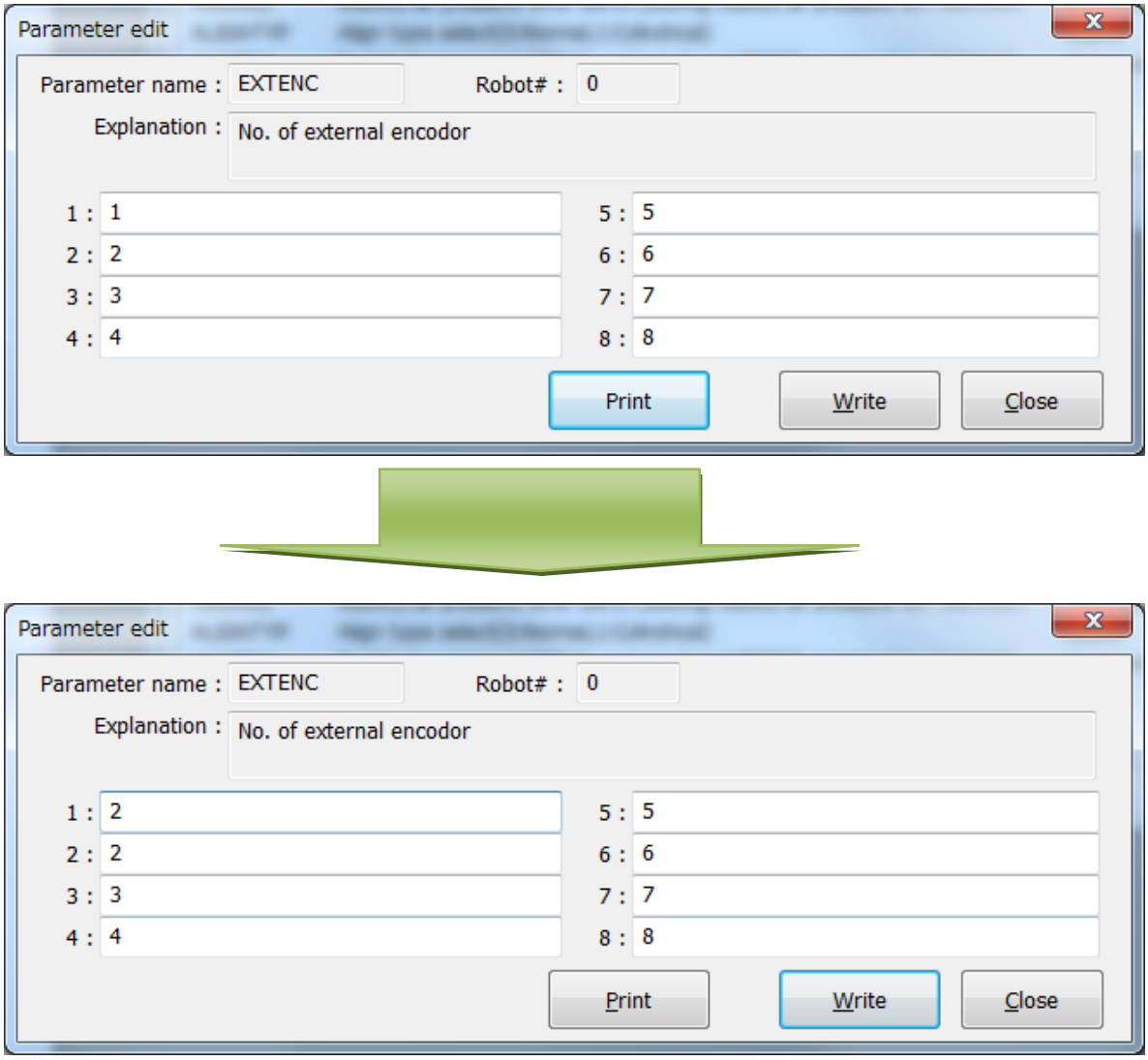


In this way, in the case of connection to channel 2, the encoder data is stored in “M_Enc(2)”.

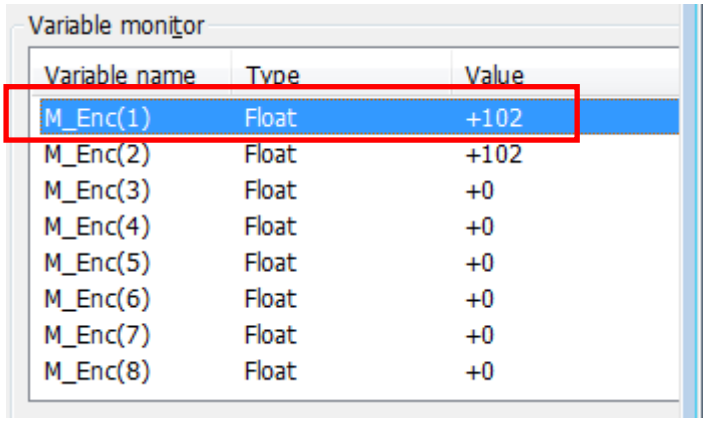
It is useful to change parameter EXTENC when confirming the encoder value by using “M_Enc(1)”and encoder value 1.

Common control to "M_Enc(1)" by parameter EXTENC

Change the first element of a parameter EXTENC into "2" from "1".



If you reset a power supply and reflect the parameter value, the encoder value is displayed in M_Enc(1)" as follows.

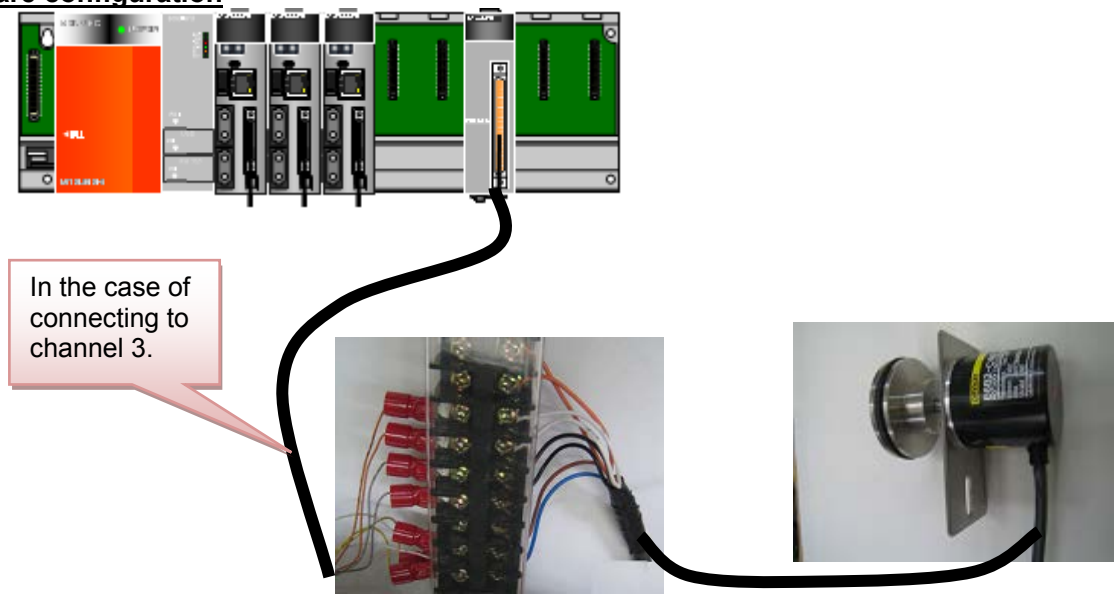


6.1.3. Example of three robot's CPU sharing one Q173DPX [D type]

For example, the setting of one Q173DPX ,three robots CPU, and one encoder is shown as follows.

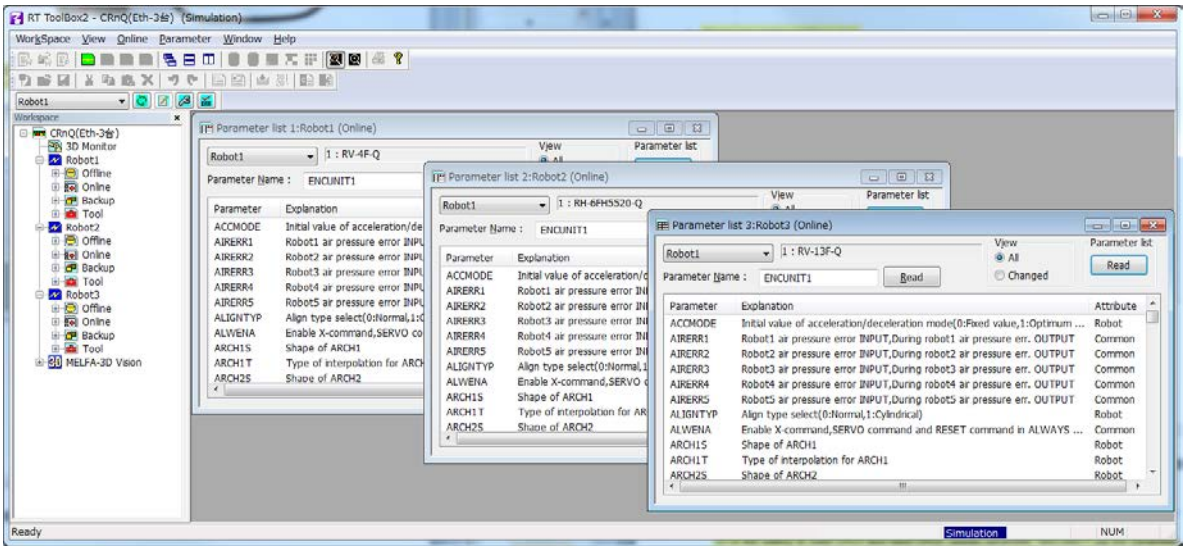
You will be able to understand some parameters ENCUNIT* and EXTENC.
[Conditions]
- An encoder is connected to the channel 3.
- Robot CPU1 and 2 use the parameter ENCUNIT1 and robot CPU3 uses the parameter ENCUNIT2.

Hardware configuration

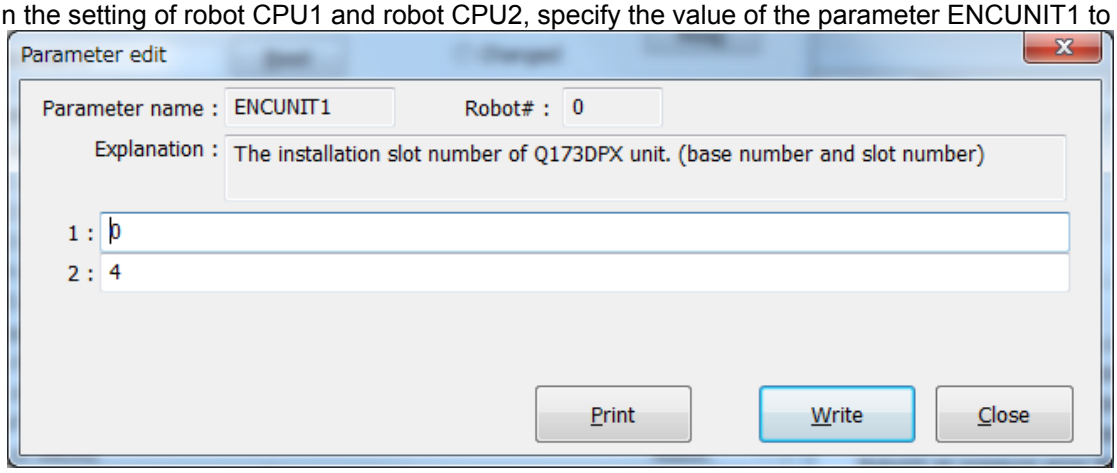


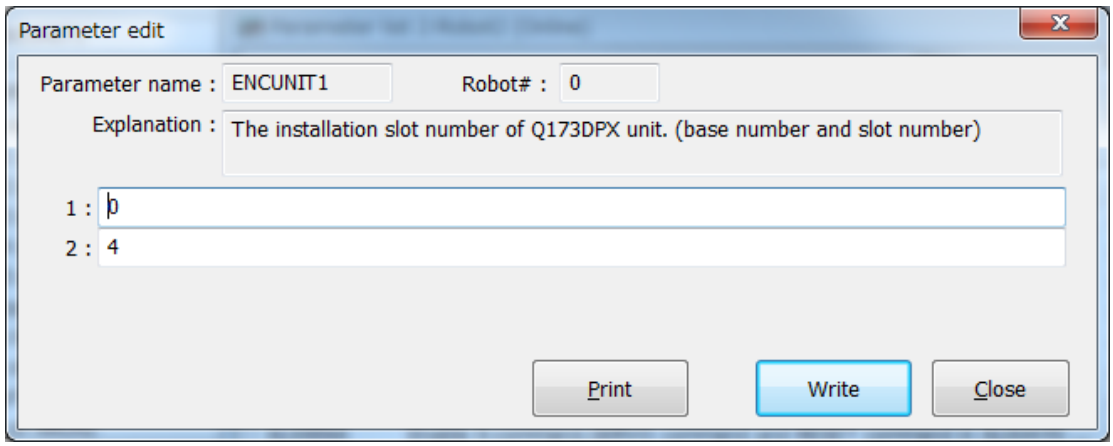
Parameter setting of the robot

(1) Display the list of parameters of three robots CPU.

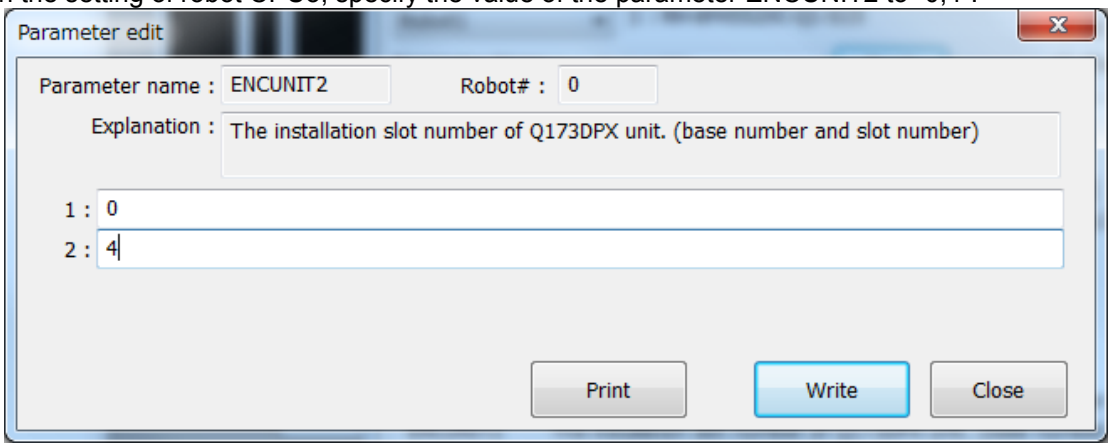


(2) In the setting of robot CPU1 and robot CPU2, specify the value of the parameter ENCUNIT1 to “0,4”.

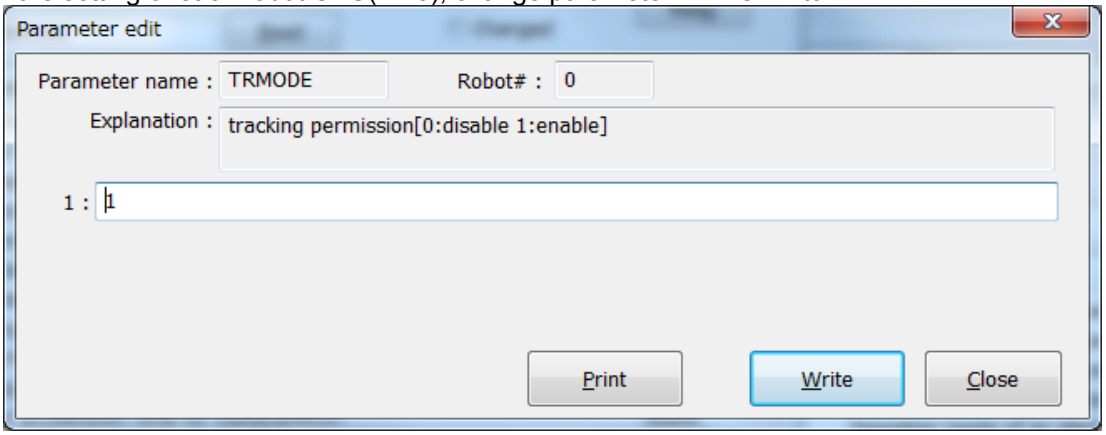




(3) In the setting of robot CPU3, specify the value of the parameter ENCUNIT2 to “0,4”.

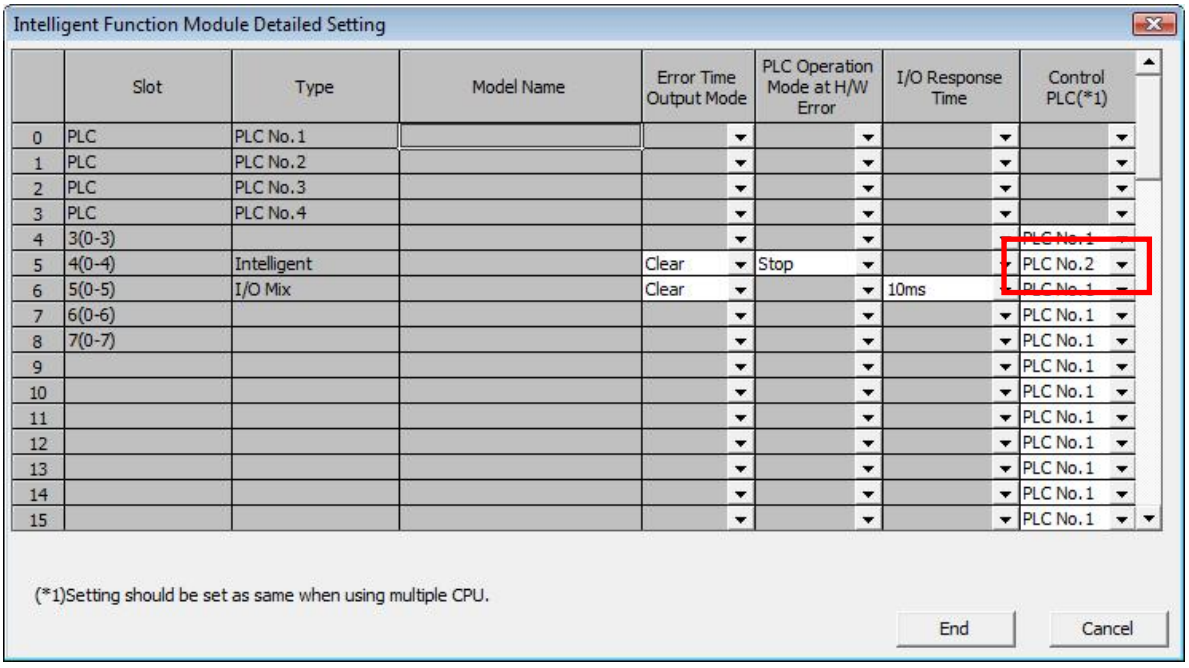


(4) In the setting of each robot CPU(1 - 3), change parameter TRMODE to “1”.



Parameter setting of GX Works

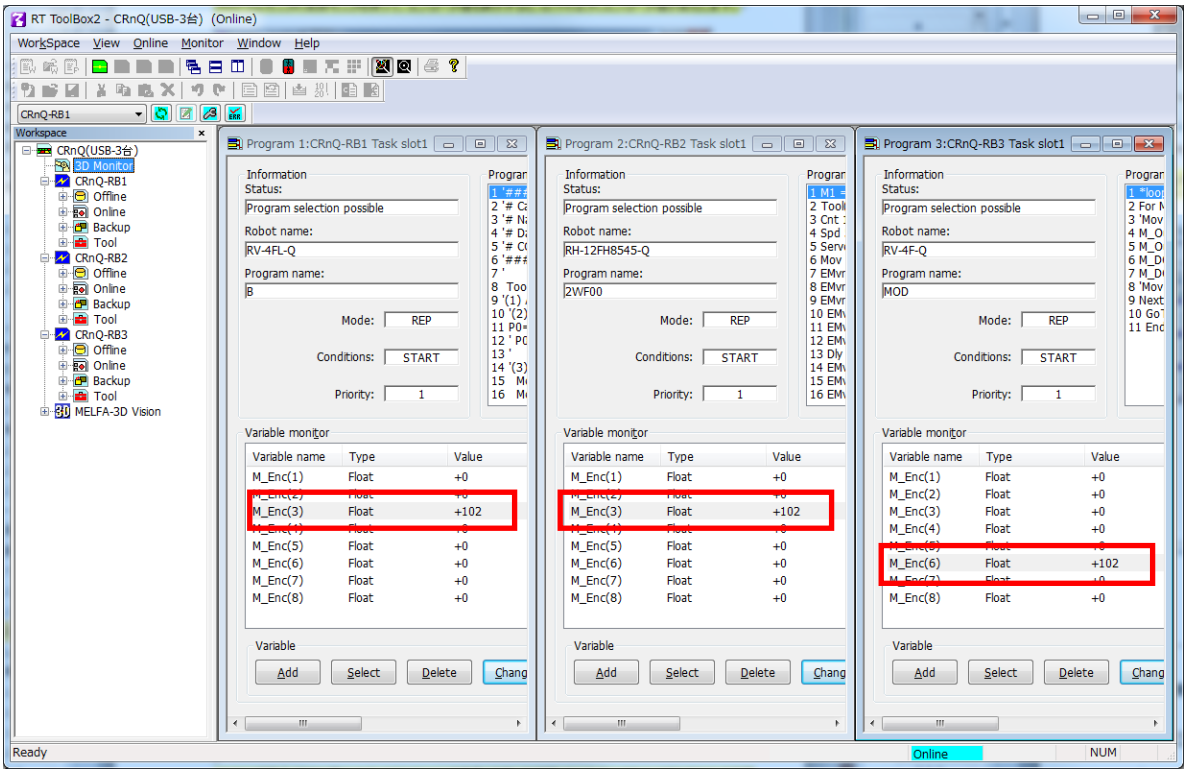
The example of the second unit (robot CPU1) controlling Q173DPX unit.

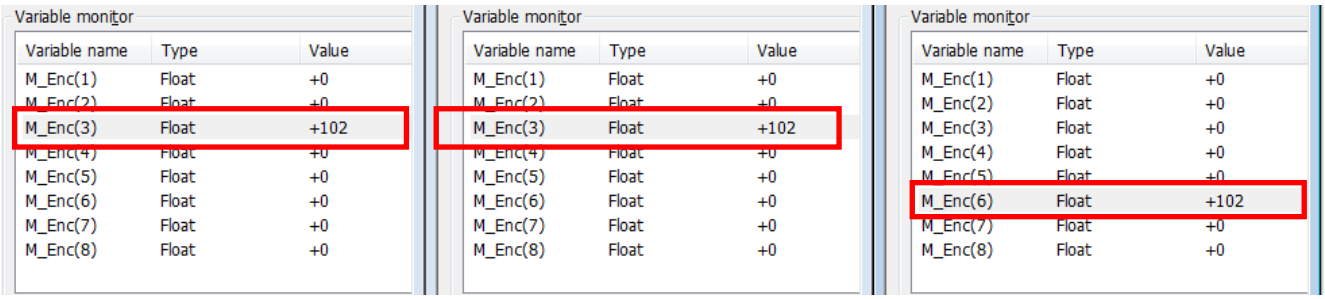


Change “Control PLC” columns to “PLC No.2” in slot 4(0-4) rows of No.5.
Reset the power supply of sequencer and the robot controller after the setting was changed.

Monitoring the encoder value

When the encoder value is showed by variable monitor of “Program monitor”, the encoder value changes as follows.



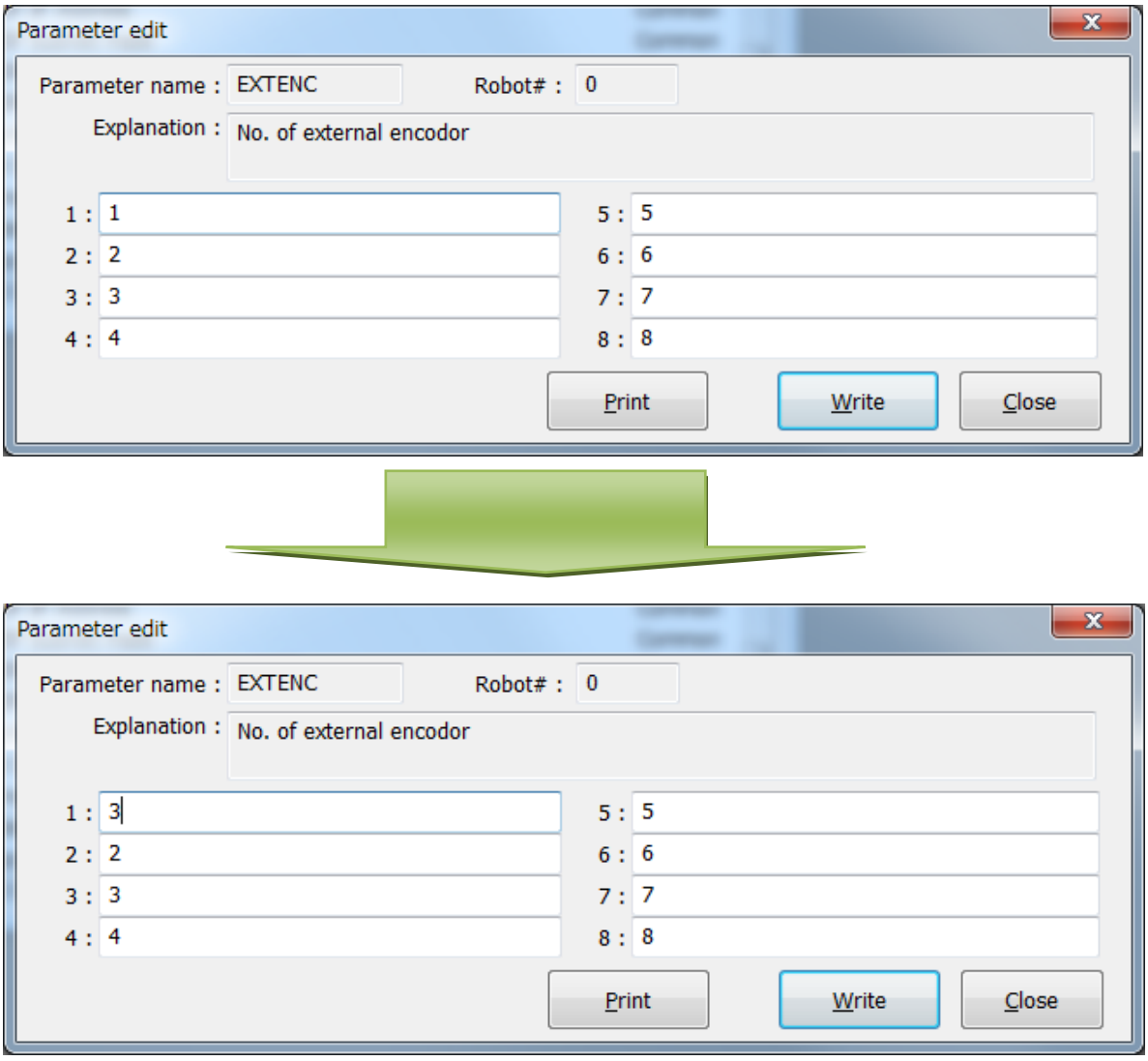


In this way, in the case of connection to channel 3, the data of robot CPU1 and robot CPU2 is stored in "M_Enc(3)".
The data of robot CPU3 is stored in"M_Enc(6)" because parameter ENCUNIT2 is specified.

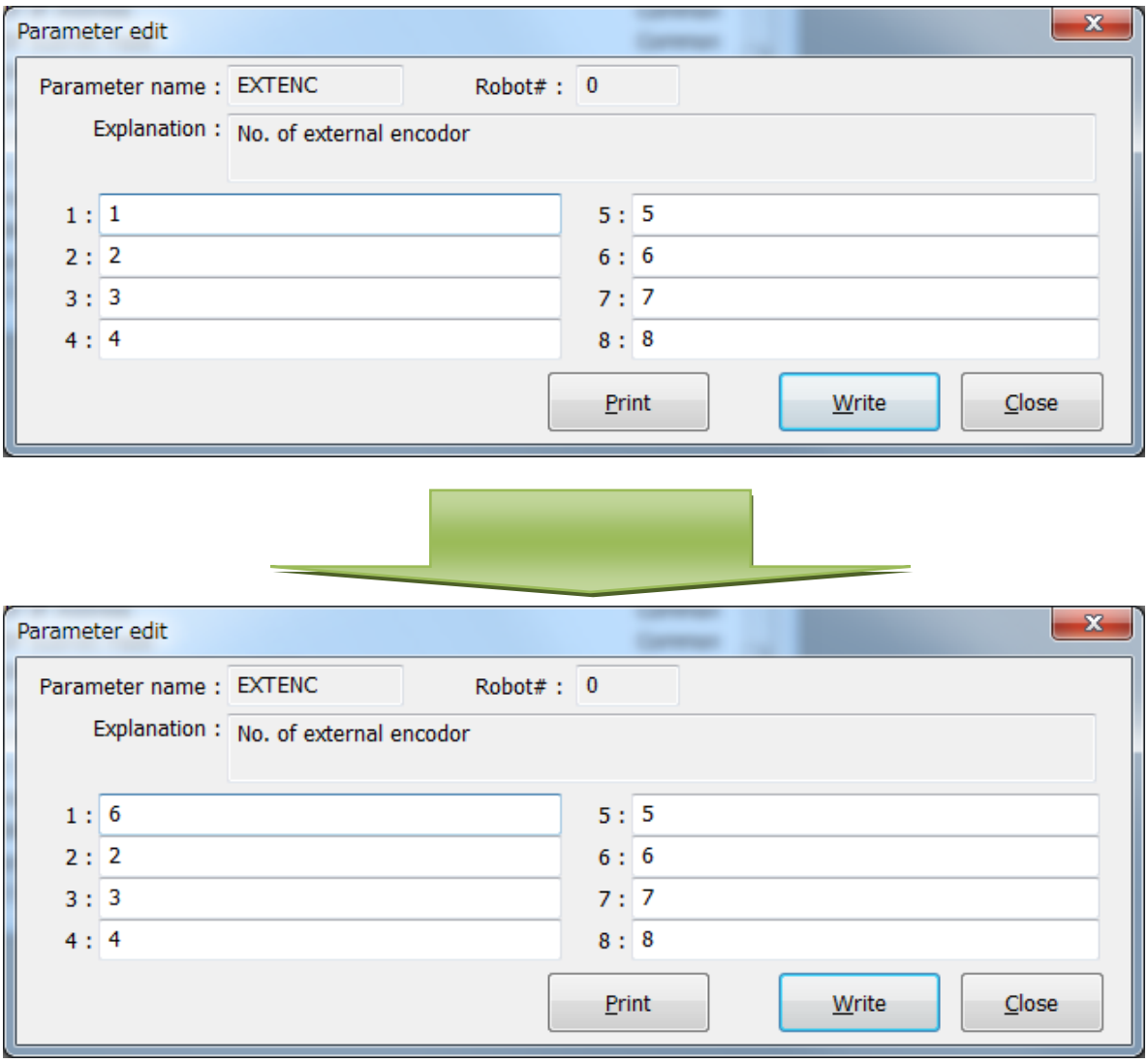
It is useful to change parameter EXTENC when confirming the encoder value by using "M_Enc(1)"and encoder value 1.

Common control to "M_Enc(1)" by parameter EXTENC

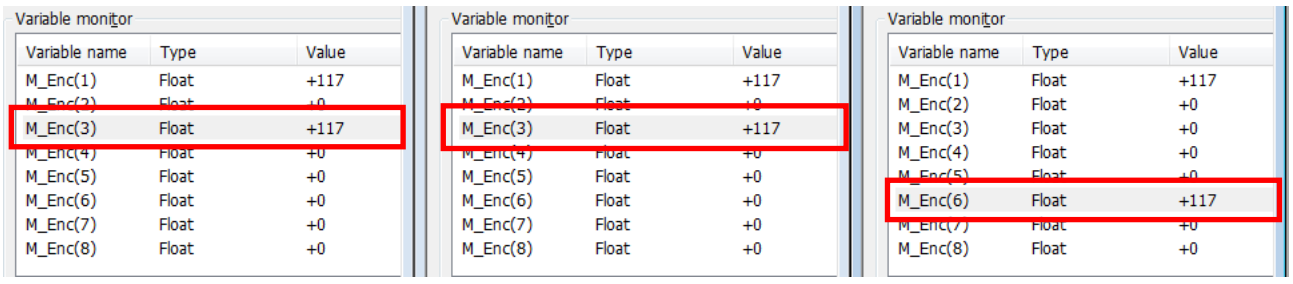
In the setting of the robot CPU1 and CPU2, change the first element of a parameter EXTENC into "3" from "1".



In the setting of the robot CPU3, changes the first element of a parameter EXTENC into “6” from “1”.



If you reset a power supply and reflect the parameter value, the encoder value is displayed in M_Enc(1)” as follows.



6.2. Operation Parameters

The following list the setting items of parameters required to operate the robot at the optimal acceleration/deceleration.

Table 6-2 List of Operation Parameter

Parameter name	Explanation	Reference value
Tool coordinate system (**MEXTL**) (*1)	A parameter "MEXTL" designates a coordinate system of a tool installed in the mechanical interface side of the robot (hand). For example it's possible to move and revolve based on a tip of a hand.	Defaults: (0,0,0,0,0,0,0) For example: (0,0,+80,0,0,0,0)
Tool data 1 - 16 (**MEXTL1 - 16**) (*1)	I can point out the tool data for 16 as needed. For example when changing a hand by a multi-hand and a hand changer, it's possible to establish and change the respective tool coordinate systems.	Defaults: (0,0,0,0,0,0,0) For example: (0,0,+80,0,0,0,0)
Optimal acceleration/ deceleration hand data (**HANDDAT1**)	Specify hand weight and so on to make settings that allow optimal acceleration/deceleration operations. For example, if the hand weighs 3 kg, changing the weight setting value from 10 kg to 3 kg makes the robot movement faster. (Hand weight (kg), size (mm) X, Y, Z, gravity (mm) X, Y, Z)	(3,0,0,0,0,0,0) The setting values are different for each robot model. **Use these values as reference only**.
Optimal acceleration/ deceleration workpiece data (**WRKDAT1**)	Specify workpiece weight and so on to make settings that allow optimum acceleration/deceleration operations. If a workpiece is grabbed via the HClose instruction, the acceleration/deceleration becomes slower. If a workpiece is released via the HOpen instruction, acceleration/deceleration becomes faster. (Workpiece weight (kg), size (mm) X, Y, Z, gravity (mm) X, Y, Z)	(1,0,0,0,0,0,0) The setting values are different for each robot model. **Use these values as reference only.**

(*1) Refer to “8.1.1 Setting of tool length” about setting of a tool length.

6.3. Dedicated Input/Output Parameters

The following list the setting items of dedicated input/output parameters used to operate the robot via instructions from an external device. Set the signal numbers according to your system using the setting values in the table as reference. **It is not necessary to set these parameters if the robot operates by itself, rather than via instructions from an external device.**

Table 6-3 List of Dedicated Input/Output Parameters

Input name/output name (parameter name)	Explanation	Setting Example (*1)	
		Q	D
Stop/pausing (**STOP**) or (**STOP2**)	Input: Stop a program Output: Output program standby status	**10000, -1**	**0 , -1**
Servo OFF/servo ON disabled (**SRVOFF**)	Input: Turn the servo off Output: Output servo ON disabled status	**10011, -1**	**1 , -1**
Error reset/error occurring (**ERRRESET**)	Input: Cancel error status Output: Output error status	**10009, -1**	**2 , -1**
Start/operating (**START**)	Input: Start automatic operation Output: Output program running status	**10006, 1**	**3 , 1**
Servo ON/turning servo ON (**SRVON**)	Input: Turn the servo on Output: Output servo on status	**10010, 0**	**4 , 0**
Operation right/operation right enabled (**IOENA**)	Input: Enable/disable operation right of external signal control Output: Output external signal control operation enabled status	**10005, -1**	**5 , -1**
Program reset/program selectable (**SLOTINIT**)	Input: Initiate a program. The program execution returns to the first step. Output: Output a status where program No. can be changed	**10008, -1**	**10 , -1**

Input name/output name (parameter name)	Explanation	Setting Example (*1)	
		Q	D
General output signal reset (**OUTRESET**)	Input: Reset a general output signal	**10015, -1**	**11 , -1**
User specification area 1 (**USRAREA**)	Output an indication that the robot is in an area specified by a user Set the start number and end number	**10064, 10071**	**8 , 8**

(*1) **"-1" in the Setting value column means "not set."**

7. Installation of a sample robot program

This chapter explains the structure of the sample robot programs.

Two types of sample robot programs are provided; for conveyer tracking and for vision tracking.
Their program structures are shown in “Table 7-1　List of Sample Robot Programs（Conveyer Tracking）” and “Table 7-2　List of Sample Robot Programs（Vision Tracking）” respectively.
Refer to “RT ToolBox2 Robot Total Engineering Support Software Instruction Manual” for how to install programs to the robot controller.

7.1. Conveyer Tracking

Table 7-1　List of Sample Robot Programs（Conveyer Tracking）

Program name	Description	Explanation
A1	Conveyer - robot coordinate system calibration program	This program matches the coordinate systems of the conveyer and robot and calculates the amount of robot movement per encoder pulse.
C1	Workpiece coordinate system - robot coordinate system matching program	This program calculates the coordinates at which the robot grabs a workpiece based on the coordinates at which a sensor is activated.
1	Operation program	This program handles transporting workpieces while following recognized workpieces. (1) Movement to the robot origin (2) Workpiece suction and transportation operation while following movement
CM1	Workpiece coordinate monitor program	This program monitors encoder values and stores workpiece coordinates.

7.2. Vision Tracking

Table 7-2　List of Sample Robot Programs（Vision Tracking）

Program name	Description	Explanation
A1	Conveyer - robot coordinate system calibration program	This program matches the coordinate systems of the conveyer and robot and calculates the amount of robot movement per encoder pulse.
B1	Vision coordinate system – robot coordinate system calibration program	This program matches the vision coordinate system and the robot coordinate system.
C1	Workpiece coordinate system - robot coordinate system matching program	This program calculates the coordinates at which the robot grabs a workpiece based on the coordinates at which a vision sensor has detected the workpiece.
1	Operation program	This program handles transporting workpieces while following recognized workpieces. (1) Movement to the robot origin (2) Workpiece suction and transportation operation while following movement
CM1	Workpiece coordinate monitor program	This program monitors encoder values and stores workpiece coordinates.

8. Calibration of Conveyer and Robot Coordinate Systems (“A1” program)

This chapter explains the tasks carried out by using “A1” program.

∗ “A1” program contains operations required for both conveyer tracking and vision tracking.

Calibration of a conveyer refers to determining the movement direction of the conveyer in the robot coordinate system and the amount of movement of the robot per encoder pulse. This amount of movement is stored in the robot’s status variable “P_EncDlt.”

“A1” Program performs specified tasks and automatically calculates the amount of movement of the robot per encoder pulse mentioned above.

The procedures of operations specified by “A1” program and items to be confirmed after the operations are explained below.

Please refer to “Detailed Explanations of Functions and Operations” for the steps involved in each operation.

Please monitor status variable “M_Enc(1)" to “M_Enc(8)" before it works, rotate the encoder, and confirm the value changes.

8.1. Preliminary Preparaions

This chapter explains the knowledge about confirmation and operation necessary to a minimum before beginning work.

The contents which should be checked are "Tool length" and "change in the encoder value".

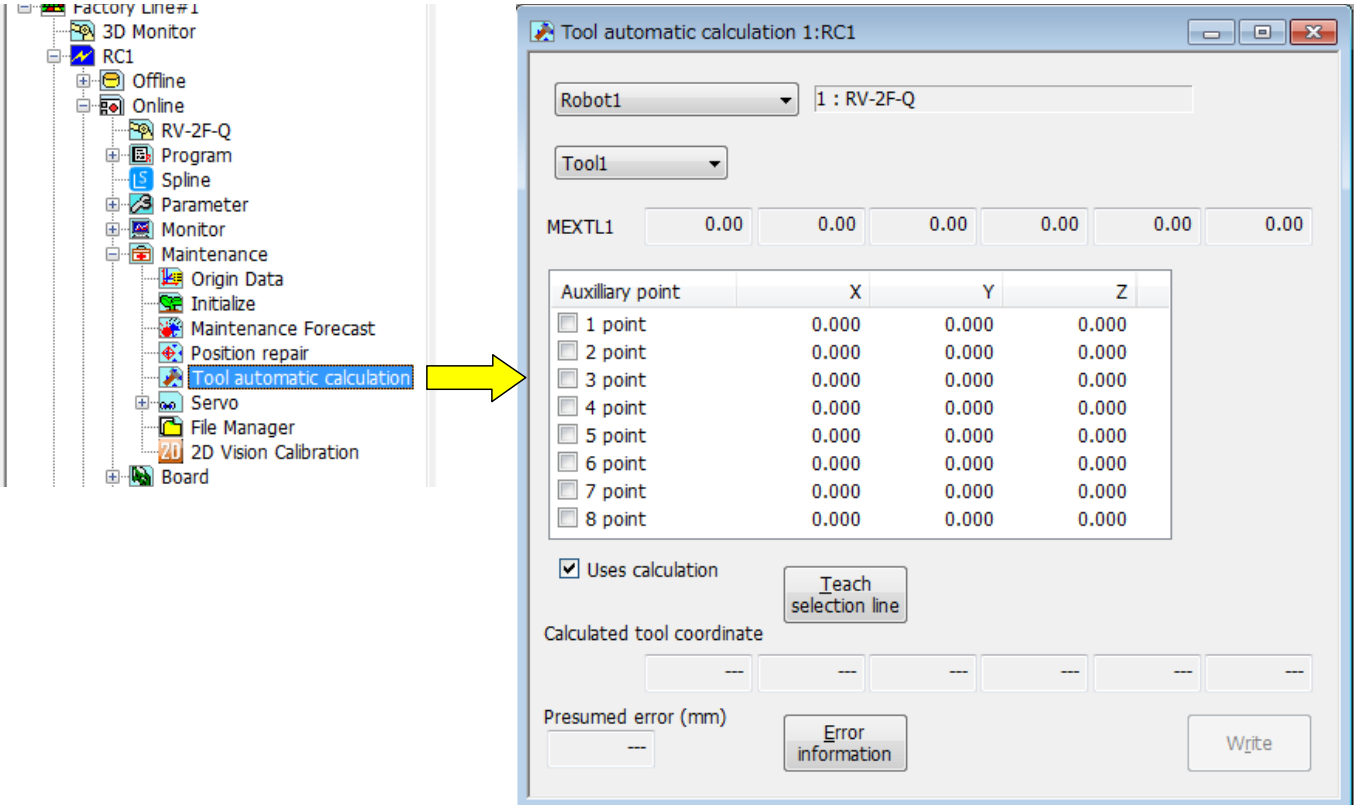
8.1.1. Setting of tool length

When you'd like to change the angle at the place which isn't a flange part of a robot(For example, tip of a hand), you have to set tool length.

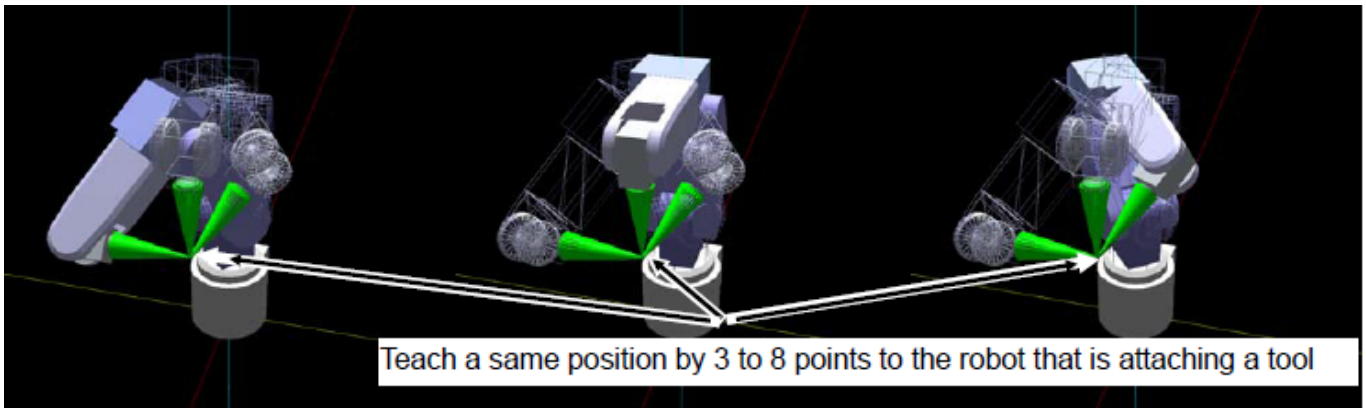
The "tool length automatic measuring system" function of RT ToolBox2 is useful when setting tool length.

Refer to “RT ToolBox2 Robot Total Engineering Support Software Instruction Manual” about operational details.

When the robot model and robot controller which have connected, correspond to this function, a [Tool automatic calculation] is displayed under [Maintenance] in the project tree. Double-click [Online] ->[Maintenance] -> [Tool automatic calculation] in the project tree.



Tool length is calculated automatically by instructing in the location of 3-8 points as follows in the screen mentioned above.



CAUTION

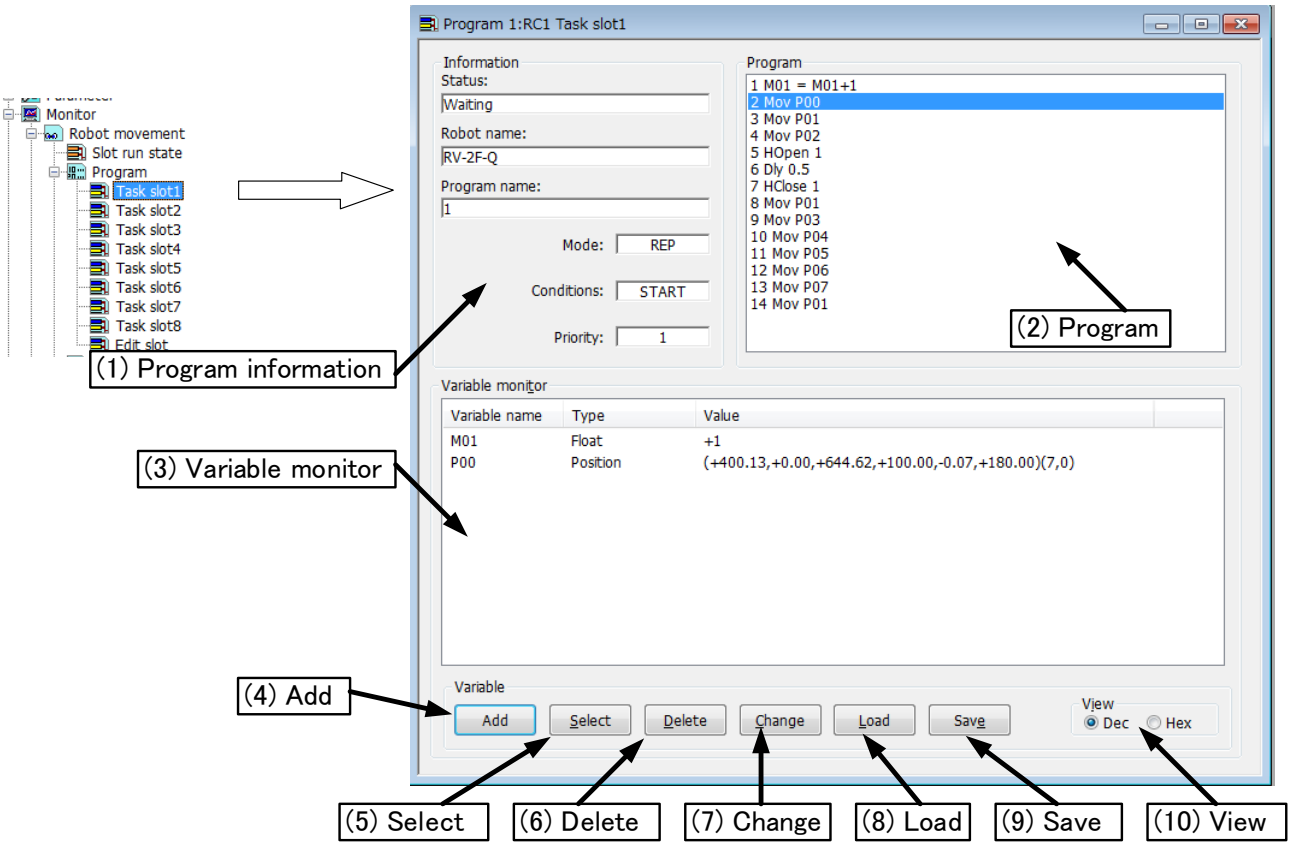
Move a robot arm to the correct location

Specify the correct location of 5-8 points as the "length" made to this work by the one of the precision of the tracking function.

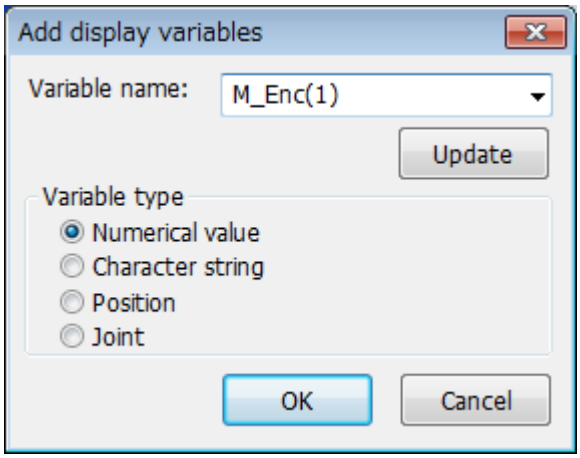
8. 1. 2. Confirm the encoder value

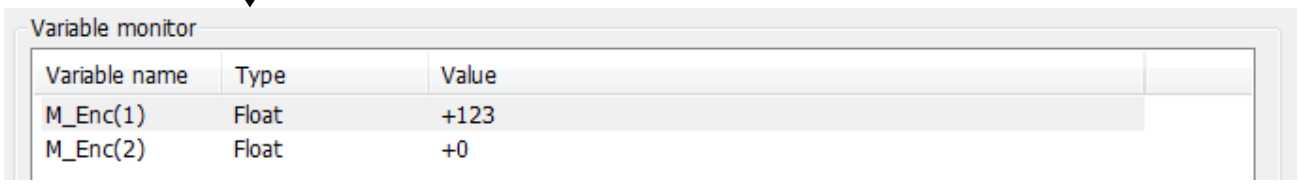
An important one is a change in the encoder value in this work. Confirm whether a robot controller grasps the turn of the encoder.

From the project tree, click the target project [Online] -> [Monitor] -> [Movement Monitor] -> [Program Monitor], then double click the "Task slot" to monitor.

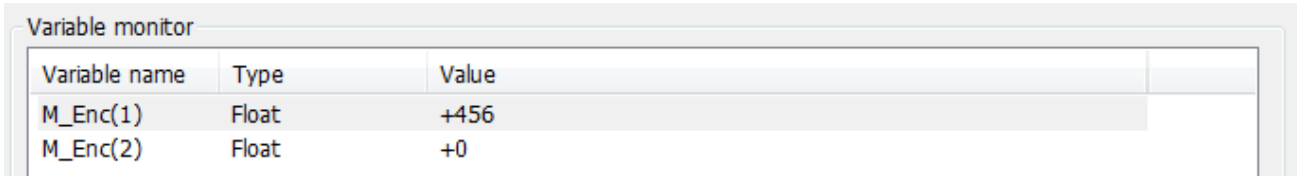


Click a [Add] button and open a "Add display variables" screen. Input "M_Enc (1)" to a space "variable name", and click a [OK] button. also input "M_Enc (2)"-"M_Enc (8)" equally, and click a [OK] button.





Confirm that the value of "M_Enc" changes by a revolution of a conveyer.

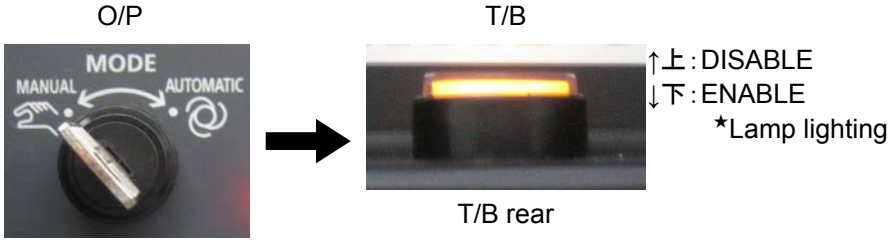


When the encoder value doesn't change, confirm the parameter setting and the wiring of "6.1.2 Robot Parameter Setting".

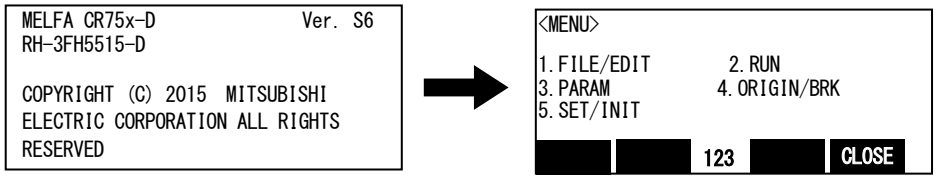
8.2. Operation procedure

Using "A1" program, operate in the following procedures.

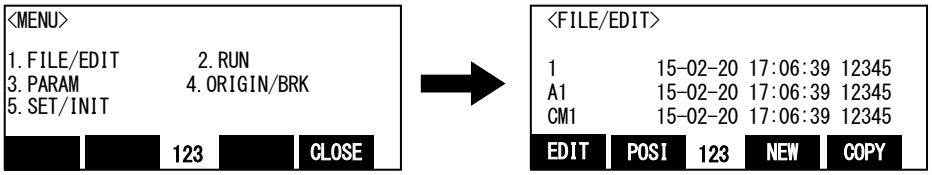
(1) Set the controller mode to "MANUAL". Set the T/B to "ENABLE".



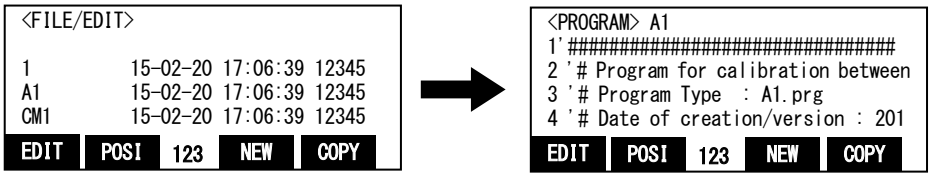
(2) Press one of the keys (example, [EXE] key) while the <TITLE> screen is displayed. The <MENU> screen will appear.



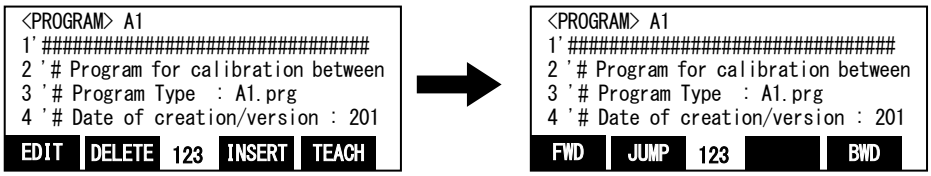
(3) Select "1. FILE /EDIT" screen on the <MENU> screen.



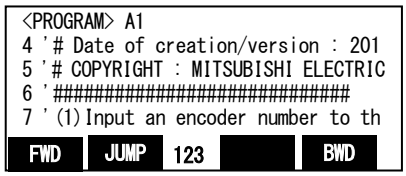
(4) Press the arrow key, combine the cursor with the program name "A1" and press the [EXE] key. Display the <program edit> screen.



(5) Press the [FUNCTION] key, and change the function display

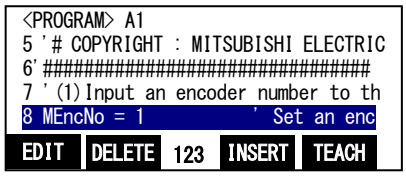


(6) Press the [F1] (FWD) key and execute step feed. "(1) Input an encoder … "is displayed. Execute work according to the comment in the robot program.

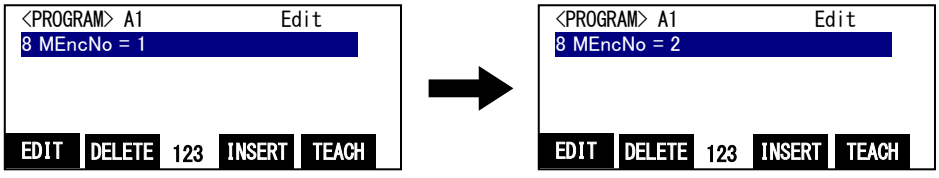


Specify the encoder number.
If you want to change the encoder number, please edit the program as follows.

(a) Display the following command.



(b) Press the [F1](FWD) key and specify the encoder number in the variable "MEncNo"
Example)When "2" is specified as the encoder number.



(c) Press the [F1] (FWD) key and the change is determined.

```
<PROGRAM> A1
5 '# COPYRIGHT : MITSUBISHI ELECTRIC
6'################################
7 ' (1)Input an encoder number to th
8 MEncNo = 2              ' Set an enc
EDIT DELETE 123 INSERT TEACH
```

(7) Press the [F1] (FWD) key and execute step feed. "(2) Attach a marking sticker…"is displayed.
Attach a marking sticker on the conveyer (a sticker with an X mark is the best choice for the marking sticker).Drive the conveyer and stop it when the marking sticker comes within the robot movement range.

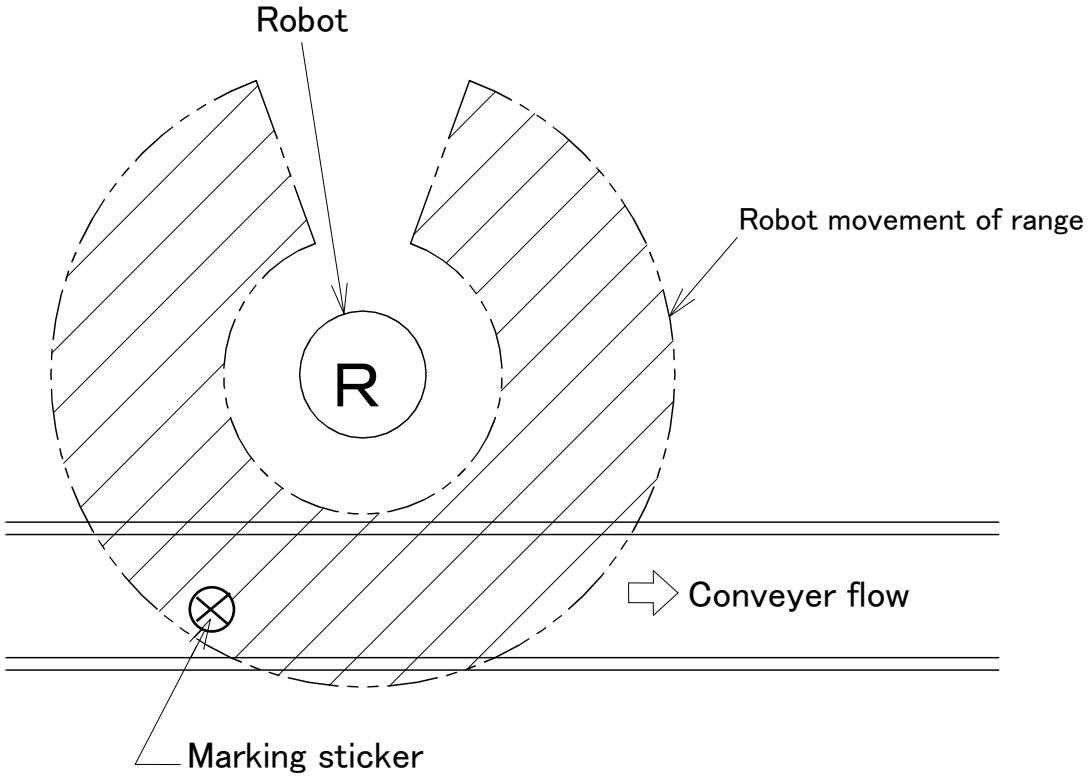
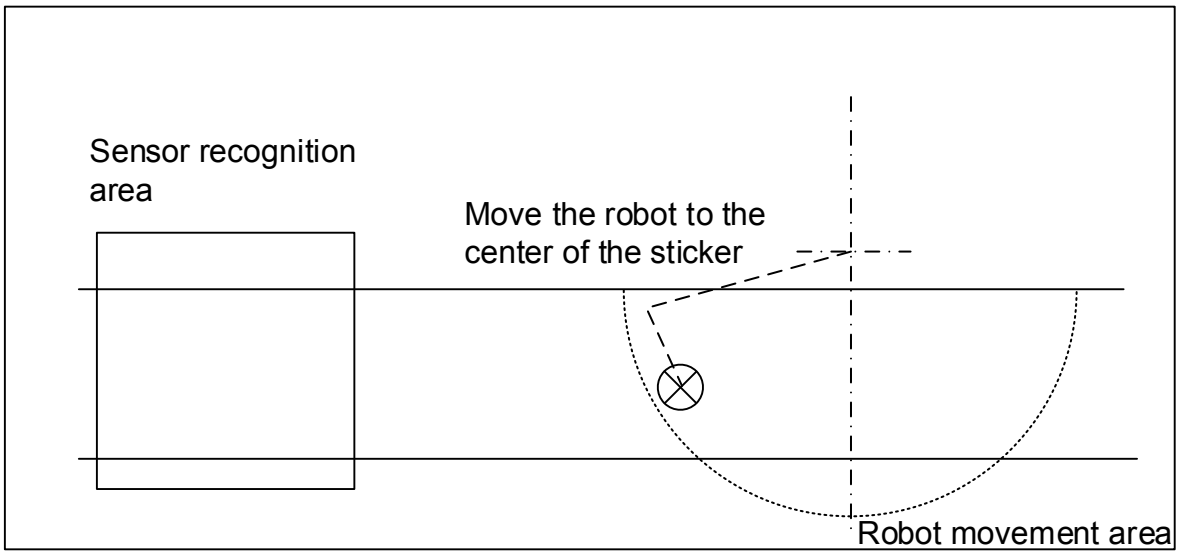


Figure 8-1 Position of Marking Sticker on Conveyer

(8) Press [F1] (FWD) key and execute step feed "(3) Move the robot to the po…"is displayed.
Move the robot to the position right at the center of the marking sticker on the conveyer.
*** With this operation, encoder data and robot position are acquired.**



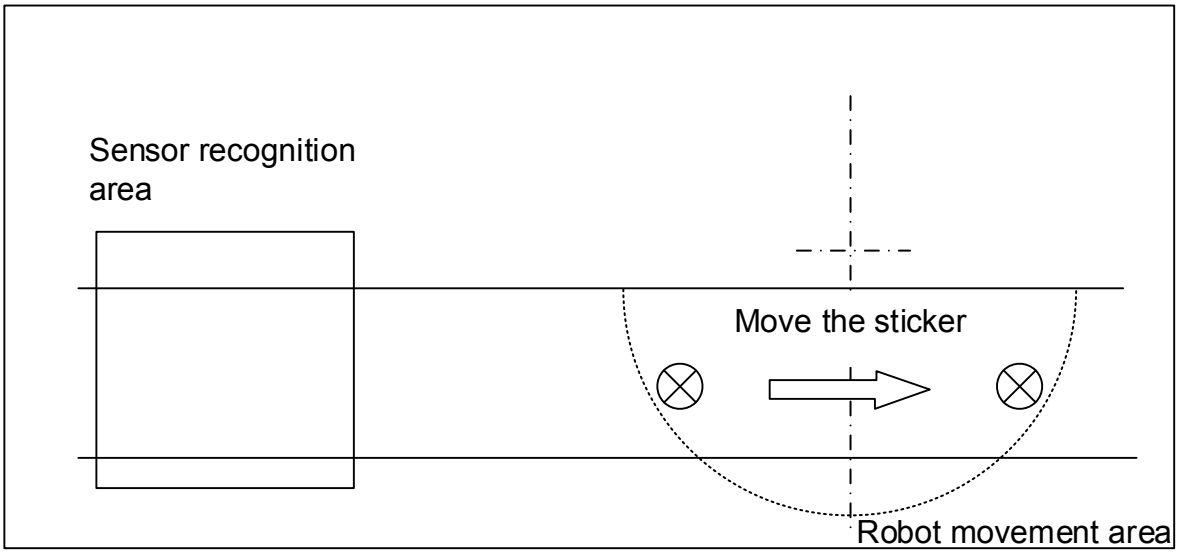
⚠ CAUTION

Move the robot to an accurate position.

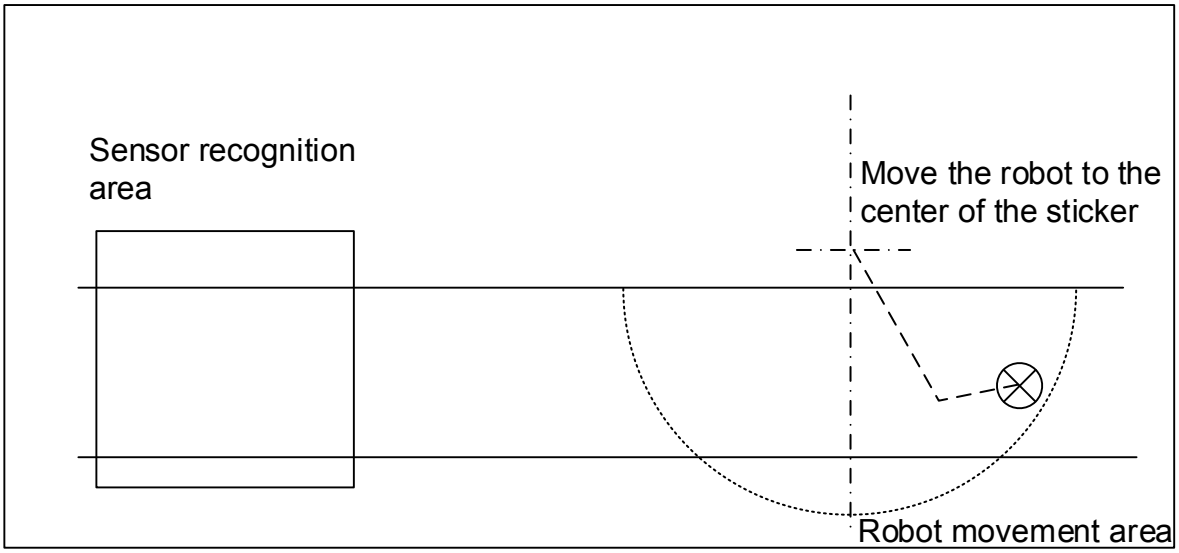
Be sure to move the robot to the position exactly at the center of the marking sticker because the amount of robot movement per encoder pulse is determined by the robot positions specified for the first and second times. Moreover, pay attention to the robot height as well because this amount of movement includes changes of robot position in the Z axis direction.

(9) Press [F1] (FWD) key and execute step feed "(4) Raise the robot" is displayed.
Raise the robot.

(10) Press [F1] (FWD) key and execute step feed "(5) Move the sticker in the…" is displayed.
Drive the conveyer and stop at a position where the marking sticker is immediately outside the robot movement range.



(11) Press [F1] (FWD) key and execute step feed "(6) Move the robot to the posi…" is displayed.
Move the robot to the position right above the center of the marking sticker on the moved conveyer.
*** With this operation, encoder data and robot position are acquired.**



(12) Press [F1] (FWD) key and execute step feed "(7) Raise the robot" is displayed.
Raise the robot.

(13) Press [F1] (FWD) key and execute step feed "(8) Perform step operation…" is displayed.

Perform step operation until "End."

*** The amount of robot movement per encoder pulse is calculated based on this operation.**

8.3. Confirmation after operation

Check the value of "P_EncDlt" using T/B.

*** This value indicates the movement of each coordinate (mm) of the robot coordinate system, corresponding to the movement of the conveyer per pulse.**

Example) If "0.5" is displayed for the Y coordinate only

This means that if the conveyer moves for 100 pulses, the workpiece moves 50 mm (0.5 x 100 = 50) in the +Y direction in the robot coordinate system.

When backing up, the data of "P_EncDlt" is not backed up.

Please work referring to "14.2.5 Restore backup data to another controller" when you restore data to another tracking system.

8.4. When multiple conveyers are used

Carry out the same operations as above when multiple conveyers are used as well, but pay attention to the following points.

Example) When using conveyer 2 (encoder number "2"),

(a) Copy the "A1" program, please create a "A2" program.

(b) Please change the encoder number for variable "MEncNo" in the "A2" program to "2".

9. Calibration of Vision Coordinate and Robot Coordinate Systems (“B1” program)

This chapter explains the tasks carried out by using “B1” program.

*** “B1” program only contains operations required when constructing a vision tracking system.**
These operations are not necessary when constructing a conveyer tracking system.

Calibration of a vision sensor refers to converting the position of a workpiece recognized by the vision sensor to the corresponding position in the robot coordinate system.

This calibration operation is easily performed by the “Mitsubishi robot tool” in In-Sight Explorer. Refer to "Mitsubishi robot tool manual for EasyBuilder" for the details of this function.

“B1” program performs specified tasks and allows acquiring the workpiece coordinates recognized by the vision sensor in the robot coordinate system (position coordinates of robot movement).

The procedures of operations specified by “B1” program and items to be confirmed after the operations are explained below.

This chapter explains on the assumption that "Mitsubishi robot tool" is used.

Please refer to “Detailed Explanations of Functions and Operations” for the steps involved in each operation.

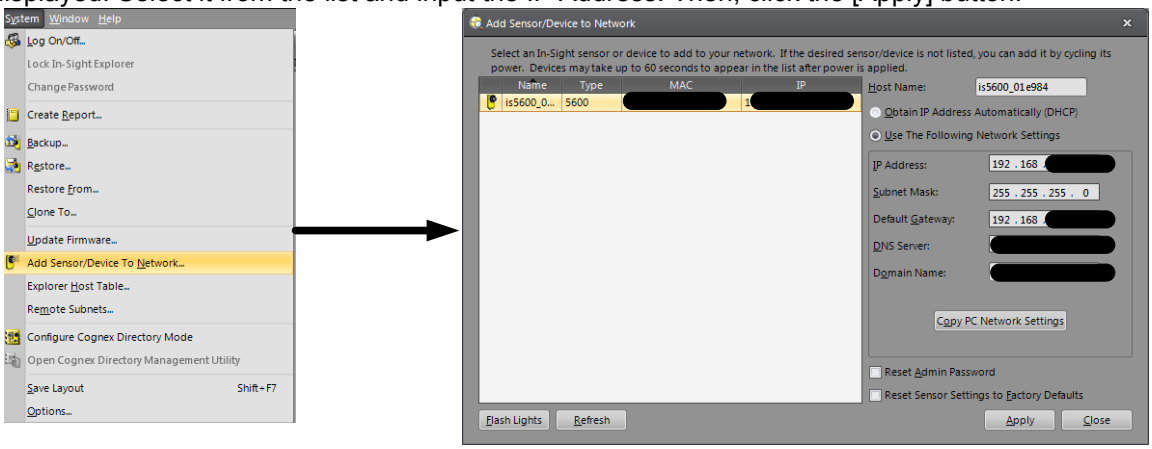
This operation needs a Calibration sheet (Appendix 15.5 Calibration sheet).Print the Calibration sheet in advance.

9.1. Operation procedure

1) Start In-Sight Explorer and set the IP Address of vision sensor.

From the menu of In-Sight Explorer, select [System]-[Add Sensor/Device To Network…].

In the “Add Sensor/Device To Network” screen, the sensor or device which can add to the network is displayed. Select it from the list and input the IP Address. Then, click the [Apply] button.



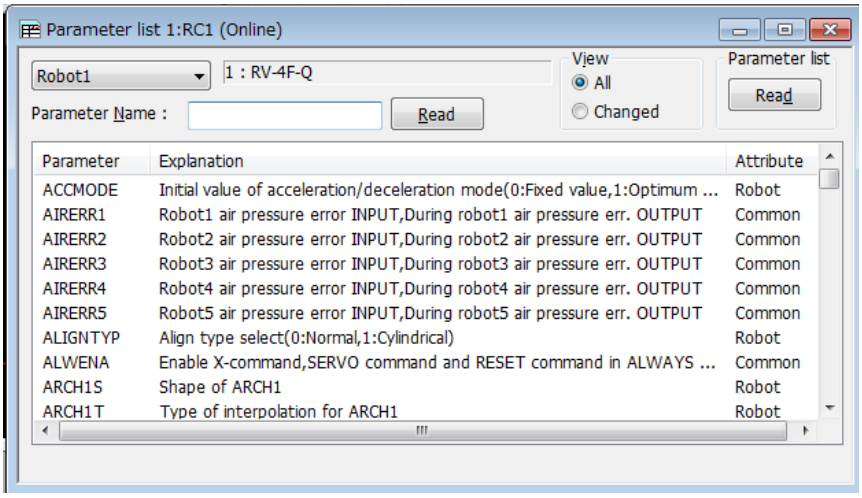
2) To communicate the Mitsubishi robot tool and the vision sensor, set a necessary parameter by using RT ToolBox2.

A necessary parameter is three (“NETIP", "Element 9 of NETTERM", and “CTERME19").

In RT ToolBox2, select [Online]-[parameter]-[parameter list].

Input the following parameters to "Parameter Name" of the displayed "Parameter list" screen and change a “Setting value”.

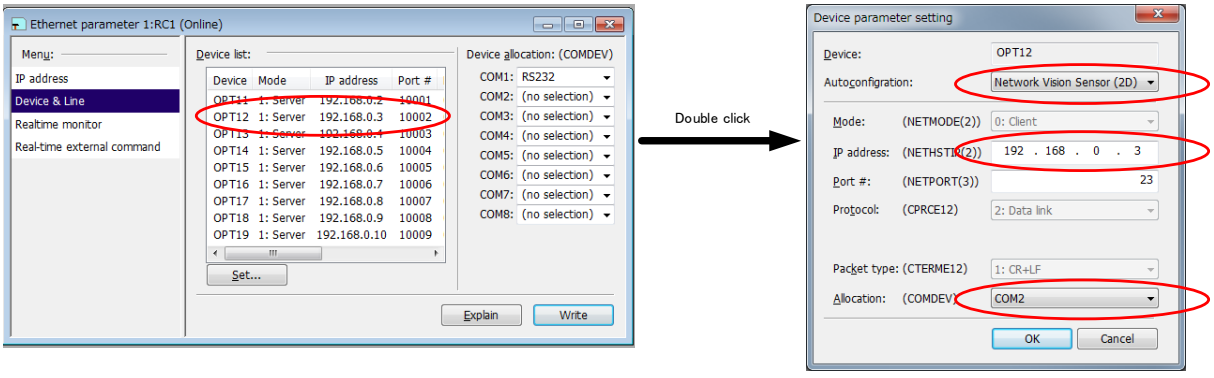
Parameter Name	Initial value	Setting value	Explanation
NETIP	Q type：192.168.100.1 D type：192.168.0.20	xxx.xxx.xxx.xxx	IP address of robot controller
NETTERM(Element 9)	0	1	The end code is added with communication.
CTERME19	0	1	The end code of port 10009 is changed to “CR+LF”.



Please confirm whether the following parameters are initial values.

Parameter Name	Initial value	Explanation
NETPORT(Element 10)	10009	Port number allocated to device OPT19
CPRCE19	0	The protocol used is “Non-procedure”
NETMODE(Element 9)	1	Opens as “Server”.

In RT ToolBox2, select [Online]-[Parameter]-[Communication parameter]-[Ethernet].
“OPT12” is selected “COM2” that exists in “Device & Line” column on the displayed “Ethernet parameter” screen. Double-click “OPT12” that exists in “Device list”.
Select “Network Vision Sensor (2D)” from “Autoconfigration”, input IP address of the vision sensor to “IP address” and select “COM2” from “Allocation” column. Click [OK] button. And, click [Write] button on “Ethernet parameter” screen.



Turn on robot controller's power supply again to make the set parameter effective.

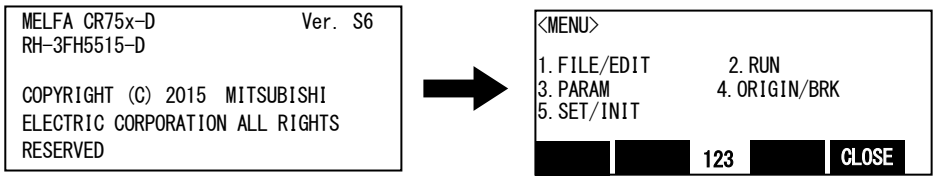
(1) Open “B1” program using T/B.
Set the controller mode to "MANUAL". Set the T/B to "ENABLE".

O/P T/B

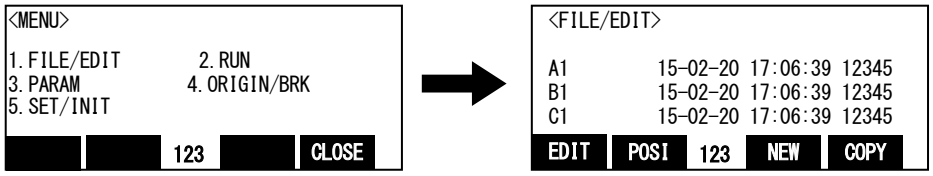
Up : DISABLE
Down : ENABLE
★Lamp lighting

T/B rear

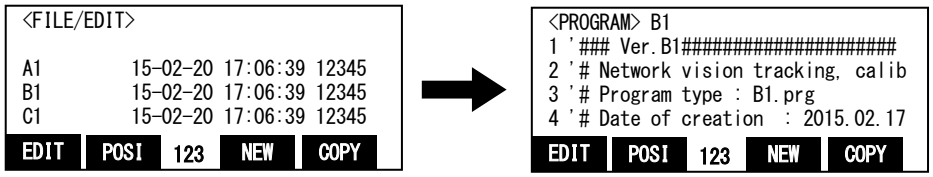
(2) Press one of the keys (example, [EXE] key) while the <TITLE> screen is displayed. The <MENU> screen will appear.



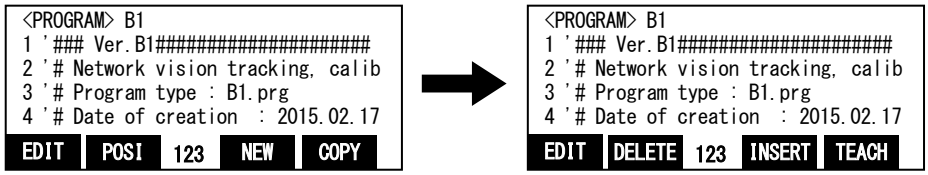
(3) Select "1. FILE /EDIT" screen on the <MENU > screen.



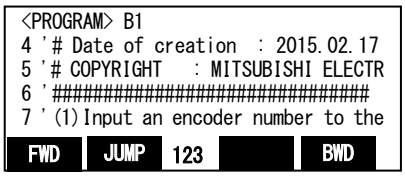
(4) Press the arrow key, combine the cursor with the program name "B1" and press the [EXE] key. Display the <program edit> screen.



(5) Press the [FUNCTION] key, and change the function display

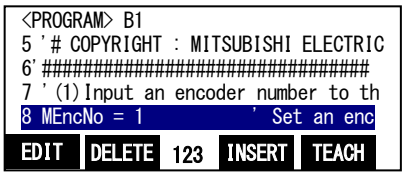


(6) Press the [F1] (FWD) key and execute step feed. "(1) Input an encoder … "is displayed. Execute work according to the comment in the robot program.

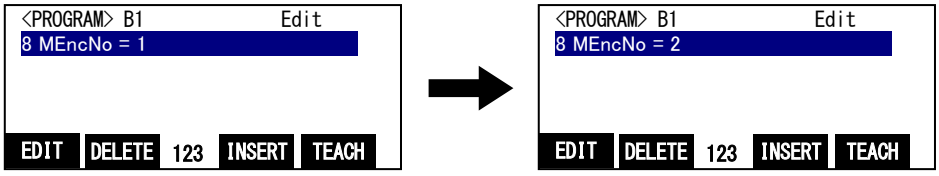


Specify the encoder number.
If you want to change the encoder number, please edit the program as follows.

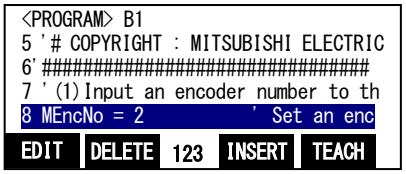
(a) Display the following command.



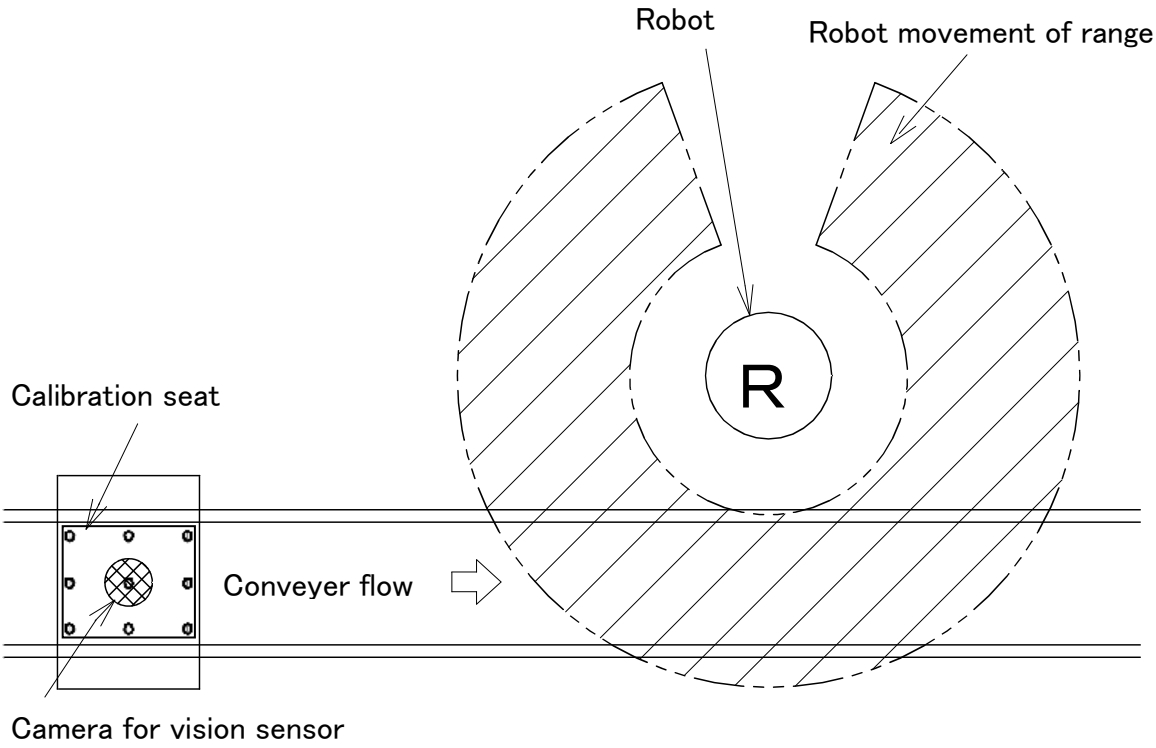
(b) Press the [F1](FWD) key and specify the encoder number in the variable "MEncNo"
Example)When "2" is specified as the encoder number.



(c) Press the [F1](FWD) key and the change is determined.



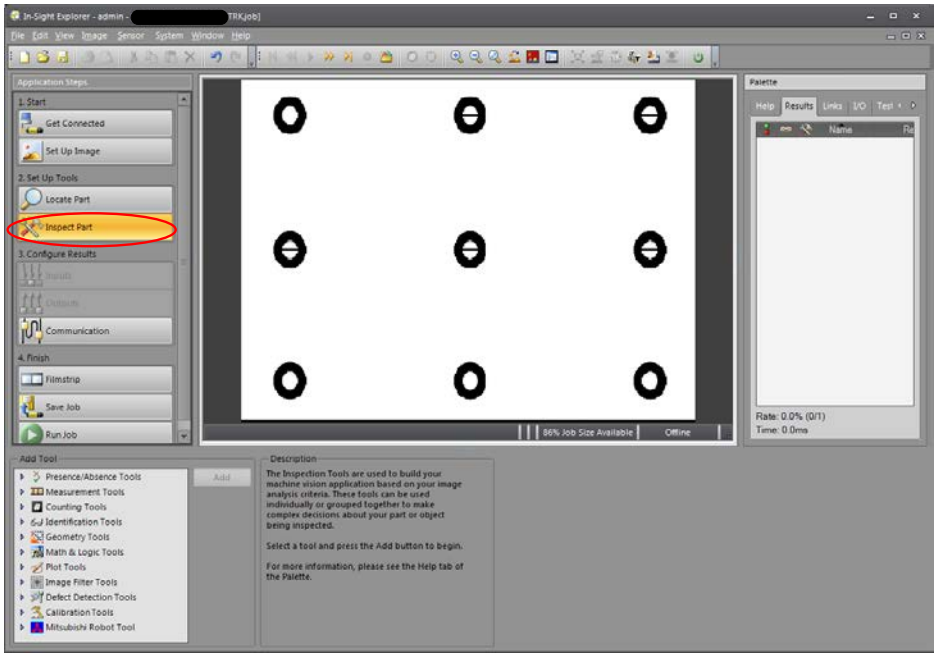
(7) Press the [F1] (FWD) key and execute step feed. "(2)Place the calibration sheet…"is displayed.
Paste appendix calibration seat to "15.5 Calibration sheet" on the conveyer.



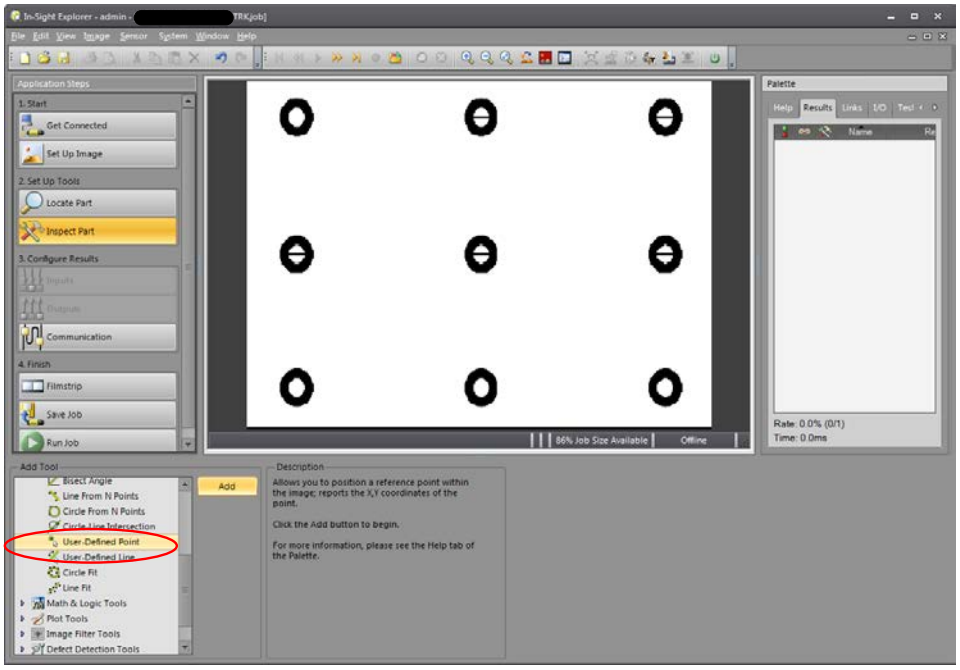
(8) Press the [F1] (FWD) key and execute step feed. "(3)Check that the calibration…"is displayed.
Paste calibration seat within the field of vision checking the live images of In-Sight Explorer.
*** With this operation, encoder data is acquired.**

(9) Press the [F1] (FWD) key and execute step feed. "(4)Specify the mark in three…"is displayed.
Specify the mark in three points or more by using "Mitsubishi Robot Tool" on "In-Sight Explorer"

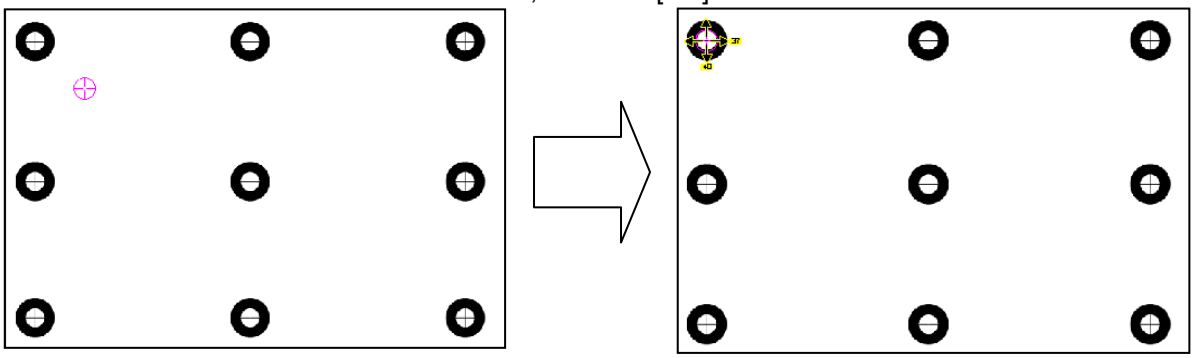
1) End [Live Video] of In-Sight Explorer, and select [Inspect Part] button of "Application Steps".



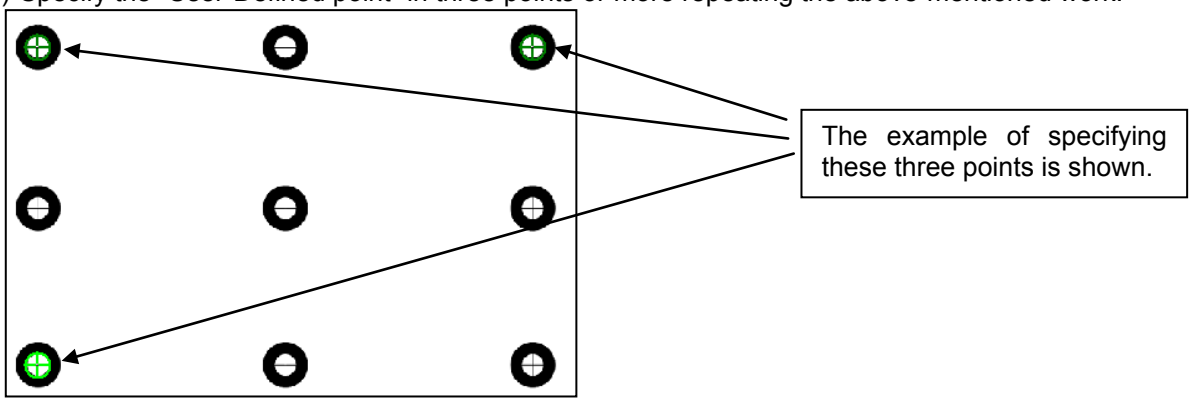
2) Select [Geometry Tools] - [User-Defined Point] in "Add tool".



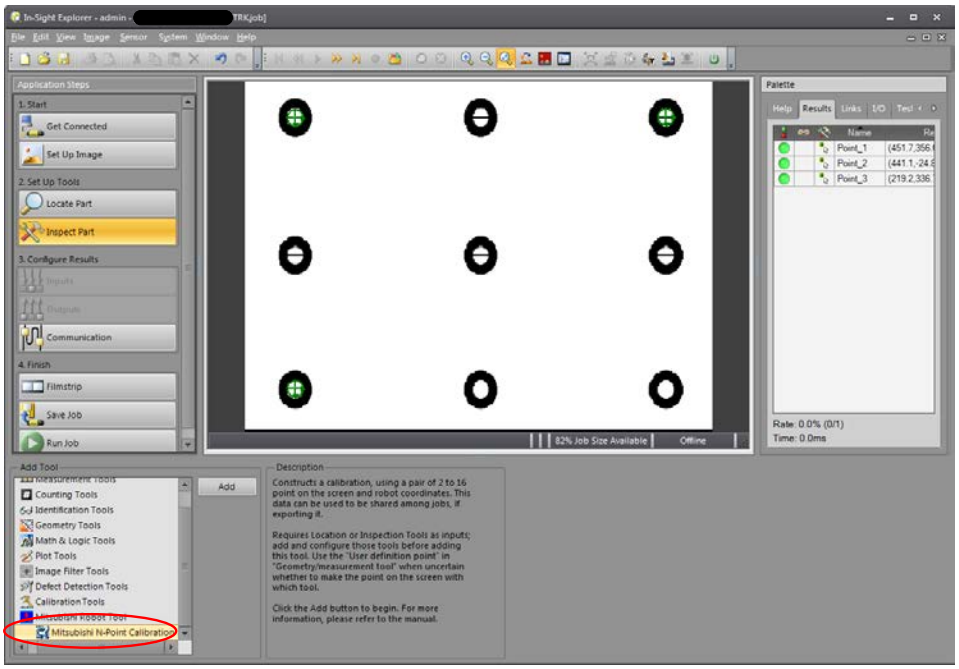
3) Click [Add] button. Then, the cross sign enclosed with circle on the screen is displayed.
Move it to the mark of the calibration seat, and click [OK] button.



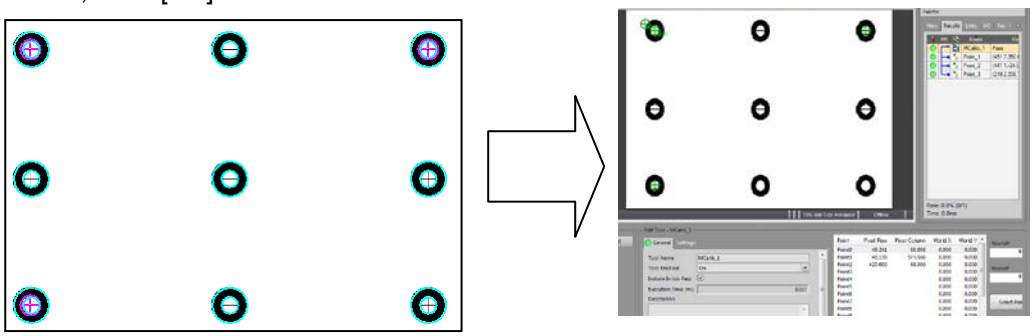
4) Specify the "User-Defined point" in three points or more repeating the above-mentioned work.



5) Select [Mitsubishi Robot Tool] – [Mitsubishi N-point calibration] in "Add Tool" column of this tool.



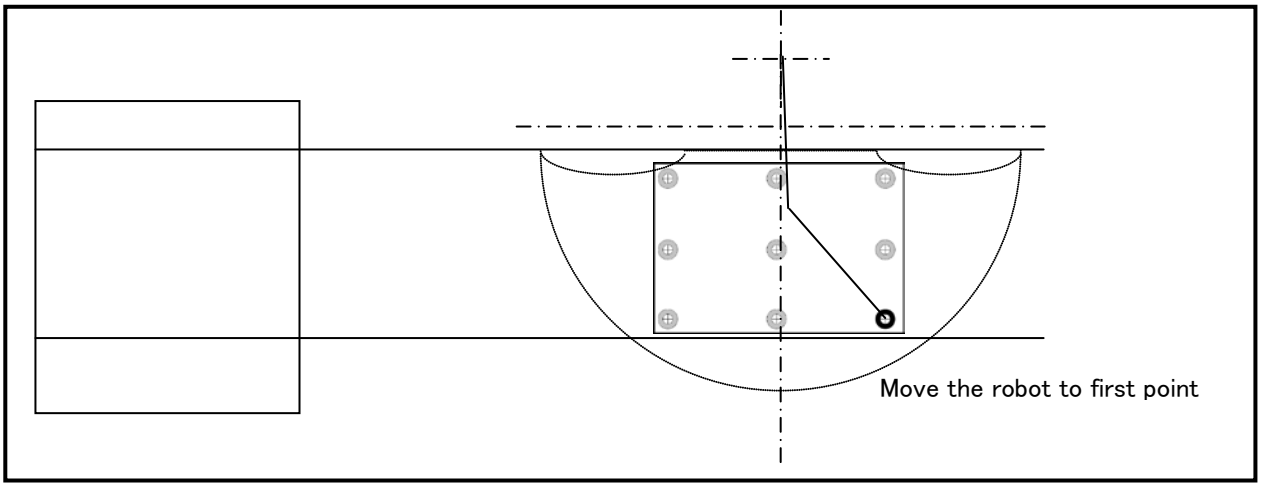
6) Click [Add] button. Select "User-Defined point" three points specified ahead from nine displayed marks. Then, Click [OK] button.



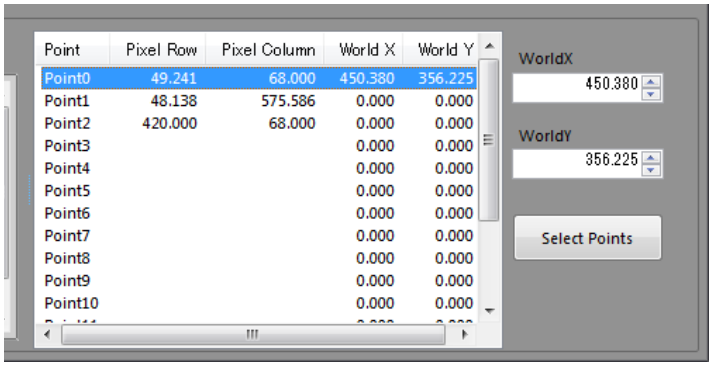
7) Open the [Settings] tab screen from the "Edit Tool", and input IP address set to "Robot IP address".

(10) Press the [F1] (FWD) key and execute step feed. "(5)Move the calibration sheet…"is displayed. Move the calibration seat by starting the conveyer within the robot movement range.

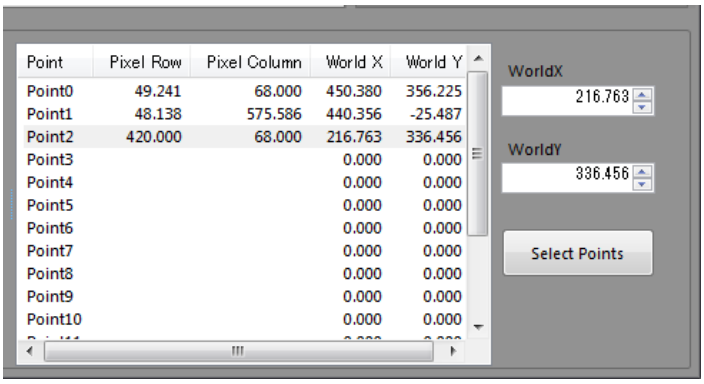
(11) Press the [F1] (FWD) key and execute step feed. “(6)Move the robot hand to the…”is displayed.
Move the robot to the position right above the first mark on the conveyer.



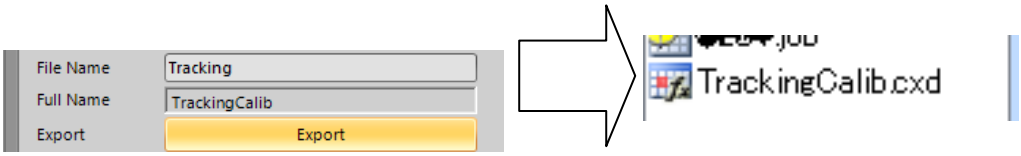
(12) Press the [F1] (FWD) key and execute step feed. “(7) Acquire the robot present…”is displayed.
Click [Get position] button in “Edit Tool” column of In-Sight Explorer.
Confirm the current position of the robot was displayed in [world X] and [world Y].



(13) Press the [F1] (FWD) key and execute step feed. “(8) Acquire the position of the…”is displayed.
Similarly, move the robot hand to the mark of the second point and the third point, and acquire the current position of the robot with [Get position] button of In-Sight Explorer.



(14) Press the [F1] (FWD) key and execute step feed. “(9) Click the Export button. Then…”is displayed.
Input an arbitrary name to "File name" in the tool edit column of In-Sight Explorer, and click the export button. (In this example, File Name is “Tracking”) And, confirm the calibration file of the specified name was made in the vision sensor.



(15) Press the [F1] (FWD) key and execute step feed. "(10) Raise the robot arm" is displayed.
Raise the robot.
＊ With this operation, encoder data is acquired.

9.2. Confirmation after operation

Check the value of "M_100()" using T/B.
Enter the **encoder number** in the array element.
Confirm that the differences between the encoder values acquired on the vision sensor side and the encoder values acquired on the robot side are set in "M_100()."

9.3. When multiple conveyers are used

Carry out the same operations as above when multiple conveyers are used as well, but pay attention to the following points.

Example） When using conveyer 2 (encoder number "2"),
(a) Copy the "B1" program, please create a "B2" program.
(b) Please change the encoder number for variable "MEncNo" in the "B2" program to "2".

10. Workpiece Recognition and Teaching ("C1" program)

This chapter explains the tasks carried out by using "C1" program.

*** "C1" program contains operations required for both conveyer tracking and vision tracking, but different operations are performed. Refers to "10.1 Conveyer Tracking" for operations in the case of conveyer tracking and "10.2 Vision Tracking" for operations in the case of vision tracking.**

Please refer to "Detailed Explanations of Functions and Operations" for the steps involved in each operation.

10.1. Conveyer Tracking

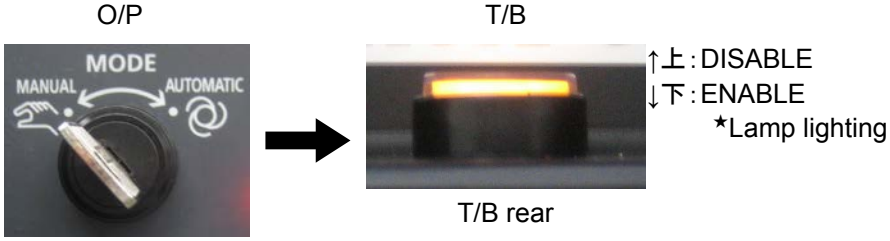
In "C1" program for conveyer tracking, encoder data at the positions where a sensor is activated and where the robot suctions a workpiece is acquired so that the robot can recognize the workpiece coordinates when the sensor is activated at later times.

The operation procedure and items to be confirmed after operation in "C1" program for conveyer tracking are explained below.

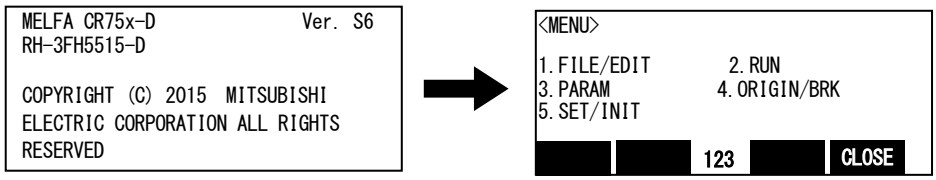
10.1.1. Operation procedure

Using "C1" program, operate in the following procedures.

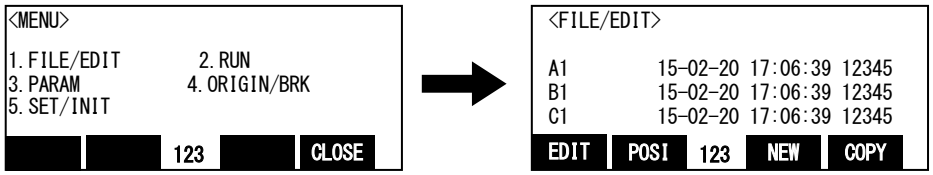
(1) Set the controller mode to "MANUAL". Set the T/B to "ENABLE".



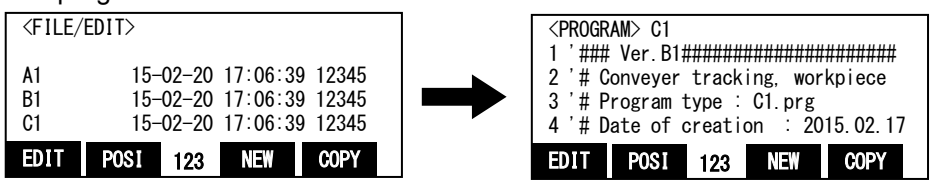
(2) Press one of the keys (example, [EXE] key) while the <TITLE> screen is displayed. The <MENU> screen will appear.



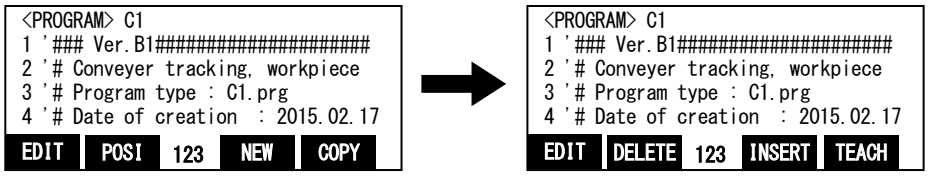
(3) Select "1. FILE /EDIT" screen on the <MENU > screen.



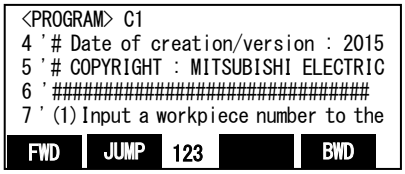
(4) Press the arrow key, combine the cursor with the program name "C1" and press the [EXE] key. Display the <program edit> screen.



(5) Press the [FUNCTION] key, and change the function display

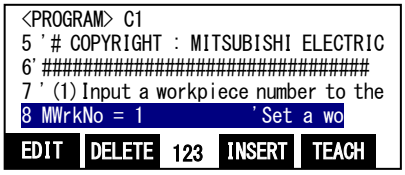


(6) Press the [F1] (FWD) key and execute step feed. "(1) Input a workpiece … "is displayed. Execute work according to the comment in the robot program.

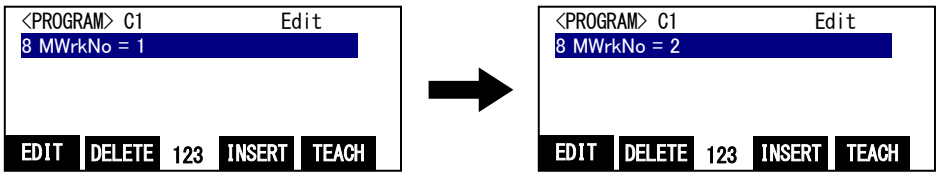


Specify the workpiece number.
If you want to change the workpiece number, please edit the program as follows.

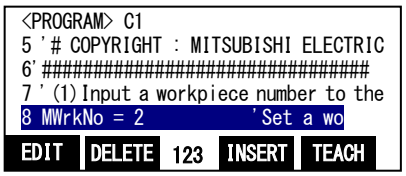
(a) Display the following command.



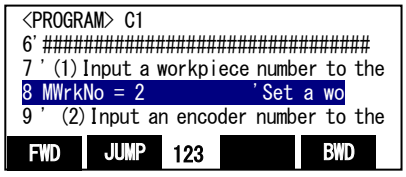
(b) Press the [F1](FWD) key and specify the workpiece number in the variable "MWrkNo"
Example)When "2" is specified as the workpiece number.



(c) Press the [F1] (FWD) key and the change is determined.



(7) Press the [F1] (FWD) key and execute step feed. "(2)Input an encoder number…"is displayed.



Specify the workpiece number.
If you want to change the workpiece number, please edit the program as similar procedure (6).

(8) Press the [F1] (FWD) key and execute step feed. "(3)Input the number of the sensor…"is displayed.

```
<PROGRAM> C1
8 MWrkNo = 1      'Set a workpiece
9 '(2)Input an encoder number to the
10 MEncNo = 1     'Set an encoder
11 '(3)Input the number of the sensor
```

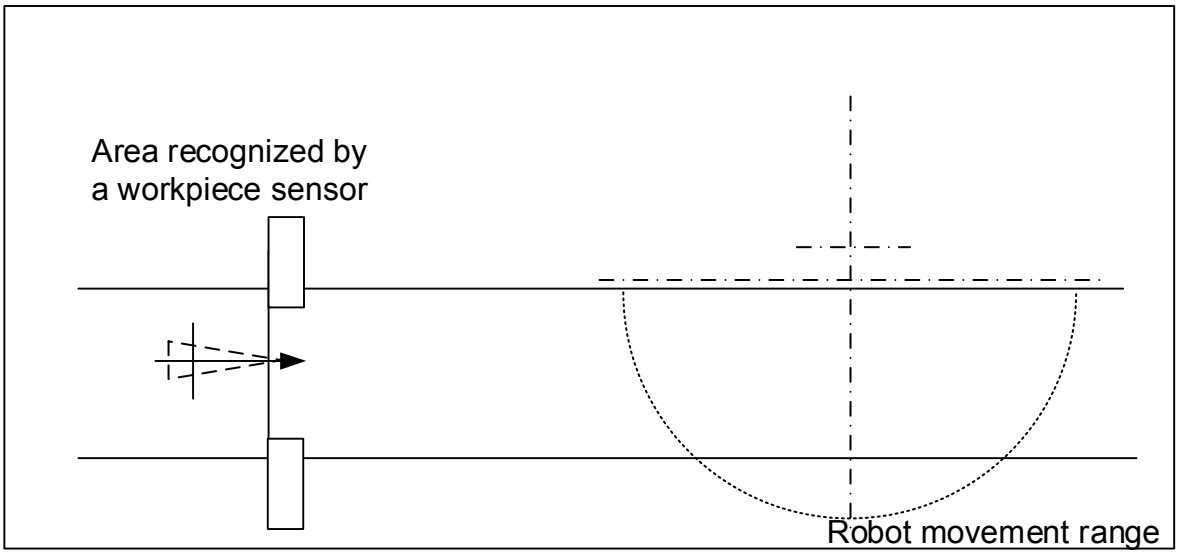
FWD | JUMP | 123 | | BWD

Specify the number of the sensor that monitors workpiece.
If you want to change the sensor number, please edit the program as similar procedure (6).

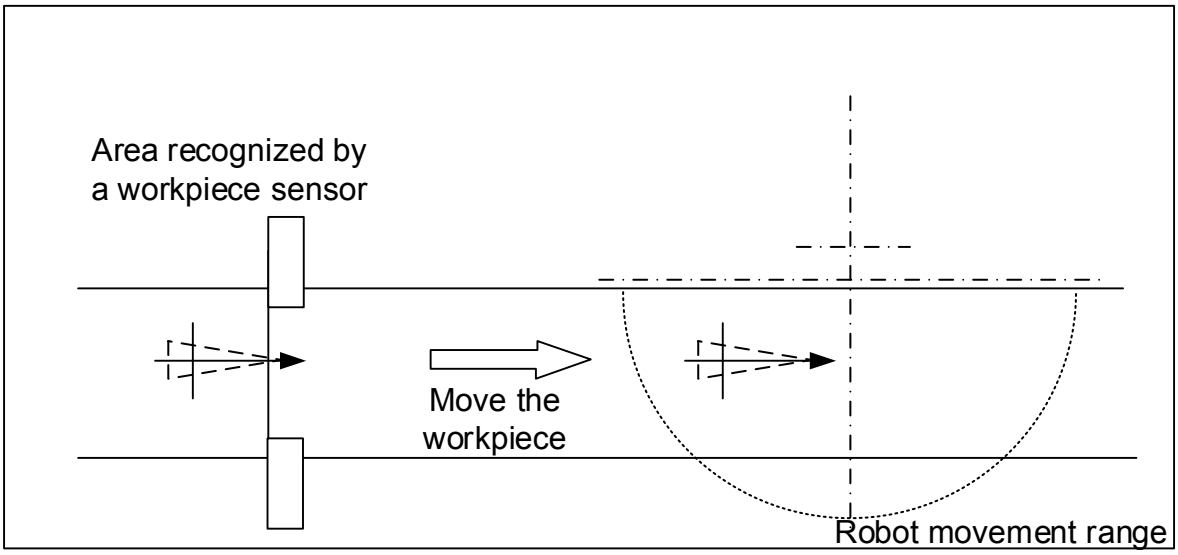
[Q type]
Tracking enable signal number is 810.

[D type]
Input signal number is 16.

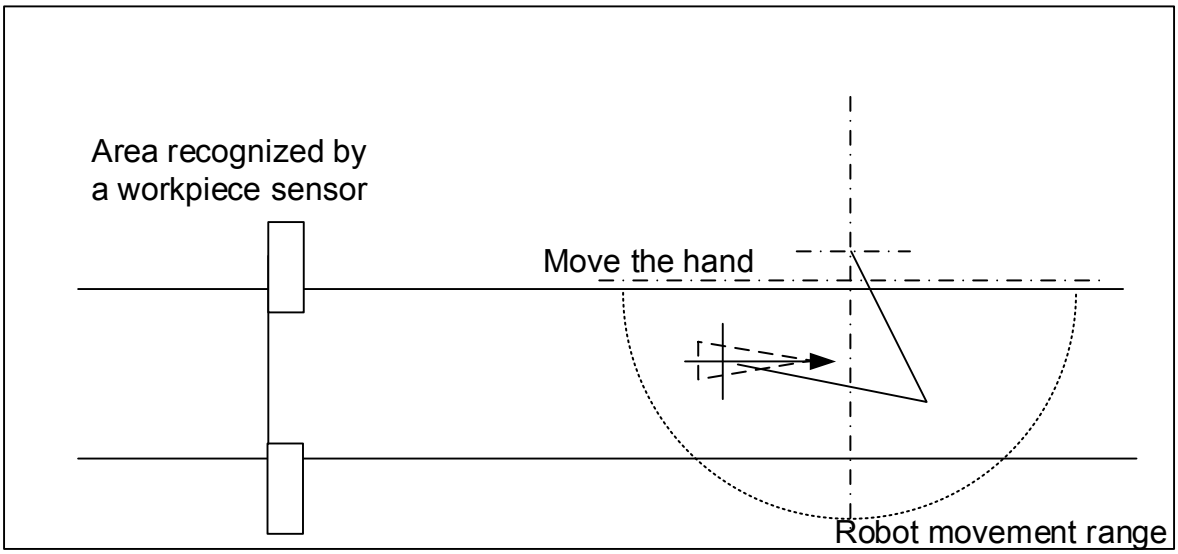
(9) Press the [F1] (FWD) key and execute step feed. "(4)Move a workpiece to the position…"is displayed.
Move a workpiece to the location where the sensor is activated.
*** With this operation, encoder data is acquired.**



(10) Press the [F1] (FWD) key and execute step feed. "(5)Move a workpiece on the conveyer…"is displayed.
Drive the conveyer to move the workpiece within the robot movement range.



(11) Press the [F1] (FWD) key and execute step feed. "(6)Move the robot to the suction position…"is displayed.
Move the robot to the position where it suctions the workpiece.
***With this operation, encoder data and robot position are acquired.**



1) Perform step operation until "End."
*** With this operation, the robot is able to calculate the position of a workpiece as soon as the sensor is activated.**

10.1.2. Confirmation after operation

Confirm the values of "M_101()," "P_100()" and "P_102()" using T/B.
Enter **encoder numbers** in array elements.

- "M_101()": Differences between the encoder values acquired at the position of the photoelectronic sensor and the encoder values acquired on the robot side.
- "P_100()": Position at which workpieces are suctioned
- "P_102()": The value of the variable "MEncNo" and "MSenNo"

Check that each of the values above has been entered correctly.

10.1.3. When multiple conveyers are used

Carry out the same operations as above when multiple conveyers are used as well, but pay attention to the following points.

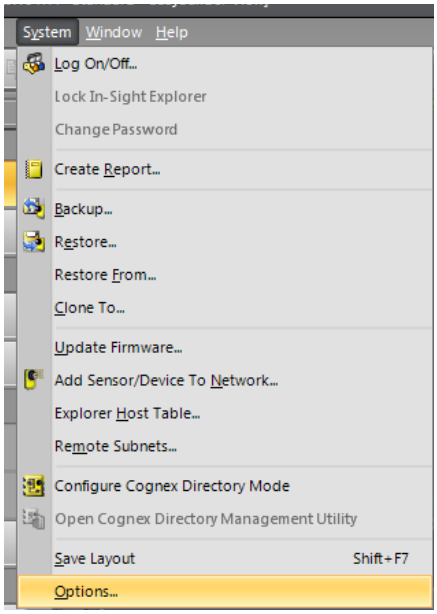
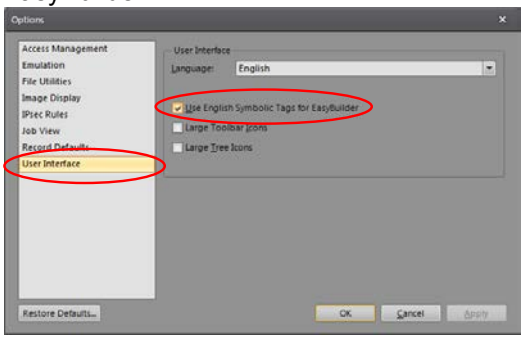
Example) When using conveyer 2 (encoder number "2"), kind number "2",
(a) Copy the "C1" program, please create a "C2" program.
(b) Please change the kind number for variable "MWrkNo" in the "C2" program to "2".
(c) Please change the encoder number for variable "MEncNo" in the "C2" program to "2".

10. 2. Vision Tracking

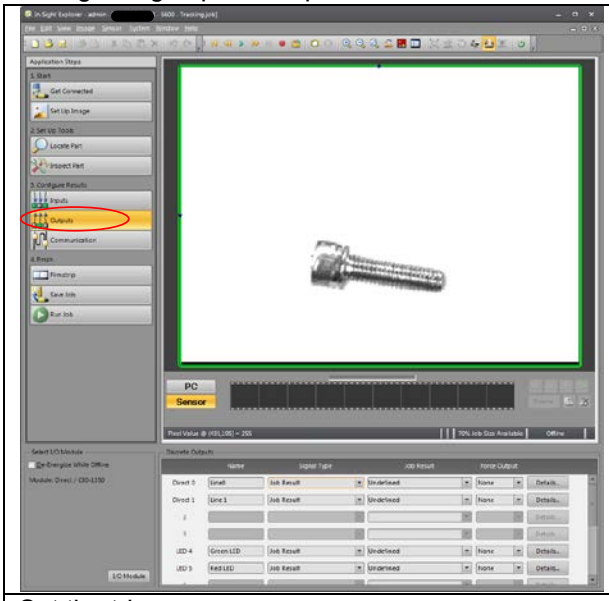
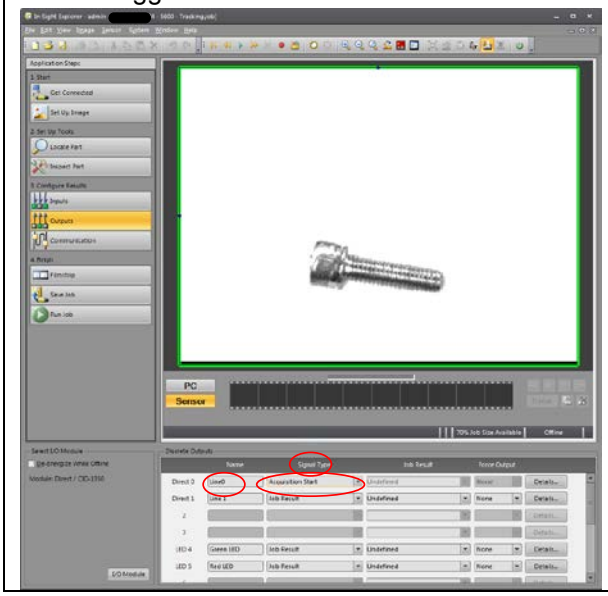
Vision tracking "C1" program acquires encoder data at the position where the vision sensor recognizes workpieces and where the robot suctions workpieces such that the robot can recognize the work coordinates recognized by the vision sensor. The following explains the operation procedure and items to confirm after operation in vision tracking "C1" program.

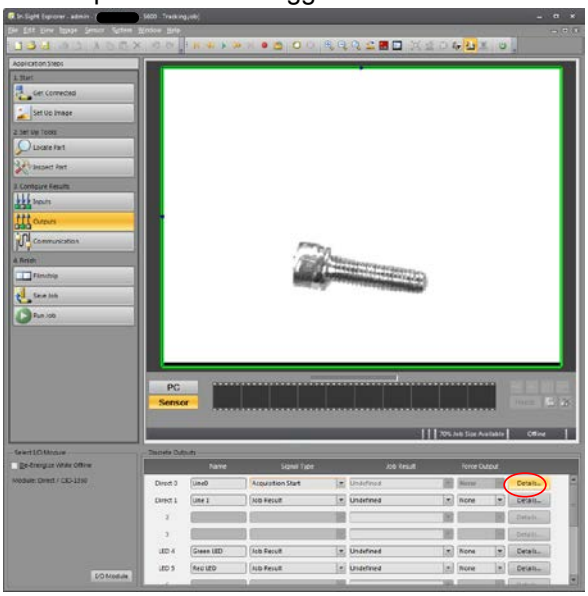
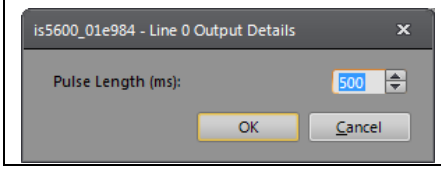
10. 2. 1. Tasks

(1) Setting of the English Symbolic tag

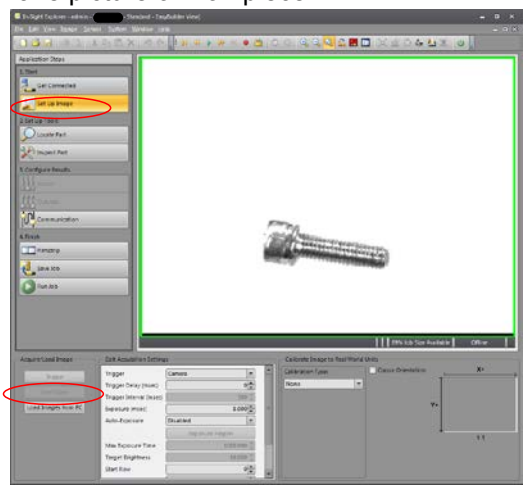
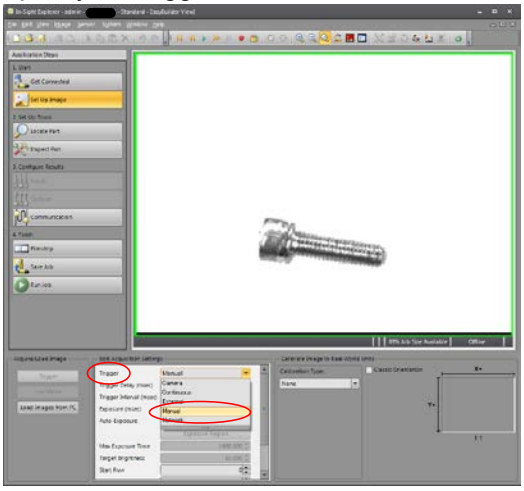
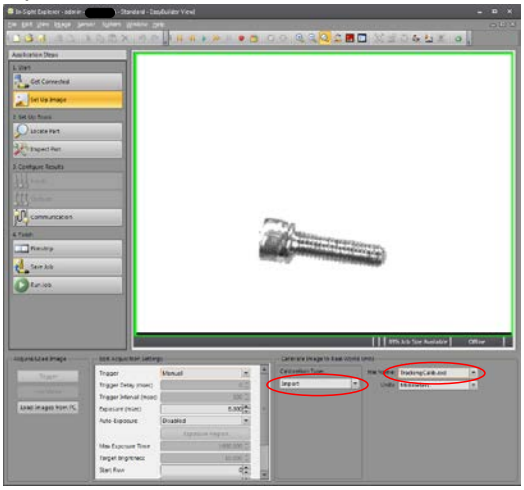
Set up using the English Symbolic tag. 	Select [System]-[Option] from the EasyBuilder menu.
Check the "Use English Symbolic Tags for EasyBuilder" . 	Select [User Interface] from [option], check the "Use English Symbolic Tags for EasyBuilder" and click the "OK" button.

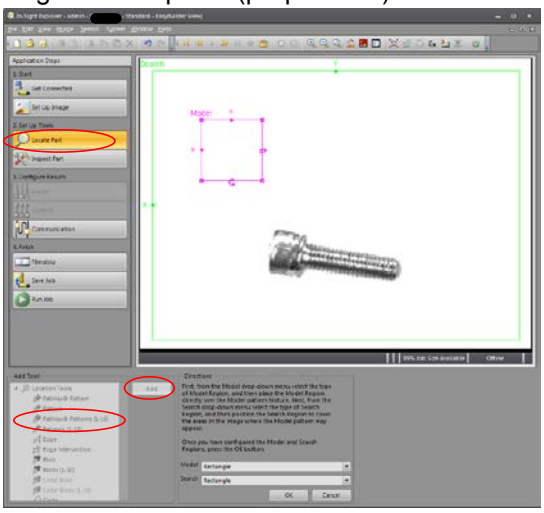
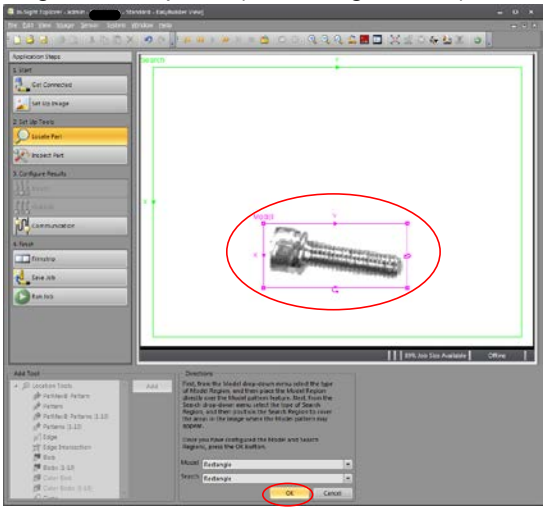
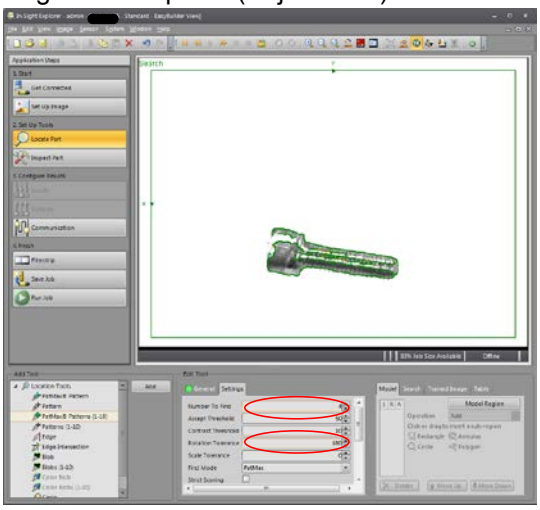
(2) Setting of high speed output.

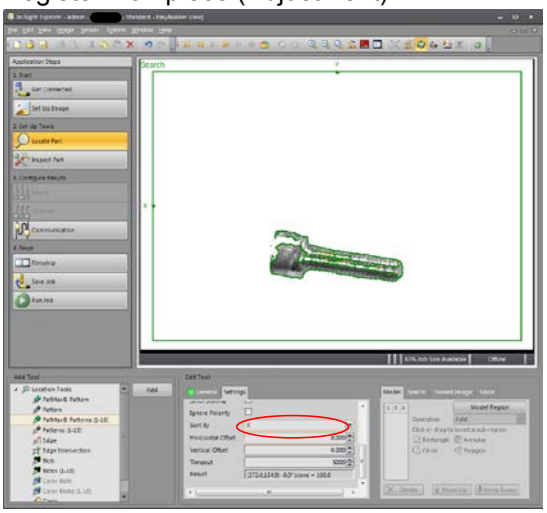
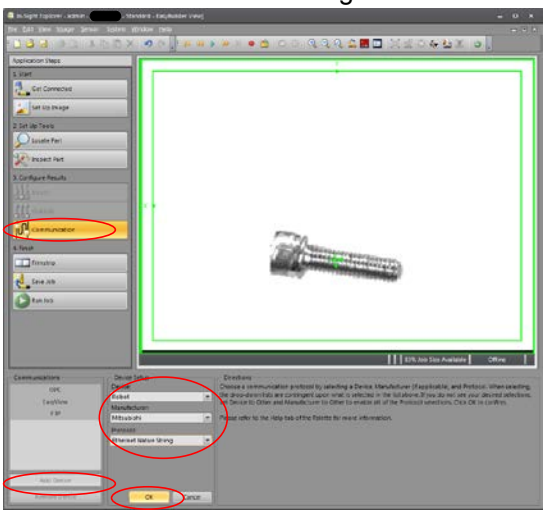
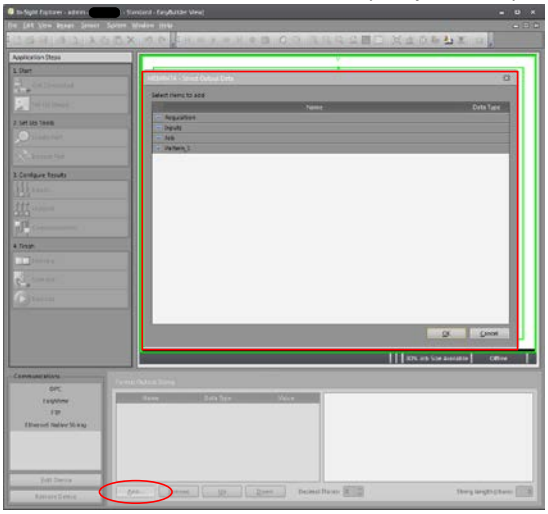
	Make the vision sensor offline. Click [Output] from “Application Steps”.
Set the trigger. 	Select “Acquisition start” from [Signal type] of [Discrete output].

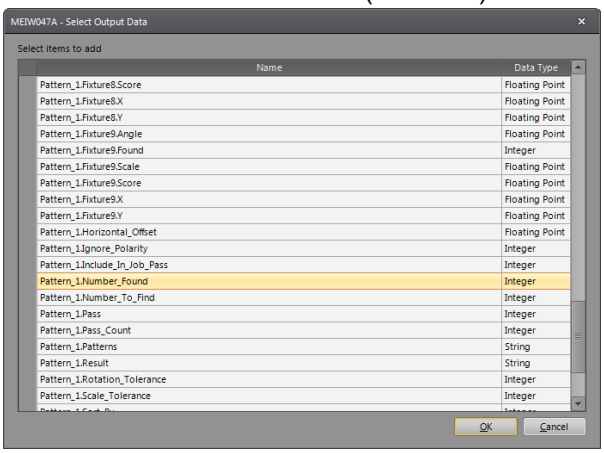
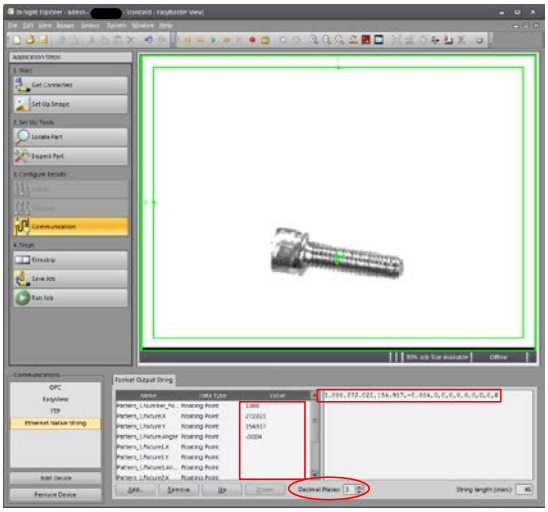
Set the pulse width of trigger.  	Click [Detalls] from [Discrete Outputs]. The optional value (Initial value:500)is set as a pulse width. Ckick [OK].

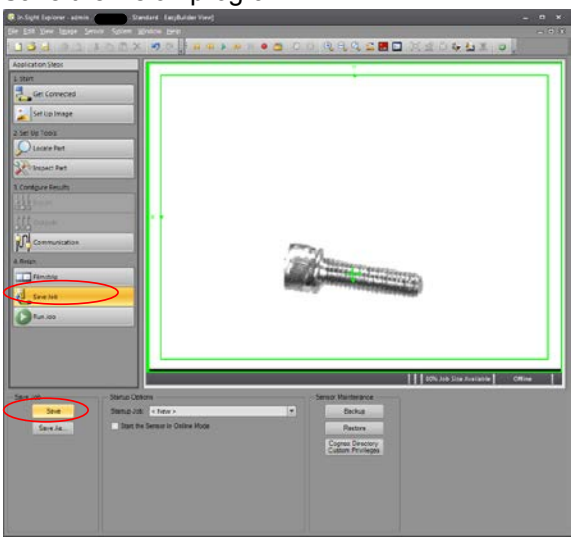
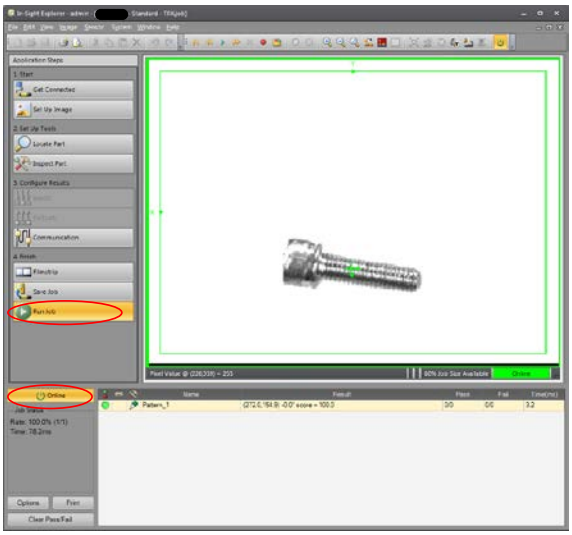
(3) Make the vision program.

Take picture of workpiece. 	Select [File] – [New Job] from the menu. Click [Set Up Image] button from “Application Steps”. Click [Live Video] button. Take picture of workpiece that does the tracking. Again, stop a live image clicking [Live Video] button.
Specify the trigger. 	**Change [Trigger] from "Camera" to "Manual”.** 8640(The image trigger is abnormal) error occurs when the robot controller outputs the taking picture demand to the vision sensor when you do not change.
Inport the calibration data. 	In [Calibration type], select "Import". In [File Name], select the Calibration file (For example,“TrackingCalib.cxd") registered when working about the B1 program.

Register workpiece (preparation) 	Click [Locate Part] from “Application Steps”. Select "PatMax Pattern(1-10)" from “Add Tool”, and click [Add] button.
Register workpiece (Model registration). 	Move the displayed "Model" frame, and enclose workpiece. Click [OK] button in “Directions”.
Register workpiece (Adjustment) 	Click [Settings] tab from "Edit Tool", and change the [Rotation Tolerance] value to “180". (The vision sensor can recognize workpiece up to ±180 degrees.) Change the [Accept Threshold], and adjust the recognition rate of workpiece. The default [Accept Threshold] is “50”. At this stage it is enough as it is. And change [Number To Find]“1”to“4”

Register workpiece (Adjustment) 	Change [Sort By] “X” or “Y”. If you sort the recognized multiple workpieces to the right direction of screen, select “X”. If you sort the recognized multiple workpieces to the left direction of screen, select “Y”.
Do the communication setting. 	Click [Communication] from “Application Steps”. Click [Add Device] from "Communications". Select the following from "Device Setup". [Device:] "Robot" [Manufacturer:] "Mitsubishi” [Protocol:] "Ethernet Native String" Click [OK] button.
Set the communication format (Preparation) 	Click [Add] button from "Format Output String". -> "Select Output Data" screen opens.

Set the communication format (selection)  	Click [+] sign of “Pattern_1”, and select it in the following order while pushing the [Ctrl] key. (1) Pattern_1.Number_Found (2) Pattern_1.Fixture.X (3) Pattern_1.Fixture.Y (4) Pattern_1.Fixture.Angle (5) Pattern_1.Fixture1.X (6) Pattern_1.Fixture1.Y (7) Pattern_1.Fixture1.Angle (8) Pattern_1.Fixture2.X (9) Pattern_1.Fixture2.Y (10) Pattern_1.Fixture2.Angle (11) Pattern_1.Fixture3.X (12) Pattern_1.Fixture3.Y (13) Pattern_1.Fixture3.Angle Click [OK] button.
Confirmation of communication format. 	Confirm the value enclosed with a square frame. Data sent to the robot controller is shown in a right square frame. Change the value of [Decimal Places], and change the number of decimal positions of transmitted data.

Save the vision program. 	Click [Save Job] from "Application Steps". Click [Save] from "Save Job". Make the name of the job that saves it **"TRK"**. Change the line of "CPRG$=" C1 program when not assuming "TRK".
Make it to online. 	Click [Run Job] from "Application Steps". Click [Online] on "Job Status".

10. 2. 2. Operation procedure

Using "C1" program, operate in the following procedures.

(1) Set the controller mode to "MANUAL". Set the T/B to "ENABLE".

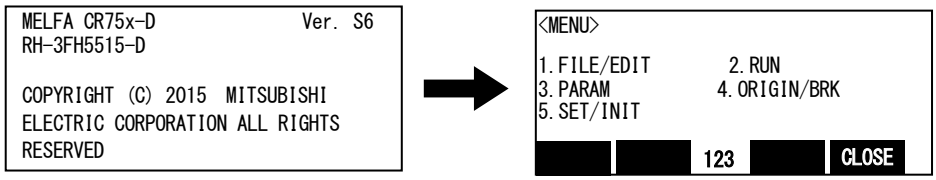
O/P

T/B

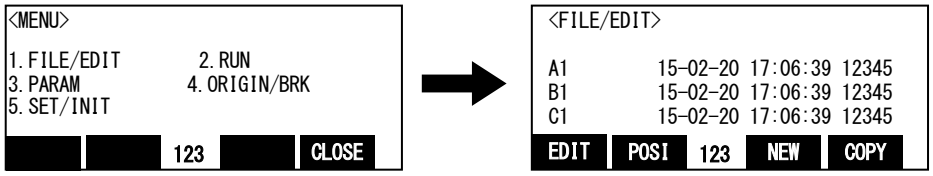
↑上:DISABLE
↓下:ENABLE
★Lamp lighting

T/B rear

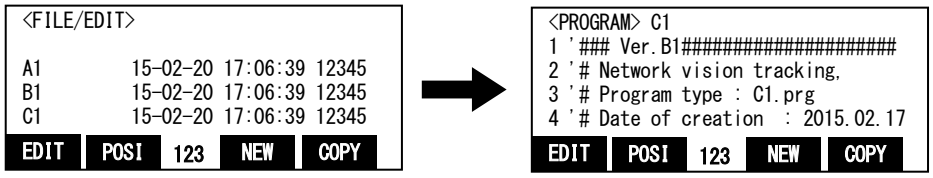
(2) Press one of the keys (example, [EXE] key) while the <TITLE> screen is displayed. The <MENU> screen will appear.



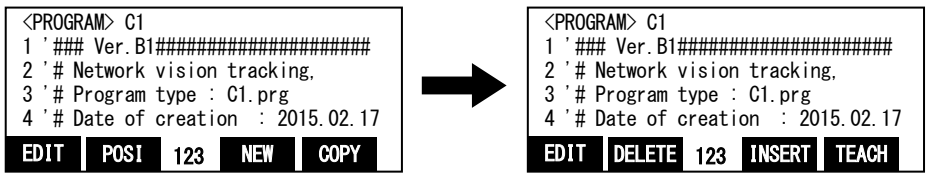
(3) Select "1. FILE /EDIT" screen on the <MENU > screen.



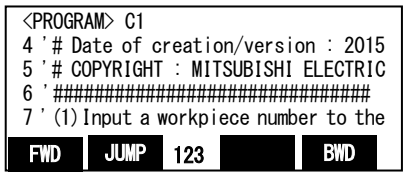
(4) Press the arrow key, combine the cursor with the program name "C1" and press the [EXE] key. Display the <program edit> screen.



(5) Press the [FUNCTION] key, and change the function display

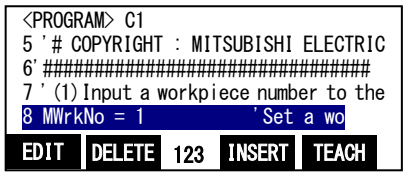


(6) Press the [F1] (FWD) key and execute step feed. “(1) Input a workpiece … “is displayed. Execute work according to the comment in the robot program.

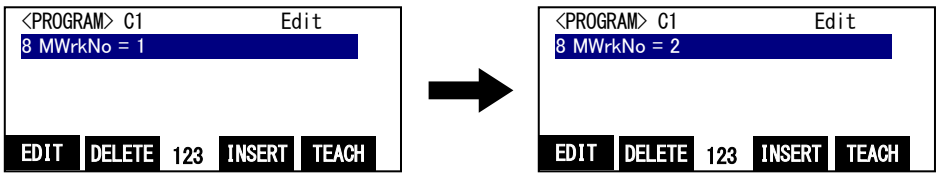


Specify the workpiece number.
If you want to change the workpiece number, please edit the program as follows.

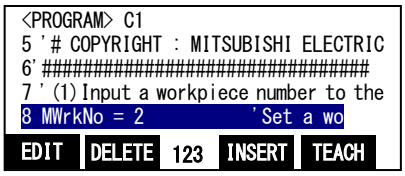
(a) Display the following command.



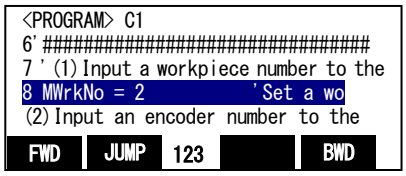
(b) Press the [F1] (FWD) key and specify the workpiece number in the variable “MWrkNo”
Example）When “2” is specified as the workpiece number.



(c) Press the [F1] (FWD) key and the change is determined.

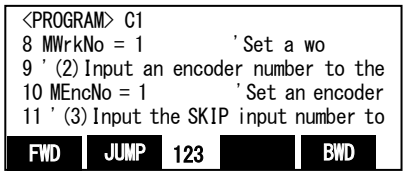


(7) Press the [F1] (FWD) key and execute step feed. "(2) Input an encoder number…"is displayed.



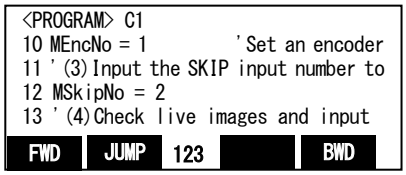
Specify the workpiece number.
If you want to change the workpiece number, please edit the program as similar procedure (6).

(8) Press the [F1] (FWD) key and execute step feed. "(3) Input the SKIP input number…"is displayed.



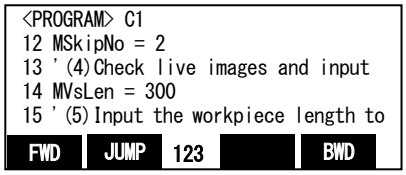
Specify the SKIP input number.
If you want to change the SKIP input number, please edit the program as similar procedure (6).

(9) Press the [F1] (FWD) key and execute step feed. "(4) Check live images and input the…"is displayed.



Specify the length in the movement direction.
Start In-Sight Explorer and make the vision sensor into the off-line. Select the [Live Video] of "Set Up Image" in "Application Steps" Menu and display the picture which the vision sensor picturized on real time. Check the images and set the field of vision in the moving direction of the conveyer (mm) in the variable "MVsLen" in the program, respectively.
If you want to change the length in the movement direction, please edit the program as similar procedure (6).

(10) Press the [F1] (FWD) key and execute step feed. "(5) Input the workpiece length…"is displayed.



Specify the workpiece length.
If you want to change the workpiece length, please edit the program as similar procedure (6).

(11) Press the [F1] (FWD) key and execute step feed. "(6) Input the COM port number to…"is displayed.

```
<PROGRAM> C1
14 MVsLen = 300
15 '(5)Input the workpiece length to
16 MWrkLen = 50
17 '(6)Input the COM port number to
 FWD   JUMP  123        BWD
```

Specify a communication line to be connected with the vision sensor.
If you want to change the communication line, please edit the program as similar procedure (6).

(12) Press the [F1] (FWD) key and execute step feed. "(7) Input the vision program name…"is displayed.

```
<PROGRAM> C1
16 MWrkLen = 50
17 '(6)Input the COM port number to
18 CCOM$="COM2:"    'Set the number
19 '(7)Input the vision program name
 FWD   JUMP  123        BWD
```

Specify a vision program to be started.
If you want to change the vision program, please edit the program as similar procedure (6).

(13) Press the [F1] (FWD) key and execute step feed. "(8) Place workpieces to be tracked…"is displayed.
Place a workpiece to be recognized within the area that the vision sensor can recognize.

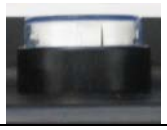
(14) Press the [F1] (FWD) key and execute step feed. "(9) Place the vision sensor in…"is displayed.
Using In-Sight Explorer, place the vision sensor in the online status.

(15) Press the [F1] (FWD) key and execute step feed. "(10) When the program stops…"is displayed.
Using T/B, close the opened "C1" program once and then run the modified "C1" program automatically with the robot controller.

Note) When your controller has no operation panel, use the dedicated external signals corresponding to the following step to operate the robot.
Although the image of the operation panel is the CRnD-700 controller, the operation method is the same in other controllers.

Changing of mode

T/B disabled



Set the T/B [ENABLE] switch to "DISABLE".

Controller enabled

Set the controller [MODE] switch to "AUTOMATIC".

Selection of a program number

Display of a program number

Press the [CHNG DISP] key and display "PROGRAM NO." on the STATUS NUMBER display.

Selection of
a program number

Press the [UP] or the [DOWN] key and display program name"C1"

Start of automatic operation

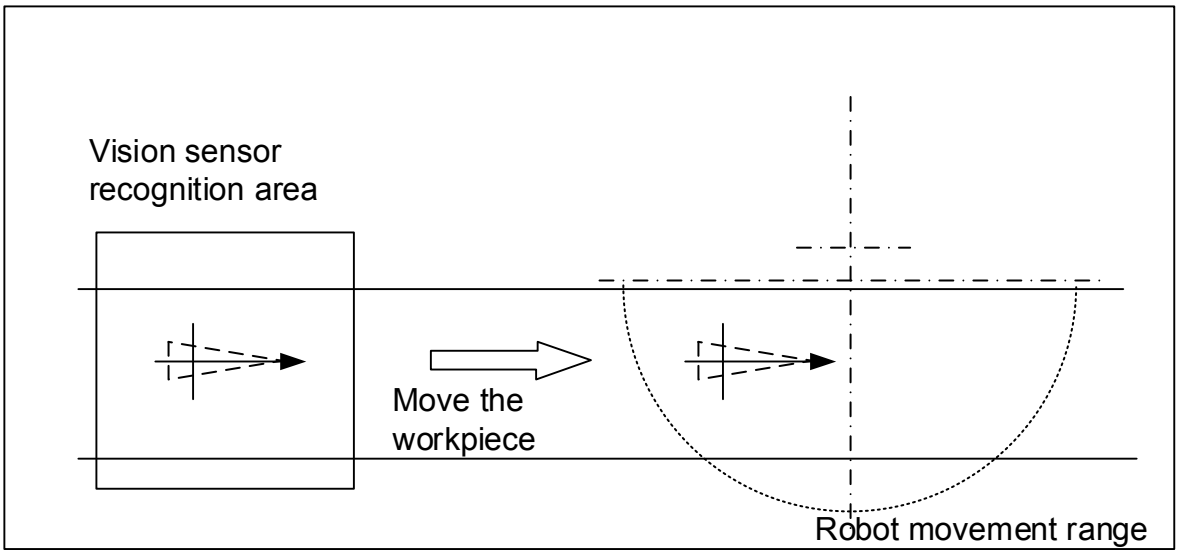
Start

Press the [START] key.

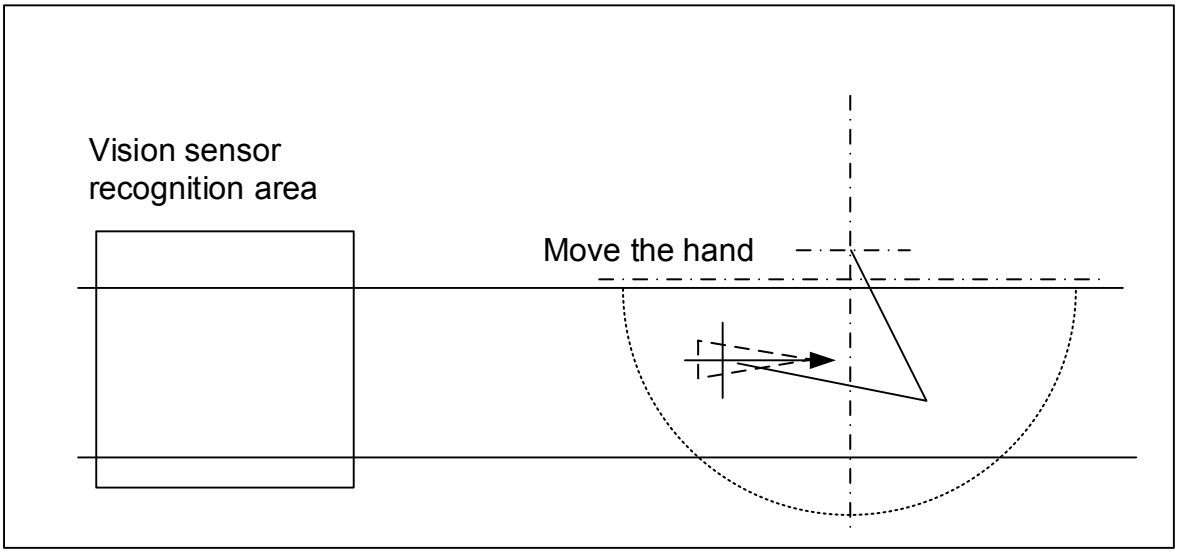
After automatic operation, "C1" program automatically stops and the LED of the [STOP] button is turned on. Open "C1" program again with T/B. Press the [F1] (FWD) key to display the subsequent operation messages.

*** With this operation, encoder data and workpiece position recognized by the vision sensor are acquired.**

(16) Press the [F1] (FWD) key and execute step feed. "(11) Move a workpiece on the…"is displayed.
Rotate the conveyer forward and move a workpiece within the vision sensor recognition area into the robot movement range.



(17) Press the [F1] (FWD) key and execute step feed. "(12) Move the robot to the suction…"is displayed.
Move the robot to the position where it is able to suction the workpiece.
*** With this operation, encoder data and robot position are acquired.**



(18) Press the [F1] (FWD) key and execute step feed. "(13) Perform step operation until END"is displayed.
Perform step operation until "End."
*** With this operation, the robot becomes able to recognize the position of the workpiece recognized by the vision sensor.**

10. 2. 3. Confirmation after operation

Check the values of the following variables using T/B.
Enter the model number for the array number.

- Value of "M_101()": Differences between encoder values when a workpiece is within the vision sensor area and when the workpiece is on the robot side
- Value of "P_100()": Position at which workpieces are suctioned
- Value of "P_101()": Position at which workpieces are recognized by vision sensor
- Value of "P_102()": Data in the variables "MEncNo" and "MSkipNo"(encoder number/SKIP input number)
- Value of "P_103()": Data in the variables "MVsLen" and "MWrkLen" (recognition field of image view/workpiece size)
- Value of "C_100$()": COM number
- Value of "C_101$()": Vision program name

Confirm that each of the above values is entered.

10. 2. 4. When multiple conveyers are used

Carry out the same operations as above when multiple conveyers are used as well, but pay attention to the following points.

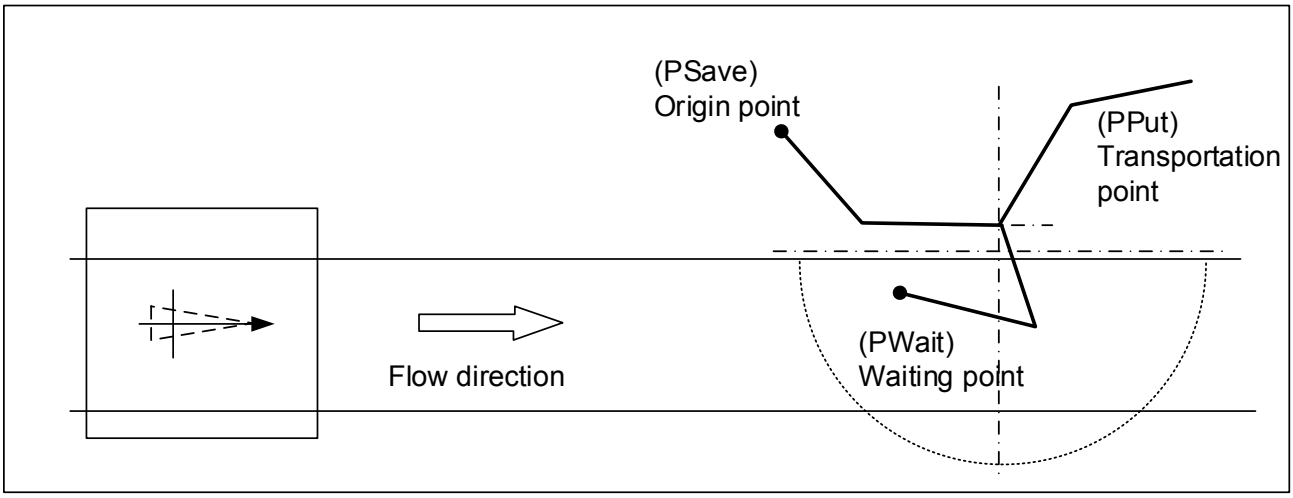
Example) When using conveyer 2 (encoder number "2"), kind number "2",
(a) Copy the "C1" program, please create a "C2" program.
(b) Please change the kind number for variable "MWrkNo" in the "C2" program to "2".
(c) Please change the encoder number for variable "MEncNo" in the "C2" program to "2".

11. Teaching and Setting of Adjustment Variables ("1" program)

This chapter explains operations required to run "1" program.
In addition, this chapter explains a method to check the operation in the condition that it was designated, and to coordinate again.

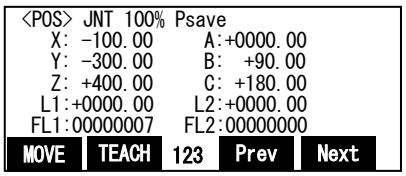
11.1. Teaching

The teaching of "Origin point position (position in which system is started)", "Waiting point position (position in which it is waited that workpiece arrives)" and "Transportation point position (position in which the held workpiece is put)" is executed.



Teach the origin position, waiting position and transportation point. The following explains how to perform these operations.

1) Open "1" program using T/B.
2) Open the [Position data Edit] screen.
3) Display "PSave" in order to set the robot origin position when the system is started.
4) Move the robot to the origin position and teach it the position.
5) Display "PWait" in order to set the waiting position in which it is waited that workpiece arrives.
6) Move the robot to the waiting position and teach it the position.
7) Display "PPut" in order to set the transportation position.
8) Move the robot to the transportation position and teach it the position.
9) Display "PSave" at the starting point position on the [Position data Edit] screen. Turn on the servo by gripping the deadman switch.
10) Push [F1] (MOVE) and move the robot to the position of "PSave".



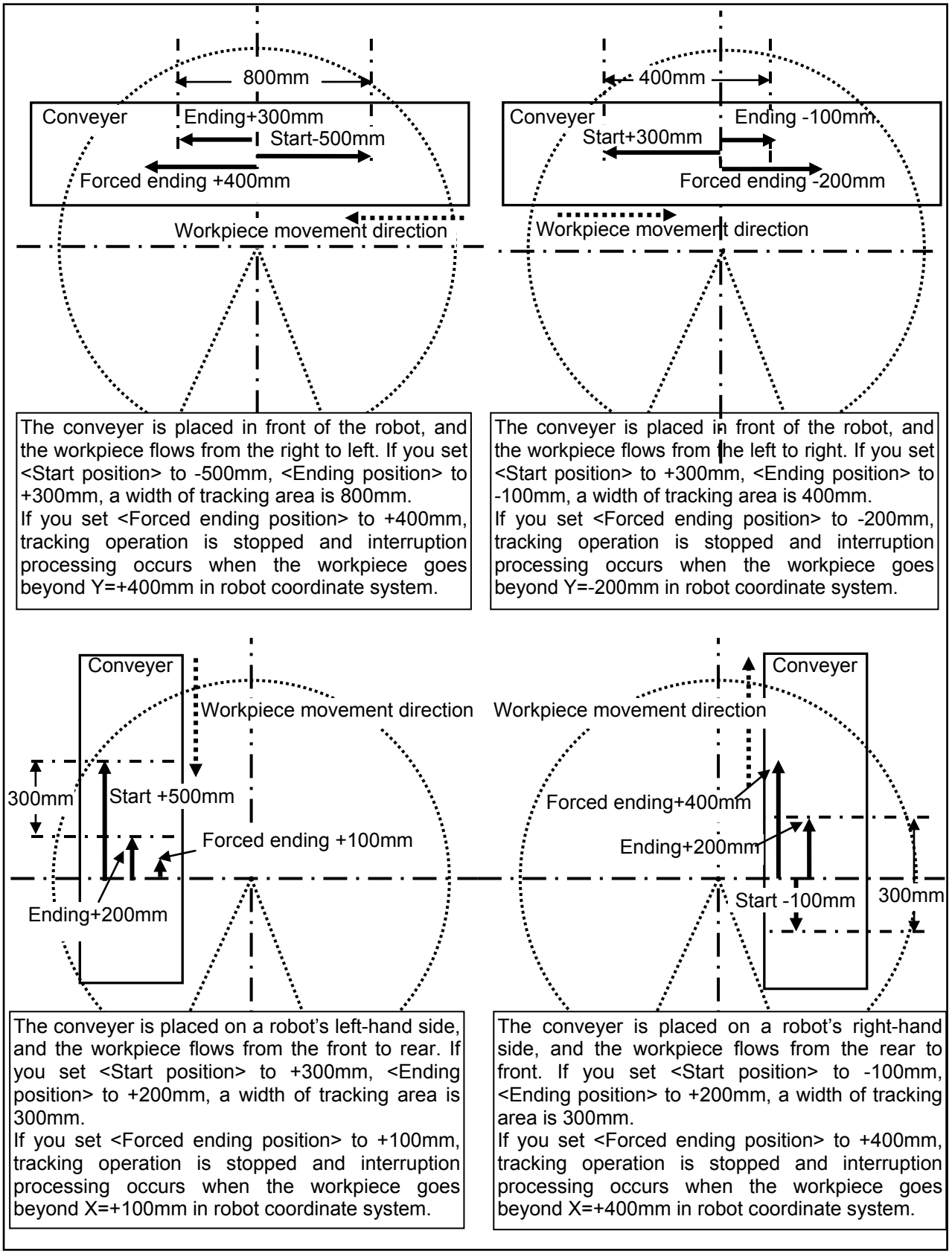
11) Move the robot to the position of "PWait" and "PPut" pushing F1 (MOVE).

11.2. Setting of adjustment variables in the program

The following section explains how to set adjustment variables, which are required at transportation, and details about their setting.
Please refer to separate manual "Detailed Explanations of Functions and Operations" for how to set adjustment variables.

Table **11-1** **List of adjustment variables in the program**

Variable name	Explanation	Setting Example
PUp1	When the adsorption operation of workpiece, set the offset in the z-axis that the robot works. Offset is the amount of elevation (mm) from the position where workpiece is adsorbed. **[*]Since this variable shows the distance in a tool coordinate system, the sign changes depending on a robot model.**	When you raise the workpiece 50mm from the adsorption position: (Example) RV series: (X,Y,Z,A,B,C)=(+0,+0,-50,+0,+0,+0) (Example) Other than RV series: (X,Y,Z,A,B,C)=(+0,+0,+50,+0,+0,+0)
PUp2	When the desorption operation of workpiece, set the offset in the z-axis that the robot works. Offset is the amount of elevation (mm) from the position where workpiece is desorbed. **[*]Since this variable shows the distance in a tool coordinate system, the sign changes depending on a robot model.**	When you raise the workpiece 70mm from the desorption position: (Example) RV series: (X,Y,Z,A,B,C)=(+0, +0,-70,+0,+0,+0) (Example) Other than RV series: (X,Y,Z,A,B,C)=(+0,+0,+70,+0,+0,+0)
PDly1	Set the suction time. X = Suction time (s).	When you set the suction time to 0.5 second: (X,Y,Z,A,B,C)=(+0.5,+0,+0,+0,+0,+0)
PDly2	Set the release time. X = Release time (s).	When you set the release time to 0.3 second: (X,Y,Z,A,B,C)=(+0.3,+0,+0,+0,+0,+0)
PPri	"1" program and "CM1" program are run simultaneously (multitasking). "1" program moves the robot, and "CM1" program observes the sensor. It is possible to specify which program is processed with a higher priority, rather than performing the same amount of processing at the same time. X = Set the line numbers of "1" program to be performed (1 to 31). Y = Set the line numbers of "CM1" program to be performed (1 to 31).	When you set to run "1" program by one line and run "CM1" program by 10 lines: (X,Y,Z,A,B,C)=(+1,+10,+0,+0,+0,+0)
POffset	When the adsorption position shifts, the gap can be corrected. Set the correction value. **[*]The direction of the correction is a direction of the hand coordinate system. Please decide the correction value after changing the job mode to "Tool", pushing the [+X] key and the [+Y] key, and confirming the operation of the robot.**	When the deviation to +X direction in hand-coordinate system is 2mm, and deviation to -Y direction in hand-coordinate system is 1mm: (X,Y,Z,A,B,C)=(+2,-1,+0,+0,+0,+0)
PRng	Set the range of motion where the robot judges workpiece to be able to follow, and the forced ending distance. (When the workpiece is in the tracking possible area, the tracking is started. But if the robot speed is low, and the conveyer speed is high, the robot follows the workpiece to out of the robot operation area.) X = The start distance of the range in which the robot can follow a workpiece :(mm) Y = The end distance of the range in which the robot can follow a workpiece :(mm) Z = The distance in which follow is canceled.	**Refer to「Figure 11-1 Diagram of the adjustment variables "PRNG" in the Program」**



The conveyer is placed in front of the robot, and the workpiece flows from the right to left. If you set <Start position> to -500mm, <Ending position> to +300mm, a width of tracking area is 800mm.
If you set <Forced ending position> to +400mm, tracking operation is stopped and interruption processing occurs when the workpiece goes beyond Y=+400mm in robot coordinate system.

The conveyer is placed in front of the robot, and the workpiece flows from the left to right. If you set <Start position> to +300mm, <Ending position> to -100mm, a width of tracking area is 400mm.
If you set <Forced ending position> to -200mm, tracking operation is stopped and interruption processing occurs when the workpiece goes beyond Y=-200mm in robot coordinate system.

The conveyer is placed on a robot's left-hand side, and the workpiece flows from the front to rear. If you set <Start position> to +300mm, <Ending position> to +200mm, a width of tracking area is 300mm.
If you set <Forced ending position> to +100mm, tracking operation is stopped and interruption processing occurs when the workpiece goes beyond X=+100mm in robot coordinate system.

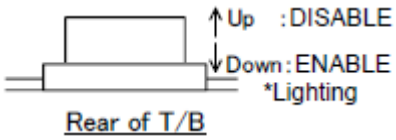
The conveyer is placed on a robot's right-hand side, and the workpiece flows from the rear to front. If you set <Start position> to -100mm, <Ending position> to +200mm, a width of tracking area is 300mm.
If you set <Forced ending position> to +400mm, tracking operation is stopped and interruption processing occurs when the workpiece goes beyond X=+400mm in robot coordinate system.

Figure 11-1 Diagram of the adjustment variables "PRNG" in the Program

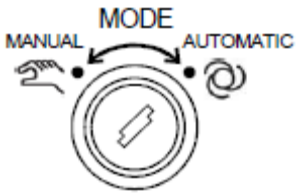
11. 3. Automatic Operation

This chapter explains how to prepare the robot before starting the system.

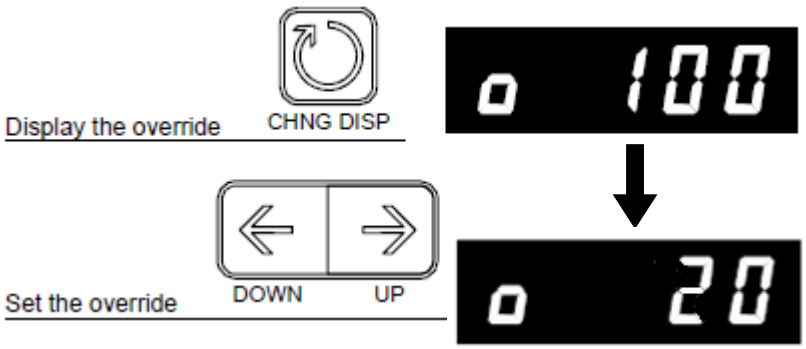
(1) Confirm that there isn't an intervention thing in the robot movement area.
(2) Set the T/B [ENABLE] switch to "DISABLE"



(3) Set the controller mode to "AUTOMATIC".

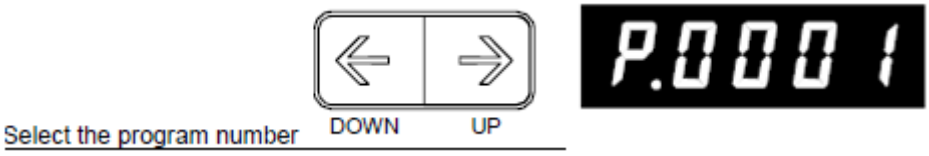


(4) Press the controller [CHNG DISP] button twice, and display the "OVERRIDE" on the STATUS NUMBER display panel, and specify the override to 20% - 30%.



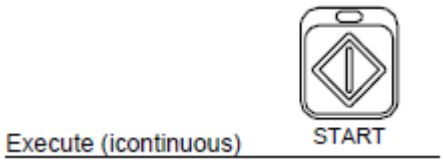
(5) Press the [CHNG DISP] button and display "PROGRAM NO." on the STATUS NUMBER display. Then press the [RESET] button to reset program.

(6) Press the [UP] key or the [DOWN] button and display "program 1" to the STATUS NUMBER display.

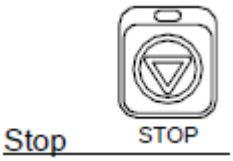


(7) Automatic operation will start when the controller [START] button is pressed.

***Prepare for the unexpected operation of the robot,please can press anytime emergency stop switch of T/B.**



(8) When the robot moves to the specified retracted position, to drive the turntable and place the workpiece.
(9) Confirm to be a work that is unloaded to the transport destination after following the workpiece.
(10)If you check the operation, press the [STOP] button and stop the robot.



⚠ POINT

T / B software in a specific version or later, you can be the automatic operation from T / B.

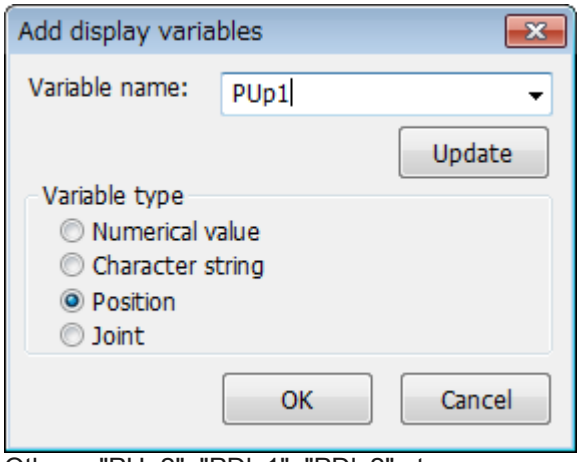
With R32/33T/B software version 1.7 or later, the program's automatic operation can be started from the T/B (With R56/57TB, version 3.0 or later). Please refer to "Detailed Explanations of Functions and Operations" for operation procedures and details.

11. 4. Adjustment of operating conditions

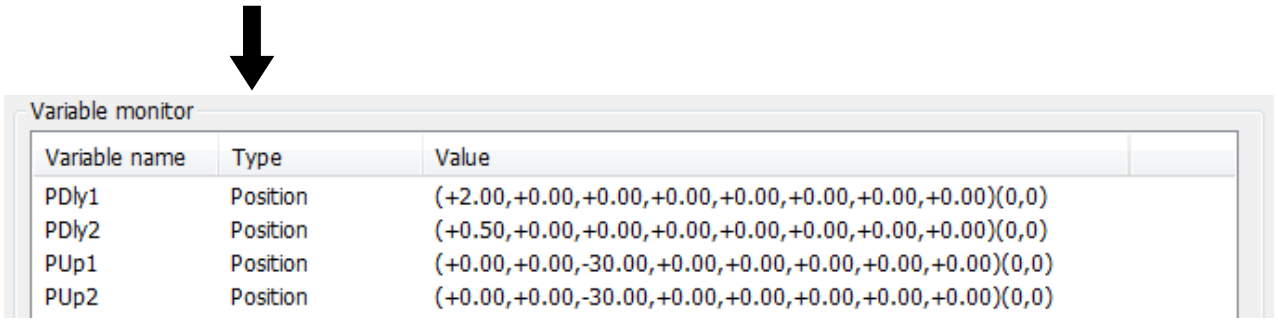
In automatic operation, if you want to adjust the vertical movement and adsorption time of the robot arm that was described in "11.2 Setting of adjustment variables in the program" should be changed in the following procedure.

(1) Start the "Program monitor" of RT ToolBox2.

(2) Click the [Add] button and open the "Add display variables" screen. Enter the variables listed in the "**Table 11-1 List of adjustment variables in the program**", and then click the [OK] button.

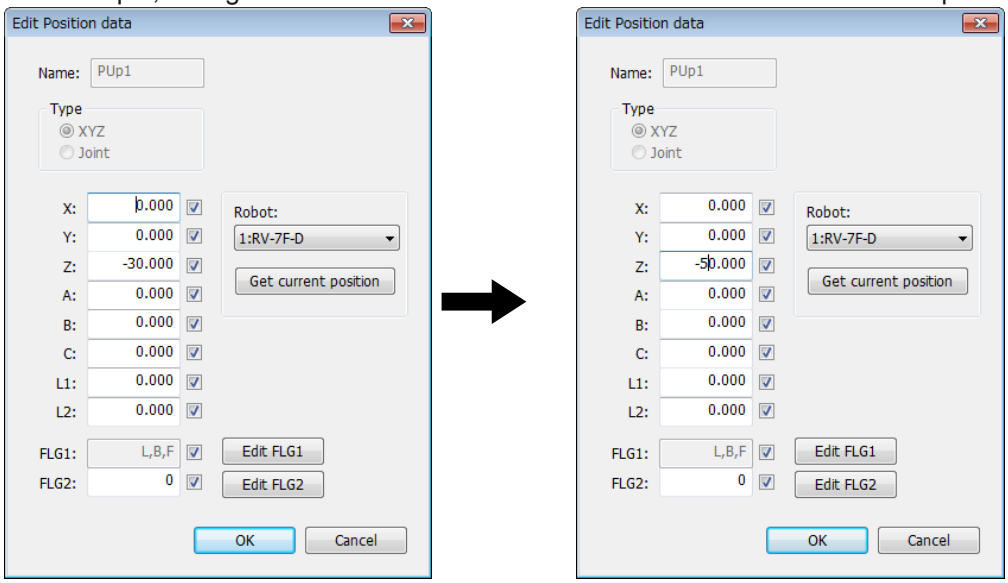


Others, "PUp2", "PDly1", "PDly2" etc.

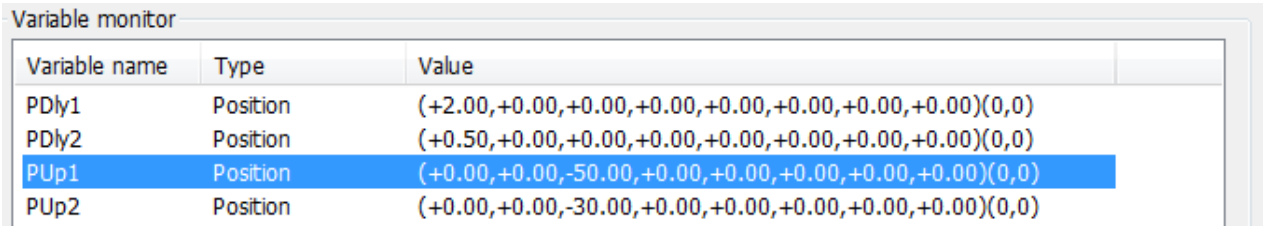


(3) Double-click the variable you want to change, and change the appropriate value for displayed in the "Edit Position data".

For example, change to "-50" from "-30" the value of the Z-coordinate of the PUp1 :



(4) Click [OK] button, and confirm that was able to change the value of the variable that is specified in the "Variable Monitor".


Variable monitor

Variable name	Type	Value
PDly1	Position	(+2.00,+0.00,+0.00,+0.00,+0.00,+0.00,+0.00,+0.00)(0,0)
PDly2	Position	(+0.50,+0.00,+0.00,+0.00,+0.00,+0.00,+0.00,+0.00)(0,0)
PUp1	Position	(+0.00,+0.00,-50.00,+0.00,+0.00,+0.00,+0.00,+0.00)(0,0)
PUp2	Position	(+0.00,+0.00,-30.00,+0.00,+0.00,+0.00,+0.00,+0.00)(0,0)

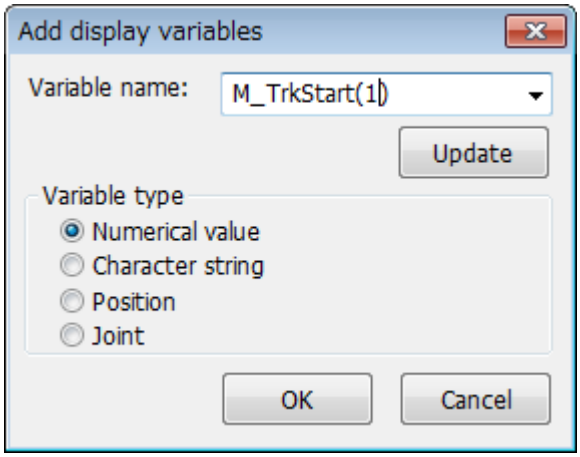
(5) Return to the "11.3 Automatic Operation", and then check to see whether the can be corrected by implementing the automatic operation.

11.5. Adjustment of Tracking starting possible area

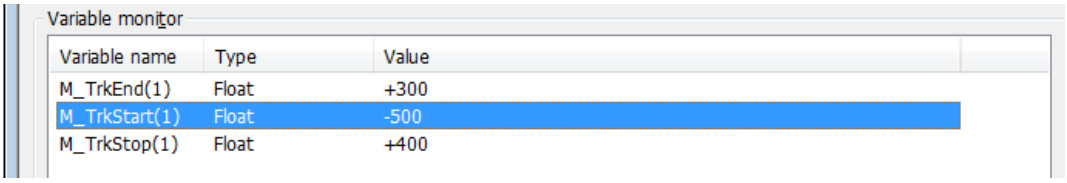
In automatic operation, if you want to adjust the Tracking starting possible area that was taught in the "8 Calibration of Conveyer and Robot Coordinate Systems ("A1" program)", change the following procedure.

(1) Start the "Program monitor" of RT ToolBox2.

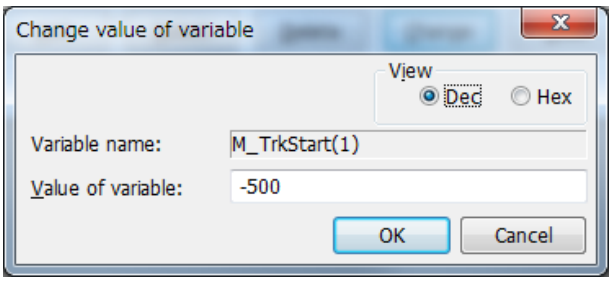
(2) Click the [Add] button and open the "Add display variables" screen. Enter the following three state variables, and then click the [OK] button.



Others, "M_TrkEnd(1)", "M_TrkStop(2)" *In (), specify the [condition number].

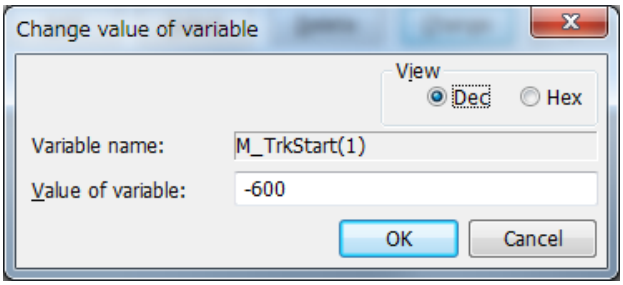


(3) Double-click the variable you want to change, and change the value in the displayed "Changing Values" screen.



Assume that the movement direction of the conveyer "plus", input the value to which the offset was added, and then click [OK].

For example, if you want the tracking started early 100mm:



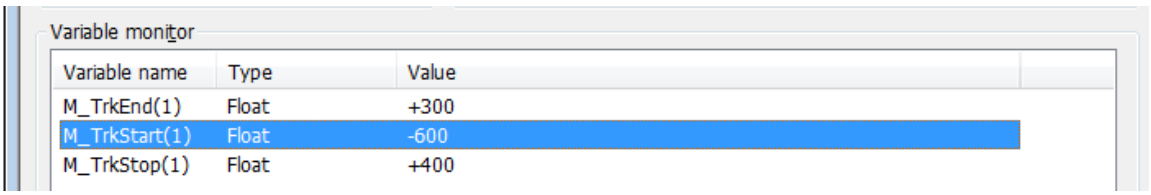
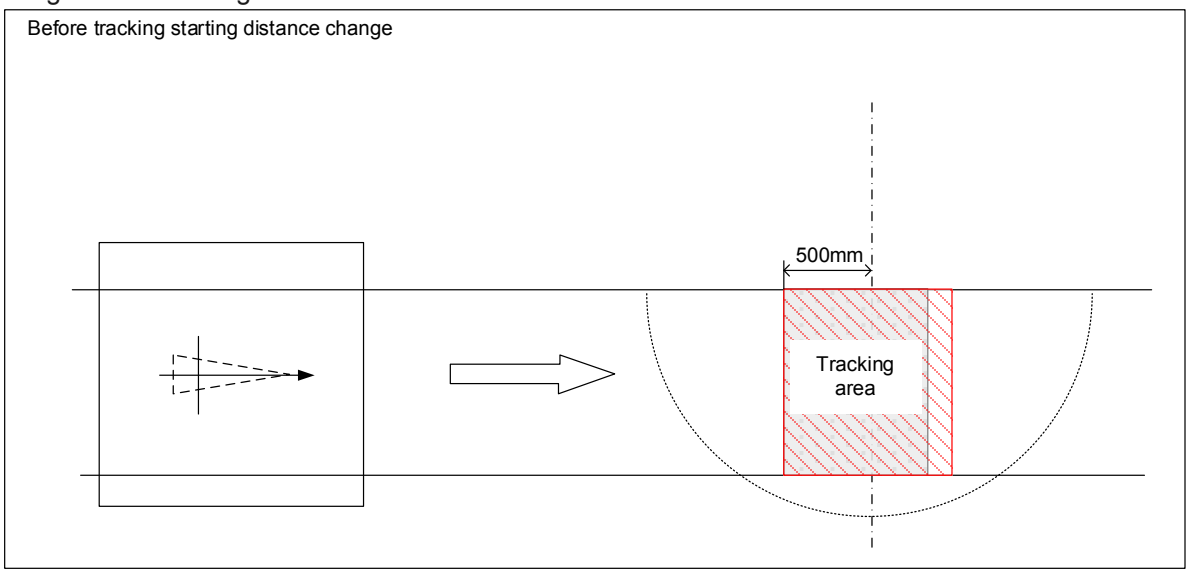
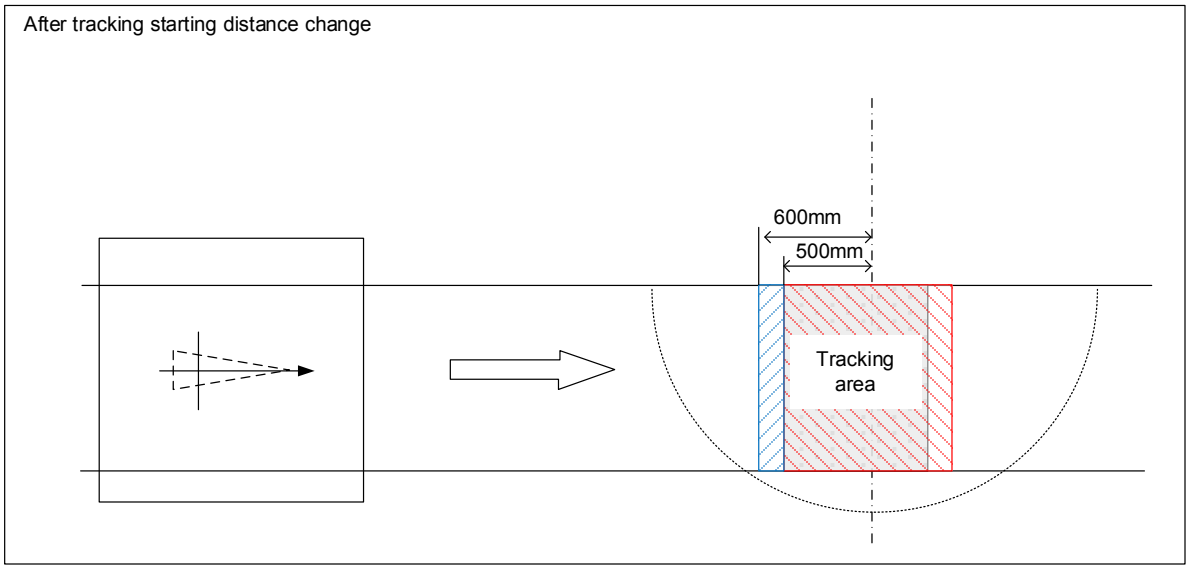


Image of the tracking area is as follows.





(4) Similarly, please adjust using the "M_TrkEnd" for the end position of the tracking starting possible area.

Also, please adjust using the "M_TrkStop" for the position to be forcibly terminated.

11.6. Occurrence of error

When an error occurred, please confirm the "14 Troubleshooting".

12. Sensor Monitoring Program ("CM1" program)

This chapter provides an overview of "CM1" program, which is run in parallel, when "1" program is run.
Different types of "CM1" programs are used for conveyer tracking and vision tracking, and different processing is performed for them. These programs are explained in the following.

12.1. Conveyer Tracking

"CM1" program calculates the workpiece coordinates in the robot coordinate system at the moment where a photoelectronic sensor is activated based on the following data acquired with "A1" program and "C1" program, and then stores the coordinates in the tracking buffer(Storage area to preserve data temporarily).
<Acquired data>

- Amount of robot movement per encoder pulse (P_EncDlt)
- Difference between the encoder value when a photoelectronic sensor is activated and the encoder value when teaching is performed on a robot
- Position at which the robot is taught to grab a workpiece

12.2. Vision Tracking

"CM1" program converts the workpiece position recognized by the vision sensor to the corresponding coordinates in the robot coordinate system based on the following data acquired with "A1" program, "B1" program and "C1" program, and then stores the coordinates in the tracking buffer.
<Acquired data>

- Amount of robot movement per encoder pulse(P_EncDlt)
- Difference between the encoder value when a marking sticker is on the vision sensor side and the encoder value when the marking sticker is on the robot side
- Workpiece position recognized by the vision sensor
- Difference between the encoder value when the vision sensor recognizes a workpiece and the encoder value when teaching on the workpiece position was performed on the robot
- Position at which the robot is taught to grab a workpiece

The timing at which the vision sensor acquires images is calculated such that images of the same workpiece are taken at least once or up to twice by the following data specified in "C1" program.
<Data specified in "C1" program>

- Field of view in the conveyer movement direction
- Length of workpieces detected by a vision sensor (length in the conveyer movement direction)

⚠ POINT

"1" program follows workpieces on a conveyer based on the workpiece information stored in the tracking buffer in "C" program.

"C" program performs processing until the recognized workpiece position is stored in the tracking buffer. The workpiece information stored in the tracking buffer is read by "1" program and the robot follows workpieces on the conveyer based on the information.

13. Maintenance of robot program

This chapter explains information required when maintaining the sample programs (robot program language MELFA-BASIC V and dedicated input/output signals).

13.1. MELFA-BASIC V Instructions

The lists of instructions, status variables and functions related to tracking operation are shown below.
Please refer to the separate manual “Detailed Explanations of Functions and Operations” for further information about MELFA-BASIC V.

13.1.1. List of Instructions

Table **13-1** **List of Instructions**

Instruction name	Function
TrClr	Clear the tracking data buffer.
TrWrt	Write workpiece data in the tracking data buffer.
TrRd	Read workpiece data from the tracking data buffer.
TrkChk	Execute the processing depending on the state of workpiece corresponding to <Condition number> specified.
TrkWait	Wait until workpiece corresponding to <Condition number> specified enters to the tracking area.
TrkMv	Execute the next processing. Validate specified interruption, Start tracking, Move to the tracking upper position by Joint interpolation movement.
TrkFine	The accuracy at the tracking is improved until “TrkFine Off” is executed.
TrkTrg	Request the specified vision sensor to capture an image, and acquires encoder value after the SKIP input receives the signal from the vision sensor.
NVOpen	Connects with the vision sensor and logs on to the vision sensor.
NVClose	Cuts off the connection with vision sensor.
NVLoad	Puts the specified vision program into the state in which it can be started.
EBRead	Reads the data for which the tag name of the vision sensor is specified.

13.1.2. List of Robot Status Variables

Table **13-2** **List of Robot Status Variables**

Variable name	Number of arrays	Function	Attribute (*1)	Data type
M_Enc	number of encoders 1 to 8	External encoder data External encoder data can be rewritten. **If this state variable does not set parameter “TRMODE" to “1", the value becomes “0".**	R/W	Double-precision real number
M_EncL	Number of encoder 1 to 8	The stored encoder data ※ **Possible to use from R1 and S1** ※ **0 always returns in S1.**	R/W	Double-precision real number
P_EncDlt	number of encoders 1 to 8	Amount of robot movement per encoder pulse ***This state variable is made by sample “A1” program.**	R/W	Position
P_TrkPAcl	Condition Number 1 to 8.	Parameter [TRPACL] value	R/W	Position
P_TrkPDcl	Condition Number 1 to 8.	Parameter [TRPDCL] value	R/W	Position
M_TrkBuf	Condition Number 1 to 8.	Buffer Number	R/W	Integer

Variable name	Number of arrays	Function	Attribute (*1)	Data type
M_TrkStart	Condition Number 1 to 8.	Tracking Starting Distance	R/W	Single-precision real number
M_TrkEnd	Condition Number 1 to 8.	Tracking Ending Distance	R/W	Single-precision real number
M_TrkStop	Condition Number 1 to 8.	Tracking Forced Ending Distance	R/W	Single-precision real number
M_TrkTime	Condition Number 1 to 8.	Timeout period of TrkWait command	R/W	Single-precision real number
P_TrkBase	Condition Number 1 to 8.	Tracking Base coordinates	R/W	Position
M_TrkChk	Condition Number 1 to 8.	TrkChk result	R	Integer
P_TrkWork	Condition Number 1 to 8.	Workpiece position when the sensor taken out from the tracking buffer reacts.	R	Position
M_TrkEnc	Condition Number 1 to 8.	Workpiece Encoder when the sensor taken out from the tracking buffer reacts.	R	Long-precision real number
M_TrkKind	Condition Number 1 to 8.	Model number of the workpiece taken out from the tracking buffer.	R	Integer
M_TrkEncNo	Condition Number 1 to 8.	Encoder number taken out from the tracking buffer.	R	Integer
P_TrkPixel	Condition Number 1 to 8.	Workpiece pixel position when the sensor taken out from the tracking buffer reacts.	R	Position
P_TrkTarget	-	The workpiece coordinate where the robot is following	R	Position
M_Trbfct	buffer No. 1 to The first argument of parameter [TRBUF]	Number of data items stored in the tracking buffer	R	Integer
P_CvSpd	number of encoders 1 to 8	Conveyer speed (mm, rad/sec)	R	Position
M_Hnd	Hand Number 1 to 8	Hand open/close instruction and Hand open/close states. ※Used when you open or close the hand during “WthIf”.	R/W	Integer
M_NvOpen	Vision Sensor Number 1 to 8	Indicates the vision sensor line connection status.	R	Integer

13. 1. 3. Explanation of Tracking Operation instructions

The instructions related to tracking operations are explained in details below.

The explanations of instructions are given using the following format.

[Function] : Describes the function of an instruction.
[Format] : Describes the entry method of arguments of an instruction.
< > indicate an argument.
[ ] indicates that entry can be omitted.
□ indicate that space is required.
[Term] : Describes meaning, range and so on of an argument.
[Example] : Presents statement examples.
[Explanation] : Provides detailed function descriptions and precautions.

TrClr (Tracking data clear)

[Function]

Clear the tracking data buffer.

[Format]

TrClr □ [<Buffer number>]

[Terminology]

<Buffer number [integer]**>** (can be omitted):

Specify the number of a general-purpose output to be output.
Setting range:1 to The first argument of parameter "TRBUF"

[Reference program]

```
1 TrClr 1                          ' Clear the tracking data buffer No. 1.
2 *LOOP
3 If M_In(8)=0 Then GoTo *LOOP     ' Jump to *LOOP if input signal No. 8, to which a photoelectronic
                                     sensor is connected, is OFF.
4 M1#=M_Enc(1)                     ' Acquire the data of encoder number 1 at the time when input signal
                                     No. 8 is turned on and store it in M1#.
5 TrWrt P1, M1#,MK                 ' Write workpiece position data P1, encoder value M1# at the time  an
                                     image is acquired and model number MK into the buffer.
```

[Explanation]

(1) Clear information stored in specified tracking buffer.
(2) Execute this instruction when initializing a tracking program.

TrWrt (Writing tracking data)

[Function]

Write position data for tracking operation, encoder data and so on in the data buffer.

[Format]

TrWrt □ <Position data> [, [<Encoder data>] [, [<Model number>] [, [<Buffer number>] [, [<Encoder number>] [,[<Pixel data>]]]]]]

[Terminology]

<Position data [Position]**>** (cannot be omitted):
Specify the workpiece position measured by a sensor.

<Encoder data [double-precision real number]**>** (can be omitted):
Specify the value of an encoder mounted on a conveyer at the time a workpiece is measured.
The encoder value acquired in the M_Enc() state variable and the TrOut instruction is specified usually.

<Model number [integer]**>** (can be omitted):
Specify the model number of workpieces.
Setting range: 1 to 65535

<Buffer number [integer]**>** (can be omitted):
Specify a data buffer number.
1 is set if the argument is omitted.
Setting range: 1 to 4(The first argument of parameter [TRBUF])

<Encoder number [integer]**>** (can be omitted):
Specify an external encoder number.
The same number as the buffer number is set if the argument is omitted.
Setting range: 1 to 8

<Pixel data [position]**>** (can be omitted):
Specify the workpiece pixel position measured by a sensor.

[Reference program]

(1) Tracking operation program

```
1 TrBase P0                  ' Specify the workpiece coordinate origin at the teaching position.
2 TrRd P1, M1, MK, 1, ME, P3  ' Read the workpiece position data from the data buffer.
3 Trk On,P1,M1               ' Start tracking of a workpiece whose measured position is P1 and encoder
                               value at the time of measurement is M1.
4 Mvs P2                     ' Setting the current position of P1 as P1c, make the robot operate while
                               following workpieces with the target position of Inv(P0) * P2.Add that to the
                               target location.And tracking.
5 HClose 1                   ' Close hand 1.
6 Trk Off                    ' End the tracking operation.
```

(2) Sensor data reception program

```
1 *LOOP
2 If M_In(8)=0 Then GoTo *LOOP   ' Jump to +LOOP if input signal No. 8, to which a photoelectronic
                                   sensor is connected, is OFF.
3 M1#=M_Enc(1)                   ' Acquire data of encoder number 1 at the time when input signal
                                   No. 8 is turned on and store it in M1#.
4 TrWrt P1, M1#,MK               ' Write workpiece position data P1, encoder value M1# at the time
                                   an image is acquired and model number MK in the buffer.
```

[Explanation]

(1) This function stores the workpiece position (robot coordinates) at the time when a sensor recognizes a workpiece, encoder value, model number, encoder number and workpiece position (pixel coordinates) in the specified buffer.

(2) Arguments other than the workpiece position (robot coordinates) can be omitted. If any of the arguments are omitted, the robot operates while following changes of position data.

(3) Workpieces within the same workpiece judgment distance set in the “TRCWDST” parameter are regarded as the same workpiece. Even if the data is written twice in the buffer with the TrWrt instruction, only one data set is stored in the buffer. For this reason, data for one workpiece only is read with the TrRd instruction even if images of the same workpiece are acquired twice with a vision sensor.

TrRd (reading tracking data)

[Function]

Read position data for tracking operation, encoder data and so on from the data buffer.

[Format]

TrRd □ <Position data> [, [<Encoder data>] [, [<Model number>] [, [<Buffer number>] [, [<Encoder number>] [, [Pixel data]]]]]]

[Terminology]

<Position data [Position]**>** (cannot be omitted):
Specify a variable that contains workpiece positions read from the buffer.

<Encoder data [double-precision real number]**>** (can be omitted):
Specify a variable that contains encoder values read from the buffer.

<Model number [integer]**>** (can be omitted):
Specify a variable that contains model numbers read from the buffer.

<Buffer number [integer]**>** (can be omitted):
Specify a number of a buffer from which data is read.
1 is set if the argument is omitted.
Setting range: 1 to 4(The first argument of parameter [TRBUF])

<Encoder number [integer]**>** (can be omitted):
Specify a variable that contains values of external encoder numbers read from the buffer.

<Pixel data [position]**>** (can be omitted):
Specify a variable that contains workpiece pixel positions read from the buffer.

[Reference program]

(1) Tracking operation program

```
1 TrBase P0                  ' Specify the workpiece coordinate origin at the teaching position.
2 TrRd P1, M1, MK, 1, ME, P3 ' Read the workpiece position data from the data buffer.
3 Trk On,P1,M1               ' Start tracking of a workpiece whose measured position is P1 and encoder
                               value at the time of measurement is M1.
4 Mvs P2                     ' Setting the current position of P1 as P1c, make the robot operate while
                               following workpieces with the target position of Inv(P0) * P2.Add that to the
                               target location.And tracking.
5 HClose 1                   ' Close hand 1.
6 Trk Off                    ' End the tracking operation.
```

(2) Sensor data reception program

```
1 *LOOP
2 If M_In(8)=0 Then GoTo *LOOP    ' Jump to *LOOP if input signal No. 8, to which a photoelectronic
                                    sensor is connected, is OFF.
3 M1#=M_Enc(1)                    ' Acquire data of encoder number 1 at the time when input signal
                                    No. 8 is turned on and store it in M1#.
4 TrWrt P1, M1#,MK                ' Write workpiece position data P1, encoder value M1# at the time
                                    an image is acquired and model number MK in the buffer.
```

(3) Vision data reception program

```
1 NVClose                                    ' Close communication line
2 NVOpen “COM2:” As #1                       ' Open communication line and log on
3 Wait M_NvOpen(1) = 1                       ' Wait to log on to the vision sensor
4 NVLoad #1, “test”                          ' Load the vision program
5 NVTrg #1, 5, MTR1”                         ' Imaging request + encoder value acquisition
6 EBRead #1,””,MNUM,PVS1,PVS2,PVS3,PVS4      ' Acquire data of one recognized workpiece
7 MVsX = PV1.X                               ' Acquire X data
8 MVsY = PVS1.Y                              ' Acquire Y data
9 MVsC = Deg(PVS1.C)                         ' Acquire the C data converted to the degree unit
10 PosVS = PVSCal(1, MVsX, MVsY, MVsC)       ' Acquire the position data changed from a pixel to a
                                               robot coordinate
11 TrWrt PosVS, MTR1#, 1, 1, 1, PVS1         ' Write data in the buffer
```

[Explanation]

(1) Read the workpiece position (robot coordinates), encoder value, model number, encoder number and workpiece position (pixel coordinates) stored by the TrWrt instruction from the specified buffer.

(2) If the TrRd instruction is executed when no data is stored in the specified buffer, Error 2540(There is no read data) occurs.

TrkChk (Tracking check function)

[Function]

Execute the processing depending on the state of workpiece corresponding to <Condition number> specified.

[Format]

TrkChk □ <Condition number> , <Starting position> , [<Waiting position>] , <Branch destination>

[Terminology]

<Condition number [Integer]**>**

Specify the condition number correspond to tracking.
Setting range: 1 to 8

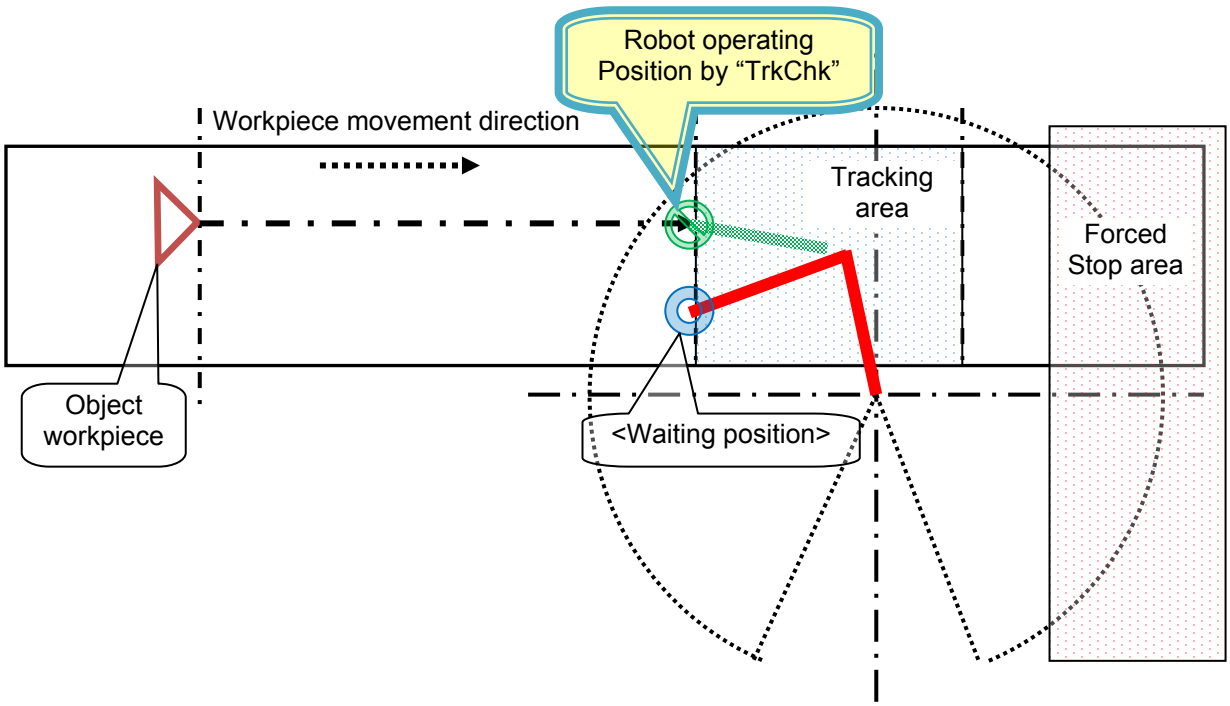
<Starting position [Position]**>**

When there is no workpiece in tracking buffer(no workpiece on the conveyor), specify the starting position to which robot moves at the beginning of the system. Mainly, specify the starting position as the system to which robot moves at the beginning of the system.

<Waiting position [Position]**> :** (can be omitted.)

Specify the waiting position until workpiece enters a tracking possible area.
In the case of vision tracking, a robot moves to the position which has grasped the position through which workpiece flows and changed the value of X and C coordinates, or Y and C coordinates from the value of X and Y of a state variable "P_EncDlt" to the specified <Waiting position>.
(*)It is effective for X or Y coordinates in “P_EncDlt”, it does not support Z-coordinates.

If you omit <Waiting position>, even if workpiece flows, the robot does not move.
By omitting <Waiting position>, you can move to the fixed position. And you can move to the arranged position by using state variable “P_TrkTarget”.



<Branch destination [label]**>**

Specify the label name that jumps when specified workpiece can be followed.

[Reference program]

```
*LBFCHK
 .....
 TrkChk 1, P1, PWAIT, *LTRST              'No workpiece->P1/ Wait for the workpiece->PWAIT/
                                                          Tracking possible->Jump to “LTRST”.
 If M_TrkChk(1) <= 1 Then GoTo *LBFCHK    '0:No workpiece   /   1: Workpiece passed over ->“LBFCHK”.
 TrkWait *LBFCHK                          'Wait for the workpiece / Jump to “LBFCHK” at the timeout.
```

[Explanation]

(1) Workpiece information is taken out of the tracking buffer of state variable "M_TrkBuf" corresponding to <condition number >.The position of the workpiece is checked by using the range specified for robot state variable “M_EncSensor”,”M_EncStart”,“M_EncEnd”,“M_EncStop”,“M_TrkStart”,“M_TrkEnd”,“M_TrkStop” The checked result is stored in robot state variable “M_TrkChk”.

(2) Workpiece information which is taken out of the specified tracking buffer is in state variable “P_TrkWork”, “M_TrkEnc”, “M_TrkKind” , “M_TrkEncNo” and “P_TrkPixel” when “TrkChk” is executed.

(3) If state variable “M_TrkBuf” is not specified when “TrkChk” is executed, buffer number is assumed to be “1”.

(4) Execute the following processings according to the execution result of this command.

M_TrkChk value	Execution result	Processing	Robot operation
0	No workpiece in the tracking buffer.	Execute the process that move to specified <Starting position>.	Robot move from current position to <Starting position>.
1	There is workpiece information in the tracking buffer. And the workpiece has passed the tracking starting possible area.	No processing.	Robot does not move.
2	There is workpiece information in the tracking buffer. And the workpiece exists in front of the tracking starting possible area.	Confirm the workpiece position. Change the position data of specified <Waiting position>. Move to the position.	Robot moves from the current position to the position to which the workpiece flows.
3	There is workpiece information in the tracking buffer. And the workpiece exists in the tracking starting possible area.	Jump to the specified <Branch destination>.	Robot does not move.

(5) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110_99000 (Argument value range over) error to occur.

(6) If you appoint the label which does not exist as “Branch destination”, error L3600_00000 (Jump destination does not exist) occurs.

TrkWait (Tracking wait function)

[Function]

Wait until workpiece correspond to appointed <Condition number> enters to the tracking area.

[Format]

TrkWait □ < Branch destination >

[Terminology]

<Branch destination [label]**>** :(can be omitted.)

Even if the time specified as the state variable "M_TrkTime" passes, when the specified work piece does not go into tracking area, specify the label name to jump.

If < Branch destination > is omitted, the timeout does not occur, and workpiece information is written into the tracking buffer by “TrWrt”, waits until the workpiece enters to the tracking possible area.

[Reference program]

```
M_TrkTime(1) = 60                       ' The timeout period is 60 seconds.
.....
'/// Tracking buffer check ///
*LBFCHK
TrkChk 1, PSave, PWait, *LTRST          'No workpiece->PSave/ Wait for the workpiece->PWait/
                                                Tracking possible->Jump to “LTRST”.

If M_TrkChk(1) <= 1 Then GoTo *LBFCHK   ' 0:No workpiece  /  1: Workpiece passed over->“LBFCHK”.
TrkWait *LBFCHK                         ' Wait for the workpiece / Jump to “LBFCHK” at the timeout.
```

[Explanation]

(1) Take workpiece information out of “TrkChk”, wait until the workpiece enters to the range specified for state variable “M_TrkStart” and “M_TrkEnd”.

(2) When work piece passes away by discontinuation etc., the following work piece information is taken out from a tracking buffer, and it waits until the work piece goes into the range specified as a state variable "M_TrkStart" and "M_TrkEnd."

(3) If specified workpiece does not enter to the tracking area when the time specified for state variable "M_TrkTime" is exceeded at waiting time, jump to <Branch destination>.

(4) When robot state variable "M_TrkBuf" is not executed, the buffer number is assumed to be “1".

(5) If <Branch destination> is omitted or state “M_TrkTime” is “0.00”, the timeout does not occur, and workpiece information is written in into the tracking buffer by “TrWrt”, waits until the workpiece enters to the tracking possible area.

(6) If you appoint the label which does not exist as <Branch destination>, error 3600_00000 (Jump destination does not exist) occurs.

TrkMv (Tracking movement function)

[Function]

Execute the next processing. Validate specified interruption, Start tracking, Move to the tracking upper position by Joint interpolation movement.

[Format]

```
TrkMv □ On , <Tracking upper position> [, <Interrupt number> , <Branch destination>]
TrkMv □ Off
```

[Terminology]

<Tracking upper position [position]**>**

Specify the tracking upper position to follow. (Example : PGT * PGUP1)

<Interrupt number [Integer]**>** : (can be omitted.)

Specify the interrupt number checks the following.

・When tracking, does the workpiece reach <Forced Ending Distance > specified for robot state variable "M_TrkStop()"?

Setting range: 1 to 8

<Branch Destination [Label]**>** :(can be omitted.)

Specify the jumping label name when specified workpiece reach <Forced Ending Distance >.

[Reference program]

```
M_TrkBuf(1) = 1                          ' <Buffer number> is "1".
P_TrkBase(1) = PTBASE                    ' P_TrkBase(1) variable is PTBASE variable.
・・・・・
'/// Tracking buffer check ///
*LBFCHK
TrkChk 1, PSave, PWait, *LTRST           'No workpiece->PSave/ Wait for the workpiece->PWait/
                                         Tracking possible->Jump to "LTRST".
If M_TrkChk(1) <= 1 Then GoTo *LBFCHK    ' 0:No workpiece  /  1: Workpiece passed over->"LBFCHK".
TrkWait *LBFCHK                          ' Wait for the workpiece  / Jump to "LBFCHK" at the timeout.
・・・・・
'/// Start tracking operation ///
*LTRST
TrkMv On, PGTUP, 1, *S91STOP             'Start the interrupt check->Trk On->Move to the tracking upper
                                         position / In the case of exceeding the distance specified by
                                         "M_TrkStop"-Trk Off→Jump to "S91STOP"
・・・・・ adsorption / Release / assembly etc. ・・・・・
TrkMv Off                                'Stop the interrupt check -> Trk Off
```

[Explanation]

(1) In the case of "TrkMv On", if the workpiece position exceed the distance specified by "M_TrkStop", execute the interrupt processing that jump to label specified for <Branch destination> by using <Interrupt number>.

(2) After the starting of the above interrupt monitoring, start tracking on upper position.

(3) In the case of "TrkMv Off", stop the interrupt monitoring specified in "TrkMv On", stop tracking.

(4) <Position data>, <Encoder data>, <Reference position data>, <Encoder number> which is necessary for conventional "Trk On" uses the data in the tracking buffer correspond to <Condition number> specified by "TrkChk" (Buffer number specified by state variable "M_Trkbuf") and the data specified by state variable "P_TrkBase".

(5) The data in the tracking buffer is confirmed by state variable "P_TrkWork","M_TrkEnc","M_TrkKind" and "M_TrkEncNo".

(6) When there is no work piece in back from the starting position of tracking area and this command is executed, L2580 (Workpiece isn't in tracking area) error occurs.

(7) If you omit <Interrupt number> and <Branch destination>, the interrupt processing does not become effective. But you can specify another interrupt processing by using "Def MoTrg" and "Def Act".

(8) If you appoint the label which does not exist as "Branch destination", error L3600_00000(Jump destination does not exist) occurs.

TrkFine(Tracking follow positioning function)

[Function]

The accuracy at the tracking is improved until “TrkFine Off” is executed.

[Format]

TrkFine □ On TrkFine □ Off

[Reference program]

```
M_TrkBuf(1) = 1                          ' <Buffer number> is “1".
P_TrkBase(1) = PTBASE                    ' P_TrkBase(1) variable is PTBASE variable.
・・・・・
'/// Tracking buffer check ///
*LBFCHK
TrkChk 1, PSave, PWait, *LTRST           'No workpiece->PSave/ Wait for the workpiece->PWait/
                                         Tracking possible->Jump to “LTRST”.
If M_TrkChk(1) <= 1 Then GoTo *LBFCHK    ' 0:No workpiece   /   1: Workpiece passed over->“LBFCHK”.
TrkWait *LBFCHK                          ' Wait for the workpiece   / Jump to “LBFCHK” at the timeout.
・・・・・
'/// Start tracking operation ///
*LTRST
TrkFine On                               'Validate TrkFine
TrkMv On, PGTUP, 1, *S91STOP             'Start the interrupt check->Trk On->Move to the tracking upper
                                         position / In the case of exceeding the distance specified by
                                         “M_TrkStop”-Trk Off→Jump to ”S91STOP”
・・・・・ adsorption / Release / assembly etc. ・・・・・
TrkFine Off                              'Invalidate TrkFine
TrkMv Off                                'Stop the interrupt check -> Trk Off
```

[Explanation]

(1) The system default value is TrkFine Off.

(2) When the tracking function valid state (Trk On), the TrkFine command will be ignored even if it is valid (i.e., it will be treated as invalid, but the status will be kept).

(3) When the follow positioning function valid state (TrkFine On), the Cnt command will be ignored even if it is valid (i.e., it will be treated as invalid, but the status will be kept).

(4) When the follow positioning function valid state (TrkFine On), the Fine command will be ignored even if it is valid (i.e., it will be treated as invalid, but the status will be kept).

TrkTrg(Vision sensor trigger)

[Function]

Request the specified vision sensor to capture an image, and acquires encoder value after the SKIP input receives the signal from the vision sensor.

[Format]

TrkTrg □ #<Vision sensor number>, <SKIP input number>, <Encoder 1 value read-out variable> [,
[<Encoder 2 value read-out variable >], [<Encoder 3 value read-out variable >],
[<Encoder 4 value read-out variable >], [<Encoder 5 value read-out variable >],
[<Encoder 6 value read-out variable >], [<Encoder 7 value read-out variable >],
[<Encoder 8 value read-out variable >]]

[Terminology]

<Vision sensor number>

Specify the number of the vision sensor to control.
Setting range：1-8

<SKIP input number>

Specify the number of the SKIP input to control.
Setting range：2-4

<Encoder n value read-out variable>:(Can be omitted from the second one on)

Specifies the double precision numeric variable into which the read out external encoder n value is set.
Note： n is 1-8

[Reference program]

```
If M_NVOpen(1)<>1 Then          'If vision sensor number 1 logon is not complete.
NVOpen "COM2:" AS #1            'Connects with the vision sensor connected to COM2.
EndIf
Wait M_NVOpen(1) = 1            'Connects with vision sensor number 1 and waits for logon to
      be completed.
NVRun #1, "TEST"                'Starts the "TEST" program.
TrkTrg #1, 2, M1#, M2#          'Requests the vision sensor to capture an image and acquires
      encoders 1 and 2 after the SKIP input receives the signal.
EBRead #1,,MNUM,PVS1,PVS2       'The data of "Job.Robot.FormatString" is stored in the read-out
      variable MNUM, PVS1 and PVS2.
 . . . . . .
 . . . . . .
NVClose #1                      'Cuts the line with the vision sensor connected to COM2.
```

[Explanation]

(1) Outputs the image capture request to the specified vision sensor and acquires the encoder value after the SKIP input receives the signal.
(2) The <SKIP input number> specify the number of the SKIP input connected.
(3) The acquired encoder value is stored in the specified numeric variable.
(4) This command moves to the next step after it has received the signal of the image processing completion from the vision sensor.
(5) If the program is cancelled while this command is being executed, it stops immediately.
(6) For receiving data from the vision sensor, use the EBRead command.
(7) When this command is used with multi-tasking, it is necessary to execute the NVOpen command in the task using this command. Also, use the <Vision sensor number> specified with the NVOpen command.
(8) A program start condition of "Always" and the continue function are not supported.
(9) Up to three robots can control the same vision sensor at the same time, but this command can not be used by more than one robot at the same time. Use this command on any one of the robots.
(10) If an interrupt condition is established while this command is being executed, the interrupt processing is executed immediately.
(11) If data type for an argument is incorrect, L.4220 (Syntax error) error occurs.
(12) If there is an abnormal number of command arguments (too many or too few), L.3120 (Illegal argument

(TrkTrg)) error occurs.
(13) If the <Vision sensor number> is anything other than “1” through “8”, L.3110 (Argument value range over (TrkTrg)) error occurs.
(14) If the NVOpen command is not opened with the number specified as the <Vision sensor number>, L.3141 (The NVOPEN is not executed) error occurs.
(15) If the <SKIP input number> is anything other than “2” through “4”, L.3110 (Argument value range over (TrkTrg)) error occurs.
(16) If the same <SKIP input number> is specified by another task, L.8623 (SKIP input number is already used) error occurs.
(17) If the vision program’s image capture specification is set to anything other than “Camera” (all trigger command), “External trigger”, or “Manual trigger”, L.8640 (The image trigger is abnormal) error occurs.
(18) If the vision sensor is “Offline”, L.8640 (The image trigger is abnormal) error occurs, so put the vision sensor “Online”.
(19) If the Communications line is cut while this command is being executed, L.8610 (The communication is abnormal) error occurs and the robot controller side line is closed.

NVOpen(Network vision sensor line open)

[Function]

Connects with the specified vision sensor and logs on to that vision sensor.

[Format]

NVOpen□"<COM number>"□As□#<Vision Sensor number>

[Term]

<Com number> (Can not be omitted):

Specify the communications line number in the same way as for the Open command.
"COM1:" can not be specified by it is monopolized by the operation panel front RS-232C.
Setting range: "COM2:" – "COM8:"

<Vision sensor number> (Can not be omitted)

Specifies a constant from 1 to 8 (the vision sensor number). Indicates the number for the vision sensor connection to the COM specified with the <COM number>.
Be careful. This number is shared with the <file number> of the Open command.
Setting range: 1 – 8

[Sample sentence]

```
If M_NVOpen(1)<>1 Then   'If vision sensor number 1 log on is not complete
 NVOpen "COM2:" As#1   ' Connects with the vision sensor connected to COM2 and sets its number as number 1.
ENDIf
Wait M_NVOpen(1)=1        ' Connects with vision sensor number 1 and waits for logon to be completed.
```

[Explanation]

(1) Connects with the vision sensor connected to the line specified with the <COM number> and logs on to that vision sensor.

(2) It is possible to connect to a maximum of 7 vision sensors at the same time. <Vision sensor numbers> are used in order to identify which vision sensor is being communicated with.

(3) When used together with the Open command, the Open command <COM number> and <File number> and the <COM number> and <Vision sensor number> of this command are shared, so use numbers other than those specified with the Open command <COM number> and <File number>.

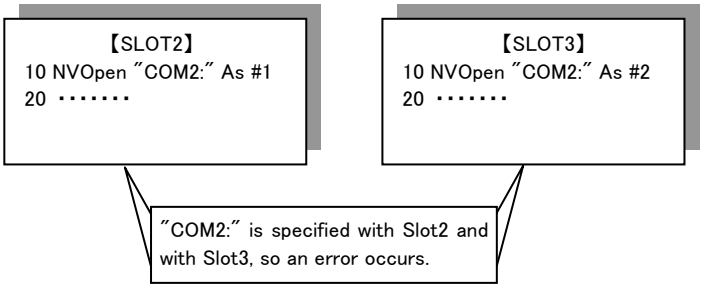
```
Example: Normal example     Error example
 1 Open "COM1:" As #1    1 Open "COM2:" As #1
 2 NVOpen "COM2:" As #2    2 NVOpen "COM2:" As #2 => <COM number> used
 3 NVOpen "COM3:" As #3    3 NVOpen "COM3:" Ass#1 => <Vision sensor number>
                                                    Used
```

It is not possible to open more than one line in a configuration with one robot controller and one vision sensor. If the same IP address is set as when the [NETHSTIP] parameter was set, an "Ethernet parameter NETHSTIP setting" error occurs.

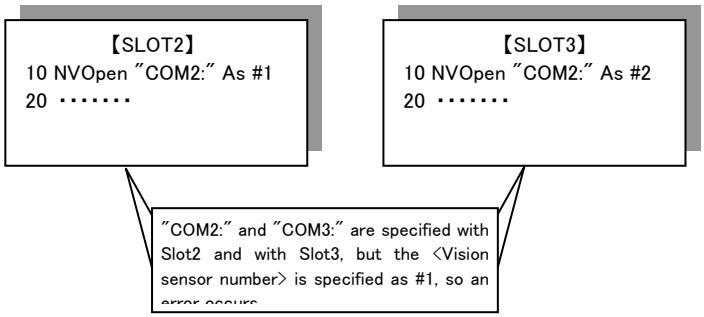
(4) Logging on to the vision sensor requires the "User name" and "Password". It is necessary to set a user name for which full access is set in the vision sensor and the password in the robot controller [NVUSER] and [NVPSWD] parameters.
The user name and password can each be any combination of up to **15** numbers (0-9) and letters (A-Z). **(T/B only supports uppercase letters, so when using a new user, set the password set in the vision sensor with uppercase letters.**)
The user name with full access rights when the network vision sensor is purchased is "admin". The password is "". Therefore, the default values for the [NVUSER] and [NVPSWD] parameters are [NVUSER] = "admin" and [NVPSWD] = "".
When the "admin" password is changed with MELFA-Vision or a new user is registered, change the [NVUSER] and [NVPSWD] parameters. When such a change is made, when the content of the [NVPSWD] parameter is displayed, "****" is displayed. If the vision sensor side password is changed, open the [NVPSWD] parameter and directly change the displayed "****" value. After the making the change, reset the robot controller power.
[Caution]
When multiple vision sensors are connected to one robot controller, set the same user name and password for all of them.

(5) The state of communications with the network vision sensor when this command is executed can be checked with M_NVOpen. For details, see the explanation of M_NVOpen.

(6) If the program is cancelled while this command is being executed, it stops immediately. In order to log on to the vision sensor, it is necessary to reset the robot program, then start.
(7) When this command is used with multi-tasking, there are the following restrictions.
The <COM number> and <Vision sensor number> must not be duplicated in different tasks.
(a) If the same <COM number> is used in another task, the **"attempt was made to open an already open communication file"** error occurs.



(b) If the same vision sensor number is used in another task, the **"attempt was made to open an already open communication file"** error occurs.



(8) A program start condition of "Always" and the continue function are not supported.
(9) Three robots can control the same vision sensor at the same time. If a fourth robot logs on, the line for the first robot is cut off, so be careful when constructing the system.
(10) The line is not closed with an End command in a program called out with a Callp command, but the line is closed with a main program End command. The line is also closed by a program reset.
(11) If an interrupt condition is established while this command is being executed, the interrupt processing is executed immediately even during processing of this command.
(12) If data type for an argument is incorrect, L4220 (syntax error in input command) error is generated.
(13) If there is an abnormal number of command arguments (too many or too few), L3120 (incorrect argument count) error occurs.
(14) If the character specified in <COM number> is anything other than "COM2:" through "COM8:", L3110 (argument out of range) error occurs.
(15) If the value specified as the <vision sensor number> is anything other than "1" through "8", L3110 (argument out of range) error occurs.
(16) If a <COM number> for which the line is already connected is specified (including the <File number> for which the line has been opened with an Open command), L3130 (attempt was made to open an already open communication file) error occurs.
(17) If the vision sensor is not connected before the line is opened, L8600 (vision sensor not connected) error occurs. (The same set manufacturer parameter [COMTIMER] as in the Ethernet specifications is used. Currently "1s")
(18) If the same <COM number> or the same <vision sensor number> is specified in another task, L3130 (attempt was made to open an already open communication file) error occurs.
(19) If the user name or password specified in the [NVUSER] parameter (user name) and [NVPSWD] (password) is wrong, the L8602 (wrong password) error occurs.
(20) If the communications line is cut while this command is being executed, L 8610(abnormal communications) error occurs and the robot controller side line is closed.
(21) If a program is used for which the starting condition is "Always", the L3287 (this command can not be used if the start condition is ERR or ALW) error occurs.

NVClose(Network vision sensor line close)

[Function]

Cuts the line with the specified vision sensor.

[Format]

NVClose□[[#<Vision sensor number>] [,[[#]<Vision sensor number>･･･]]]

[Term]

<Vision sensor number> (Can be omitted)

Specifies a constant from 1 to 8 (the vision sensor number). Indicates the number for the vision sensor connection to the COM specified with the <COM number>.
When this parameter is omitted, all the lines (vision sensor lines) opened with an NVOpen command are closed.
Also, up to 8 <vision sensor numbers> can be specified. They are delimited with commas.
Setting range: 1 - 8

[Sample sentence]

```
If M_NVOpen(1)<>1 Then      ' When logon has not been completed for vision sensor number 1
  NVOpen “COM2:” ASs#1      ' Connects with the vision sensor connected to COM2 and sets its number as
                              number 1.
EneIf
Wait M_NVOpen(1)=1          'Connects with vision sensor number 1 and waits for logon to be completed.
･････
  :
NVClose #1                  'Cuts the line with the vision sensor connected to COM2.
```

[Explanation]

(1) Cuts the line with the vision sensor connected with the NVOpen command.
(2) If the <vision sensor number> is omitted, cuts the line with all the vision sensors.
(3) If a line is already cut, execution shifts to the next step.
(4) Because up to seven vision sensors can be connected at the same time, <Vision sensor numbers> are used in order to identify which vision sensor to close the line for.
(5) If the program is cancelled while this command is being executed, execution continues until processing of this command is complete.
(6) When this command is used with multi-tasking, in the task using this command, it is necessary to close only the lines opened by executing an NVOpen command. At this time, use the <Vision sensor number> specified with the NVOpen command.
(7) A program start condition of "Always" and the continue function are not supported.
(8) If an End command is used, all the lines opened with an NVOpen command or Open command are closed. However, lines are not closed with an End command in a program called out with a CAllp command. Lines are also closed by a program reset, so when an End command or a program reset is executed, it is not necessary to close lines with this command.
(9) The continue function is not supported.
(10) If an interrupt condition is established while this command is being executed, the interrupt processing is executed after this command is completed.
(11) If the value specified as the <vision sensor number> is anything other than "1" through "8", L3110 (argument out of range) error occurs.
(12) If there are more than eight command arguments, L3120 (incorrect argument count) error occurs.

NVLoad(Network vision sensor load)

[Function]

Loads the specified vision program into the vision sensor.

[Format]

NVLoad□#<Vision sensor number>,<Vision program (job) name>

[Term]

<Vision sensor number> (Can not be omitted)

This specifies the number of the vision sensor to control.
Setting range: 1 - 8

<Vision program (job) name> (Can not be omitted)

Specifies the name of the vision program to start.
The vision program extension (.job) can be omitted.
The only characters that can be used are "0" - "9", "A" - "Z", "a" - "z", "-", and "_".

[Sample sentence]

```
If M_NVOpen(1)<>1 Then       'If vision sensor number 1 log on is not complete
  NVOpen “COM2:” As #1     'Connects with the vision sensor connected to COM2.
EndIf
Wait M_NVOpen(1)=1       ' Connects with vision sensor number 1 and waits for logon to be completed.
NVLoad #1,”TEST”             'Loads the "Test".
NVPst #1, ””,”E76”,”J81”,”L84”,0,10
'Receives the recognition count recognized with the "Test" program from the E76 cell and the recognition
                                    results from cells J81 through L84, and stores them in P_NvS1().
....
   :
NVClose #1     'Cuts the line with the vision sensor connected to COM2.
```

[Explanation]

(1) Loads the specified vision program into the specified vision sensor.
(2) This command moves to the next step at the point in time when the vision program is loaded into the vision sensor.
(3) If the program is cancelled while this command is being executed, it stops immediately.
(4) If the specified <vision program name> is already loaded, the command ends with no processing.
(5) When this command is used with multi-tasking, it is necessary to execute the NVOpen command in the task using this command. Also, use the <vision sensor number> specified with the NVOpen command.
(6) A program start condition of "Always" and the continue function are not supported.
(7) If an interrupt condition is established while this command is being executed, the interrupt processing is executed immediately.
(8) If data type for an argument is incorrect, a L4220 (syntax error in input command statement) error is generated.
(9) If there is an abnormal number of command arguments (too many or too few), L3120 (incorrect argument count) error occurs.
(10) If the <vision sensor number> is anything other than "1" through "8", L3110 (argument out of range) error occurs.
(11) If the NVOpen command is not opened with the number specified as the <vision sensor number>, L8620 (abnormal vision sensor number specification) error occurs.
(12) If the <vision program name> exceeds 15 characters, L8621 (abnormal vision program name) error occurs.
(13) If a <vision program name> uses a character other than "0" – "9", "A" – "Z", "-", or "_" (including lowercase letters), L8621 (abnormal vision program name) error occurs.
(14) If the program specified in the <vision program name> is not in the vision sensor, L8622 (vision program does not exist) error occurs.
(15) If the vision sensor is "offline", L8650 (Put online) error occurs, so put the vision sensor "Online".
(16) If the communications line is cut while this command is being executed, an L8610 (abnormal communications) error occurs and the robot controller side line is closed.

EBRead(EasyBuilder Read)

[Function]

Reads out the data by specifying the tag name of the vision sensor.

The data read from the vision sensor is stored in the specified variable.

Please read out data specifying the tag name by using this command when the vision program (job) is made with the vision tool EasyBuilder made by Cognex Corporation.

[Format]

EBRead□#<Vision sensor number>, [<Tag name>], <variable name 1> [, <variable name 2>]...[, <Time out>]

[Term]

<Vision sensor number> (Can not be omitted)

This specifies the number of the vision sensor to control.
Setting range: 1 - 8

<Tag name> (Can be omitted)

Specifies the name of symbolic tag where data read out by the vision sensor is stored .
When omitting it, the value of paraemter EBRDTAG (initial value is the custom format tag name "Job.Robot.FormatString") is set to it.

<variable name>(Can not be omitted)

Specifies the variable where the data read from the vision sensor is stored.
It is possible to use two or more variables by delimited with commas.
It is possible to specify the Numeric value variable, Position variable or String variable.
When the Position variable is specified, the value is set to X, Y, and C element, and 0 is set to other elements.

<Time out> (If omitted, 10)

Specifies the time-out time (in seconds).
Specification range: Integer 1-32767

[Sample sentence]

```
If M_NvOpen(1)<>1 Then      'If vision sensor number 1 log on is not complete
  NVOpen "COM2:" As #1      'Connects with the vision sensor connected to COM2.
End If
Wait M_NvOpen(1)=1        ' Connects with vision sensor number 1 and waits for logon to be completed.
NVLoad #1,"TEST"          'Loads the "Test" program.
TrkTrg #1,2,M1#,M2#           'Starts the "Test" program..
EBRead #1,,MNUM,PVS1,PVS2     'The data of "Job.Robot.FormatString" tag is read,
                                        and they are preserved in the variable MNUM, PVS1, and PVS2.
--------
  :
NVClose #1                    'Cuts the line with the vision sensor connected to COM2.
```

[Explanation]

(1) Gets the data by specifying the tag name from an active vision program in the specified vision sensor.
(2) The data read from the vision sensor is stored in the specified variable.
(3) When the specified variable identifier is delimited by comma and enumerated when the data of the vision sensor is two or more values (character string) delimited by comma, data is stored in order of describing the variable identifier. In this case, the type of the object data should be the same as the type of the variable.
(4) When the position variable is specified, the vision data is stored in X, Y, and C element. And the value of other elements are 0.
The value converted into the radian is set to C element.
(5) The value of receiving data are set only to the specified variables when the number of specified variables is less than that of receive data.
(6) The variable more than the number of receiving data is not updated when the number of specified variables is more than that of receive data.
(7) When the tag name is omitted, the value of parameter EBRDTAG is set instead of the tag name. (The

factory shipment setting is " Job.Robot.FormatString".)

(8) It is possible to specify the timeout time by the numerical value. Within the timeout time, does not move to the next step until the results are received from the vision sensor. However, if the robot program is stopped, this command is immediately cancelled. Processing is continued with a restart.

(9) When this command is used with multi-tasking, it is necessary to execute the NVOpen command in the task using this command. In this case, use the <vision sensor number> specified with the NVOpen command.

(10) A program start condition of "Always" and the continue function are not supported.

(11) If an interrupt condition is established while this command is being executed, the interrupt processing is executed immediately even during processing of this command. The processing is executed after completing the interrupt processing.

< Value of the variable>

The variable by executing the EBRead command is as follows.

(A) Content of specified tag (Pattern_1.Number_Found) is 10

(a) The value when "EBRead #1,"Pattern_1.Number_Found",MNUM" is executed is :

-> MNUM=10

(b) The value when "EBRead #1,"Pattern_1.Number_Found",CNUM" is executed is :

-> CMNUM="10"

(B) Content of specified tag (Job.Robot.FormatString) is 2, 125.75, 130.5, -117.2, 55.1, 0, 16.2

(a) The value when "EBRead #1,,MNUM,PVS1,PVS2" is executed is :

-> MNUM=2

PVS1.X=125.75 PVS1.Y=130.5 PVS1.C=-117.2

PVS2.X=55.1 PVS2.Y=0, PVS2.C=16.2

* The element (Excluding X and Y element) that the vision data is not set is 0.

(b) The value when "EBRead #1,,MNUM,MX1,MY1,MC1,MX2,MY2,MC2" is executed is :

-> MNUM=2

MX1=125.75 MY1=130.5 MC1=-117.2

MX2=55.1 MY2=0 MC2=16.2

(c) The value when "EBRead #1,,CNUM,CX1,CY1,CC1,CX2,CY2,CC2" is executed is :

-> CNUM="2"

CX1="125.75" CY1="130.5" CC1="-117.2"

CX2="55.1" CY2="0" CC2="16.2"

(C) Content of specified tag (Job.Robot.FormatString) is 2, 125.75, 130.5

(a) The value when "EBRead #1,,MNUM,PVS1" is executed is :

-> MNUM=2

PVS1.X=125.75 PVS1.Y=130.5

* The element (Excluding X and Y element) that the vision data is not set is 0.

(12) If data type for an argument is incorrect, L4220 (syntax error in input command statement) error is generated.

(13) If there is an abnormal number of command arguments (too many or too few), L3120 (incorrect argument count) error occurs.

(14) If the <vision sensor number> is anything other than "1" through "8", L3110 (argument out of range) error occurs.

(15) If the NVOpen command is not opened with the number specified as the <vision sensor number>, L3141 (The NVOpen command is not executed.) error occurs.

(16) If data type of the strings data received from the vision sensor and the variable substituted for it is difference, L3501 (Illegal Receive data(EBREAD)) error is generated.

(17) If the <Timeout> is other than "1" – "32767", L3110 (argument out of range) error occurs.

(18) If the vision sensor does not respond without the time specified as the <Timeout> or within the first 10 seconds if the <Timeout> parameter is omitted, L8632 (vision sensor response timeout) error occurs.

(19) If the communications line is cut while this command is being executed, L8610 (abnormal communications) error occurs and the robot controller side line is closed.

(20) If the specified tag name does not exist in the active vision program, L8636 (Vision Tag name is abnormal) error is generated.

(21) Please specify 31 variables or less

('number of the recognition' +' position in the coordinate (X,Y,Z)' x 10) .

If 32 variables or more are specified, L4220 (syntax error in input command statement) error is generated.
(22) If the <vision program name> exceeds 15 characters, L8621 (abnormal vision program name) error occurs.
(23) If a <vision program name> uses a character other than "0" – "9", "A" – "Z", "-", or "_" (including lowercase letters), L8621 (abnormal vision program name) error occurs.
(24) If the program specified in the <vision program name> is not in the vision sensor, L8622 (vision program does not exist) error occurs.
(25) If the <Recognition count cell>, <Start cell>, or <End cell> contains a number other than "0" – "399" or a letter other than "A – "Z", L3110 (argument out of range) error occurs.
(26) If there is no value in the cell specified in "Recognition count cell", L8630 (invalid value in specified for recognition count cell) error occurs.
(27) If the <Start cell> and <End cell> are reversed, L8631 (specified cell value out of range) error occurs.
(28) If the number of data included in the cell which specifies it by <Start cell> and <End cell> exceeds 90, L8631 (specified cell value out of range) error occurs.
(29) If the range specified by <Star cell> and <End cell> exceeds line 30 and row 10, L8631 (specified cell value out of range) error occurs.
(30) If the <Type> is other than "0" - "7", L3110 (argument out of range) error occurs.

M_Enc (Encoder value)

[Function]

Read the encoder value of the designated logic encoder number. It can be changed to the optional value.

[Format]

[Write]
M_Enc(<logic encoder number>) ＝ <Fixed value>
[Read]
<Numeric value> ＝ M_Enc(<logic encoder number>)

[Terminology]

< logic encoder number [integer]**>**：(can be omitted.)
Specify the logic encoder number which acquires the encoder value.
Setting range: 1 to 8
If the argument is omitted, 1 is set as the default value.

< Fixed value [double-precision real number]**>**
Specify the numerical value.

< Numeric value [double-precision real number]**>**
Specify the numeric variable in which the value.

[Reference program]

MENC1#=M_Enc(1)	'Stocks the logic encoder number encoder of 1 value in MENC1# variable.
MENC2#=M_Enc(M1%)	'Stocks the encoder value of the logic encoder number designated by M1% variable in MENC2# variable.
TrWrt P1, M_Enc(1), MK	' This variable writes in buffer 1 that the location of the workpiece which was kind number MK is P1 at the present encoder value M_Enc (1).
M_Enc(1)=0	'Changes the encoder value of the logic encoder No.1 to "0".

[Explanation]

(1) Acquire the encoder value of the designated <logic encoder number>.
The acquired encoder value is written in a tracking buffer using a TrWrt command to tracking movement.
(2) The encoder value is the double-precision real number value, so please specify a variable of double-precision real number type as<Numeric value>.
(3) It's possible to change the encoder value of the number specified as<logic encoder number> to the value specified as<Fixed value>.
(4) You can omit the step to specify <logic encoder number>.When it is omitted, logic encoder number will be treated as "1."
(5) Error L.3110 (value of the argument outside of the range) occurs when <Condition number> is outside a set range.
*When inputting the numerical value including the decimal point, its value is rounded up.

M_EncL (Latched Encoder value)

[Function]

At the instant of receipt of a TREN signal for Q17EDPX module, a stored encoder data is read.
Also, 0 is written to clear the stored encoder data to zero.

[Format]

[Write]
M_EncL(<logic encoder number>) = <Fixed value>
[Read]
<Numeric value> = M_EncL(<logic encoder number>)

[Terminology]

<Logic encoder number [Integer]**> :**(can be omitted)
Specify the value of logic encoder number

< Fixed value [double-precision real number]**>**
Specify the stored encoder data to initial value(zero or other).

<Numeric variable [double-precision real number]**>**
Specify the numerical variable to substitute.

[Reference program]

1 MENC1#=M_EncL(1)	At logic encoder number 1, assign encoder data stored at the time of receipt of a TREN signal to the variable MENC1#.
2 MENC2#=M_EncL(M1%)	At a logic encoder number specified in the variable M1%, assign encoder data stored at the time of receipt of a TREN signal to the variable MENC2#.
3 TrWrt P1, M_EncL(1), MK	Write workpiece position data P1, encoder value M_EncL(1) present at the time of receipt of a TREN signal and work category number MK onto the buffer 1 for tracking.
4 M_EncL(1)=0	Use latched data to clear the encoder to zero as it is not required until next latched data is used.

[Explanation]

(1) Stored encoder value corresponding to the encoder number specified for <logical encoder number> is acquired.
Encoder value is stored in memory at a low-to-high or high-to-low transition of TREN signal which has been specified with a DIP switch on Q173DPX module.
Encoder value thus acquired is written onto the buffer for tracking by using a TrWrk command so as to perform tracking operations.

(2) As encoder value is in double-precision real number, specify <Numerical variable> with a variable which is of double-precision real-number type.

(3) You can omit the step to specify <Logic encoder number>. When it is omitted, logic encoder number will be treated as "1."

(4) Number which you can enter to specify <Logic encoder number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
* If a number having a decimal part is entered, the fraction of 0.5 or over will be counted as one and the rest will be cut away.

(5) As latched encoder data represents a value present at a low-to-high or high-to-low transition of TREN signal, you should check input corresponding to input number in 810 to 817 range which has been assigned to TREN signal when reading it out.

(6) You can clear the encoder to zero by typing "0" after having used latched encoder data. This step may be performed as a precaution against using previously latched data.

P_EncDlt(The encoder amount of movement)

[Function]

Set the amount of robot movement per encoder pulse.
Or, the amount of robot movement per encoder pulse is returned.
The amount of robot movement : Straight line tracking :(X, Y, Z, 0, 0, 0, L1, L2)

[Format]

[Write]
P_EncDlt(<Encoder number>) = <Position Data>
[Read]
<Position Variables> = P_EncDlt(<Encoder number>)

[Terminology]

<Encoder number [Integer]**>**: (can be omitted.)
Specify a logic number indicating the external encoder that performs tracking operation.
Setting range: 1 to 8
If the argument is omitted, 1 is set as the default value.

<Position Data [Position]**>**
Specify the amount of robot movement per encoder pulse.

<Position Variables [Position]**>**
Specify a position variable that stores amount of robot movement per encoder pulse.

[Reference Program]

```
P_EncDlt(1) = P1          'Amount of robot movement per encoder pulse of encoder number 1 is set.
P2 = P_EncDlt(2)          'Amount of robot movement per encoder pulsee of encoder number 2 is stored in positional variable.
```

[Explanation]

(1) The amount of robot movement per encoder pulse of specified <Encoder number> is set. Or, the amount of robot movement per encoder pulse is returned.
(2) You can omit the step to specify <logic encoder number>.When it is omitted, logic encoder number will be treated as "1."
(3) Error L.3110 (value of the argument outside of the range) occurs when <Encoder number> is outside a set range.

P_TrkPAcl

[Function]

Change the tracking acceleration coefficient of the parameter “TRPACL” temporarily.

[Format]

[Writing]
P_TrkPAcl(<Condition number>) = <Position data>
[Referencing]
<Position variable> = P_TrkPAcl(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

<Position data [Position]**>**
Specify the tracking acceleration coefficient.
Setting area: For each component, 0.10 to 10.0

<Position variable [Position]**>**
Return the specified tracking acceleration coefficient.

[Reference program]

```
P_TrkPAcl(1) = (0.2, 0.2, 0.2, 1.0, 1.0, 1.0, 1.0, 1.0)  'Specify the tracking acceleration coefficient.
P_TrkPDcl(1) = (0.2, 0.2, 0.2, 1.0, 1.0, 1.0, 1.0, 1.0)  ' Specify the tracking deceleration coefficient.
  .....
*LTRST
TrkMv On, PGTUP, 1, *S91STOP  'Start the interrupt processing->Trk On-> Move to the tracking upper position
```

[Explanation]

(1) Specify the tracking acceleration coefficient used in tracking command “TrkMv”.
(2) You can confirm the tracking acceleration coefficient by referencing “P_TrkPAcl”.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

P_TrkPDcl

[Function]

Change the tracking deceleration coefficient of the parameter “TRPDCL” temporarily.

[Format]

[Writing]
P_TrkPDcl(<Condition number>) = <Position data>
[Referencing]
<Position variable> = P_TrkPDcl(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting area: 1 to 8

<Position data [Position]**>**
Specify the tracking deceleration coefficient.
Setting area: For each component, 0.1 to 10.0

<Position variable [Position]**>**
Return the specified tracking deceleration coefficient

[Reference program]

```
P_TrkPAcl(1) = (0.2, 0.2, 0.2, 1.0, 1.0, 1.0, 1.0, 1.0)  ' Specify the tracking acceleration coefficient.
P_TrkPDcl(1) = (0.2, 0.2, 0.2, 1.0, 1.0, 1.0, 1.0, 1.0)  ' Specify the tracking deceleration coefficient.
  .....
*LTRST
TrkMv On, PGTUP, 1, *S91STOP  'Start the interrupt processing->Trk On-> Move to the tracking upper position
```

[Explanation]

(1) Specify the tracking deceleration coefficient used in tracking command “TrkMv”.
(2) You can confirm the tracking deceleration coefficient by referencing “P_TrkPDcl”.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

M_TrkBuf

[Function]

Specify and refer to the tracking buffer number to use.

[Format]

[Writing]
M_TrkBuf(<Condition number>) = <Value>
[Referencing]
<Numeric variable> = M_TrkBuf(<Condition number>)

[Terminology]

<Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

<Value [Integer]**>**
Specify the tracking buffer number.
Setting range: 1 to the first argument of parameter “TRBUF”.
The initial value of “TRBUF” is 2, the maximum value of “TRBUF” is 8.

<Numeric variable [Integer]**>**
Return the specified tracking buffer number.

[Reference program]

```
M_TrkBuf(1) = 1              'The tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, P1, PWAIT, *LTRST  'Check the workpiece in the tracking buffer which is specified.
```

[Explanation]

(1) Specify the tracking buffer number used in tracking command “TrkChk”,”TrkWait”,”TrkMv”.
(2) You can confirm the specified tracking buffer number by referencing “M_TrkBuf”.
(3) If the tracking buffer number is not specified by using “M_TrkBuf”before executing “TrkChk” command, tracking number will be treated as “1”.
(4) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(5) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(6) Number which you can enter to specify <Value> is an integer in the range of "1" to " the first argument of parameter “TRBUF” ". Entering anything else causes L3110 (Argument value range over) error to occur.

M_TrkStart

[Function]

Specify and refer to the starting position of range in which it is possible to execute the tracking.
Starting position specifies the coordinates from the reference point (coordinate value"0.00") in the world coordinates system.

[Format]

[Writing]
M_TrkStart(＜Condition number＞) = ＜Value＞
[Referencing]
＜Numeric variable＞ = M_TrkStart(＜Condition number＞)

[Terminology]

< Condition number >

Specify the condition number corresponding to the tracking.
Setting range : 1 ～ 8

< Value >

Specify the starting position (mm) of range in which it is possible to execute the tracking.
Starting position is the coordinates from the reference point (coordinate value"0.00") in the world coordinates system.

Setting range : 0.00 ～ (Robot operation range)
Unit : mm

< Numeric variable >

Return the starting position of range in which it is possible to execute the tracking..

[Reference program]

```
M_TrkBuf(1) = 1              ' Tracking buffer corresponding to the condition number 1 uses number 1.
M_TrkStart(1) = 300          ' Starting position of range in which it is possible to execute the tracking
                             corresponding to condition number 1 is 300mm.
.....
TrkChk 1, P1, PWAIT, *LTRST  ' Check the workpiece of the specified tracking buffer.
```

[Explanation]

(1) Specify the starting position of range in which it is possible to execute the tracking used in tracking command "TrkChk""TrkWait".
(2) You can confirm the specified starting position of range in which it is possible to execute the tracking by referencing "M_TrkStart".
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

M_TrkEnd

[Function]

Specify and refer to the ending position of range in which it is possible to execute the tracking.
Ending position specifies the coordinates from the reference point (coordinate value"0.00") in the world coordinates system.

[Format]

[Writing]
Example) M_TrkEnd(<Condition number>) = ＜Value＞
[Referencing]
Example) <Numeric variable> = M_TrkEnd(<Condition number>)

[Terminology]

< Condition Number >

Specify the condition number corresponding to tracking.
Setting range ： 1 ～ 8

<Value>

Specify the ending position (mm) of range in which it is possible to execute the tracking.
Ending position is the coordinates from the reference point (coordinate value"0.00")
in the world coordinates system.
Setting range ： 0.00 ～ （Robot operation range）
Unit ： mm

< Numeric Variable >

Return end position of range in which it is possible to execute the tracking..

[Reference program]

```
M_TrkBuf(1) = 1          ' Tracking buffer corresponding to the condition number 1 uses number 1.
M_TrkStart(1) = 300      ' Starting position of range in which it is possible to execute the tracking
                         corresponding to the condition number 1 is 300mm.
M_TrkEnd(1) = -100       ' End position of range in which it is possible to execute the tracking
                         corresponding to the condition number 1 is -100mm.
.....
TrkChk 1, P1, PWAIT, *LTRST   ' Check the workpiece of the specified tracking buffer
```

[Explanation]

(1) Specify the ending position of range in which it is possible to execute the tracking used in tracking command "TrkChk""TrkWait".
(2) You can confirm the specified ending position of range in which it is possible to execute the tracking by referencing "M_TrkEnd".
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

M_TrkStop

[Function]

Specify and refer to forced ending position of range in which it is possible to execute the tracking.
Forced ending position specifies the coordinates from the reference point (coordinate value"0.00")
In the world coordinates system.

[Format]

[Writing]
Example) M_TrkStop(<Condition number>) = ＜Value＞
[Referencing]
Example) <Numeric variable> = M_TrkStop(<Condition number>)

[Terminology]

< Condition Number >

Specify the condition number corresponding to tracking.
Setting range ： 1 ～ 8

<Value>

Specify the forced ending position (mm) of range in which it is possible to execute the tracking.
Forced ending position is the coordinates from the reference point (coordinate value"0.00") in the world coordinates system.
Setting range ： 0.00 ～ （Robot operation range）
Unit ： mm

< Numeric Variable >

Return forced ending position of range in which it is possible to execute the tracking..

[Reference program]

```
M_TrkBuf(1) = 1          ' Tracking buffer corresponding to the condition number 1 uses number 1.
M_TrkStart(1) = 300      ' Starting position of range in which it is possible to execute the tracking
                         corresponding to condition number 1 is 300mm.
M_TrkEnd(1) = -100       ' End position of range in which it is possible to execute the tracking
                         corresponding to condition number 1 is -100mm.
M_TrkStop(1) = -200      ' Forced ending position of range in which it is possible to execute the tracking
                         corresponding to condition number 1 is -200mm.
.....
TrkChk 1, P1, PWAIT, *LTRST  ' Check the work of the specified tracking buffer
```

[Explanation]

(1) Specify the forced ending position of range in which it is possible to execute the tracking used in tracking command "TrkChk""TrkWait".
(2) You can confirm the specified forced ending position of range in which it is possible to execute the tracking by referencing "M_TrkStop".
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

M_TrkTime

[Function]

Specify and refer to the timeout value for “TrkWait” command.

[Format]

[Writing]
M_TrkTime(<Condition number>) = <Value>
[Referencing]
<Numeric variable> = M_TrkTime(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

<Value [Single-precision real number]**>**
Specify the timeout time waits until the workpiece enters to range in which it is possible to execute the tracking..
Setting range: 0.00 to
Unit: second

< Numeric Variable [Single-precision real number]**>**
Return specified tracking buffer number.

[Reference program]

```
M_TrkTime(1) = 60                    'Set the timeout time to 60 second.
.....
TrkChk 1, PSave, PWait, *LTRST   ' No workpiece->PSave/ Waits for the workpiece->PWait/Workpiece can
                                     be followed by tracking->*LTRST
If M_TrkChk(1) <= 1 Then GoTo *LBFCHK   '0:No workpiece/1:Workpiece passed over->Jump to *LBFCHK.
TrkWait *LBFCHK                      'Waits until workpiece enters to the tracking area
```

[Explanation]

(1) Specify the timeout time used in tracking command “TrkWait”.
(2) You can confirm the specified timeout time by referencing “M_TrkStop”.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

P_TrkBase

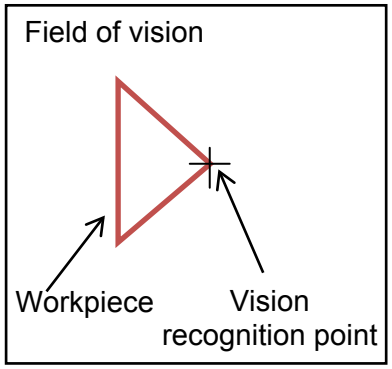
[Function]

Specify and refer to the origin (For example, the position which a vision sensor outputs) of the workpiece to be followed.

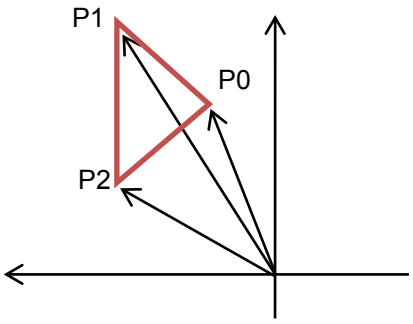
Specify the position data (For example, the position which a vision sensor outputs) used as the reference point when you teach the movement path on the workpiece, as described below

The robot moves to the relative position correspond to this reference point by the movement instruction during the tracking.

[Vision recognition position]



[Teaching position]



Teach three position P0,P1,P2

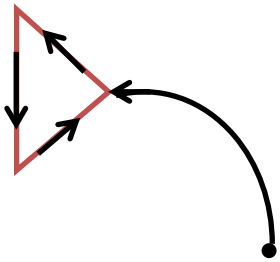
[Robot program]

```
P_TrkBase(1)=P0
  ......
    Mvs    P1
    Mvs    P2
    Mvs    P0


    TrkMv  Off
```

[Robot operation]

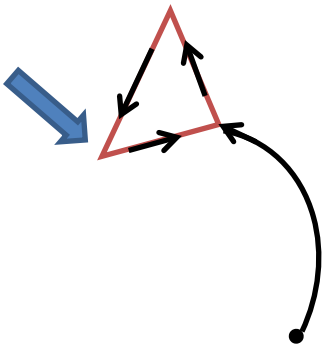
While following…



[Structure]

Regard the position outputted by vision as P0 by “P_TrkBase(1)=P”, “TrkBase P0”
Follow “Mvs P1”, “Mvs P2” as the reference position from P0.

If the workpiece Declines, P0 Inclines too, and P1,P2 declines correspond to P0.



For example, when you only absorb the workpiece (do not operate along the external form of the workpiece), you may appoint the position specified when you teach the position which absorb the workpiece (for example, “PTeach”)as “P_TrkBase”,and appoint the above “PTeach” as movement instruction that moves during “TrkOn ～TrkMv Off”(Mov PTeach).

[Format]

[Writing]
P_TrkBase(<Condition number>) = <Position data>
[Referencing]
<Position variable> = P_TrkBase(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

<Position data [Position]**>**
Specify the base position of the tracking.

<Position variable [Position]**>**
Return the base coordinates of the specified tracking.

[Reference program]

```
P_TrkBase(1) = PTBASE     'Specify the tracking base.
  .....
*LTRST
TrkMv On, PGTUP, 1, *S91STOP  'Start the interrupt processing->Trk On->Move to the tracking upper
                              position
```

[Explanation]

(1) Specify the workpiece coordinate system origin used in tracking command “TrkMv”.
(2) You can confirm the workpiece coordinate system origin by referencing “P_TrkBase”.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

M_TrkChk

[Function]

Refer to the workpiece state read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

[Format]

[Referencing]
<Numeric variable> = M_TrkChk(<Condition number>)

[Terminology]

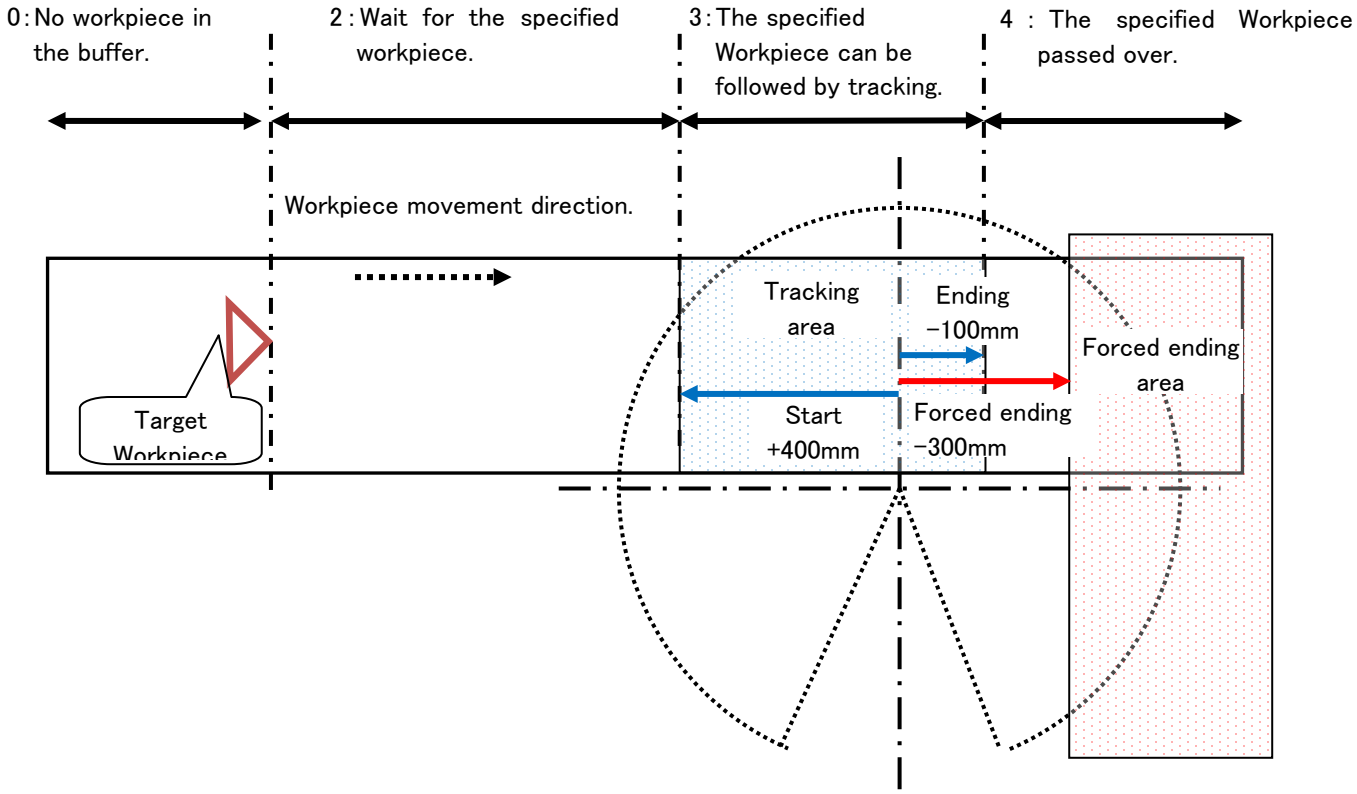
< Condition number [Integer]**>**

Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

< Numeric variable [Integer]**>**

Return the workpiece state read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

0 : No workpiece in the buffer.
1 : The specified workpiece passed over.
2 : Wait for the specified workpiece.
3 : The specified workpiece can be followed by tracking.



[Reference program]

```
M_TrkBuf(1) = 1                ' Tracking buffer corresponding to the condition number 1 uses number 1.
・・・・・
*LBFCHK
TrkChk 1, PSave, PWait, *LTRST ' Check the workpiece of the specified tracking buffer.
If M_TrkChk(1) <= 1 Then GoTo *LBFCHK   '0:No Workpiece/ 1: Workpiece passed over->Jump to
                                        “LBFCHK”.
```

[Explanation]

(1) You can confirm the workpiece state read from the tracking buffer when “TrkChk”, “TrkWait” command is executed..
(2) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as

"1."

(3) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.

(4) When you execute the writing to “M_TrkChk”, L3210 (This variable is write protected) error occurs.

P_TrkWork

[Function]

Refer to the workpiece position read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

[Format]

[Referencing]
<Position type variable> = P_TrkWork(<Condition number>)

[Terminology]

< Condition Number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

<Position variable [Position]**>**
Return the workpiece position read from the tracking buffer corresponding to the specified condition number.

[Reference program]

```
M_TrkBuf(1) = 1                ' Tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, PSave, PWait, *LTRST 'Check the workpiece of the specified tracking buffer.
.....
PWrk = P_TrkWork(1)            'Substitute the workpiece position read from the tracking buffer 1.
```

[Explanation]

(1) You can confirm the workpiece position read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.
(2) If there is no data in the tracking buffer, the data will be cleared.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(5) If you execute the writing to “P_TrkWork”, L3210 (This variable is write protected) error occurs.

M_TrkEnc

[Function]

Refer to the encoder value read from the tracking buffer when the “TrkChk”, “TrkWait” command is executed.

[Format]

[Referencing]
<Numeric variable> = M_TrkEnc(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

< Numeric variable [Long-precision real number]**>**
Return the encoder value (pulse) read from the tracking buffer correspond to the specified condition number.

[Reference program]

```
M_TrkBuf(1) = 1                 ' Tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, P1, PWAIT, *LTRST  'Check the workpiece of the specified tracking buffer.
.....
MEnc& = M_TrkEnc(1)             ' Substitute the workpiece position read from the tracking buffer 1.
```

[Explanation]

(1) You can confirm the encoder value read from the tracking buffer when the “TrkChk”, “TrkWait” command is executed.
(2) If there is no data in the tracking buffer, the data will be cleared.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(5) If you execute the writing to “M_TrkEnc”, L3210 (This variable is write protected) error occurs.

M_TrkKind

[Function]

Refer to the model number read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

[Format]

[Referencing]
<Numeric variable> = M_TrkKind(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

< Numeric variable [Long-precision real number]**>**
Return the model number read from the tracking buffer correspond to the specified condition number.

[Reference program]

```
M_TrkBuf(1) = 1                 ' Tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, P1, PWAIT, *LTRST  ' Check the workpiece of the specified tracking buffer.
.....
MKind = M_TrkKind(1)            ' Substitute the model number read from the tracking buffer 1.
```

[Explanation]

(1) You can confirm the model number read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.
(2) If there is no data in the tracking buffer, the data will be cleared.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(5) If you execute the writing to “M_TrkKind”, L3210 (This variable is write protected) error occurs.

M_TrkEncNo

[Function]

Refer to the encoder number read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

[Format]

[Referencing]
<Numeric variable> = M_TrkEncNo(<Condition number>)

[Terminology]

< Condition number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

< Numeric variable [Long-precision real number]**>**
Return the encoder number read from the tracking buffer correspond to the specified condition number.

[Reference program]

```
M_TrkBuf(1) = 1                 ' Tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, P1, PWAIT, *LTRST  ' Check the workpiece of the specified tracking buffer.
.....
MEncNo = M_TrkEncNo(1)       ' Substitute the encoder number read from the tracking buffer 1.
```

[Explanation]

(1) You can confirm the encoder number read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.
(2) If there is no data in the tracking buffer, the data will be cleared.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(5) If you execute the writing to “M_TrkEncNo”, L3210 (This variable is write protected) error occurs.

P_TrkPixel

[Function]

Refer to the workpiece pixel position read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

[Format]

[Referencing]
<Position type variable> = P_TrkPixel(<Condition number>)

[Terminology]

< Condition Number [Integer]**>**
Specify the condition number corresponding to the tracking.
Setting range: 1 to 8

<Position variable [Position]**>**
Return the workpiece pixel position read from the tracking buffer corresponding to the specified condition number.

[Reference program]

```
M_TrkBuf(1) = 1                 ' Tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, PSave, PWait, *LTRST 'Check the workpiece of the specified tracking buffer.
.....
Pixel = P_TrkPixel(1)           'Substitute the workpiece pixel position read from the tracking buffer 1.
```

[Explanation]

(1) You can confirm the workpiece pixel position read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.
(2) If there is no data in the tracking buffer, the data will be cleared.
(3) You can omit the step to specify <Condition number>.When it is omitted, condition number will be treated as "1."
(4) Number which you can enter to specify <Condition number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(5) If you execute the writing to “P_TrkPixel”, L3210 (This variable is write protected) error occurs.

P_TrkTarget

[Function]

Refer to the information (“P_TrkWork”, “M_TrkEnc”) read from the tracking buffer when “TrkChk”, “TrkWait” command is executed, and the current workpiece position calculated by the state variable “P_EncDlt”.

[Format]

[Referencing]
<Position variable> = P_TrkTarget

[Terminology]

<Position variable>

Return the information (P_TrkWork, M_TrkEnc) read from the tracking buffer when “TrkChk”, “TrkWait” command is executed, and the current workpiece position calculated from the state variable “P_EncDlt”.

[Reference program]

```
M_TrkBuf(1) = 1                 ' Tracking buffer corresponding to the condition number 1 uses number 1.
.....
TrkChk 1, P1, PWAIT, *LTRST     ' Check the workpiece of the specified tracking buffer.
.....
PWrkNow = P_TrkTarget           ' Substitute the current workpiece position.
```

[Explanation]

(1) You can confirm the current workpiece position by referencing the information read from the tracking buffer when “TrkChk”, “TrkWait” command is executed.

(2) If you execute the writing to “M_TrkTarget”, L3210 (This variable is write protected) error occurs.

M_Trbfct

[Function]

Refer to the number of workpieces which exists in a designated buffer.

[Format]

[Referencing]
< Numeric value > = M_Trbfct(<Buffer number>)

[Terminology]

<Buffer number [integer]**>** : (can be omitted.)
Specify the tracking buffer number.
Setting range : 1 to the 1st argument of a parameter "TRBUF"
If the argument is omitted, 1 is set as the default value

< Numeric value [integer]**>**
The number of workpieces in the designated buffer is returned to< Buffer number>.

[Reference program]

```
MWrk = M_Trbfct(1)          'The number of works in number 1 of tracking buffer is stocked in variable MWrk.
```

[Explanation]

(1) You can confirm the number of works in the designated buffer.
(2) You can omit the step to specify <Buffer number>.When it is omitted, buffer number will be treated as "1."
(3) Error L.3110 (value of the argument outside of the range) occurs when <Buffer number> is outside a set range.

P_CvSpd

[Function]

Return the conveyer speed.

[Format]

[Referencing]
< Position variable > = P_CvSpd(<Logic encoder number >)

[Terminology]

<Logic encoder number [integer]**>** : (can be omitted.)

Specify the number of logic encoders which do a chase movement.
Setting range: 1 to 8
If the argument is omitted, 1 is set as the default value

<Position variable [position]**>**

Return the conveyer speed.
In case of the high speed and accuracy tracking function, returns the rate in each coordinate of (X, Y, Z, 0, 0, 0, L1, L2).
（When a conveyor is arranged slantingly, the value enters X,Y,Z.）

[Reference program]

```
PCvSpd = P_CvSpd(1)                    ' Stocks the speed of logic encoder No 1 in a PCvSpd variable
```

[Explanation]

(1) Refer to speed of the conveyer and the turntable.
(2) You can omit the step to specify <Logic encoder number>.When it is omitted, logic encoder number will be treated as "1."
(3) Error L.3110 (value of the argument outside of the range) occurs when <Logic encoder number> is outside a set range.
(4) This variable is read only.

M_Hnd

[Function]

Set and refer to the hand open/close states corresponding to the specified <Hand number>.
The contents of processing of this variable are same as HOpen and HClose, but it's used for a<processing> part of Wth / WthIf join mainly.

[Format]

[Writing]
M_Hnd(<Hand number>) = <Value>
[Referencing]
<Numeric variable> = M_Hnd(<Hand number>)

[Terminology]

< Hand number [Integer]**>**
Specify the hand number: (cannot be omitted).
Setting area: 1 to 8

<Value [Integer]**>**
Describe the hand open/close states by numeric variable, constants, or numeric operation expression.
0 : Hand close
1 : Hand open

< Numeric Variable [Integer]**>**
Specify the numeric variable which stores the hand open/close states.
-1 : Undefined hand
0 : Hand close
1 : Hand open

[Reference program]

```
1 Mov P1, 50      ' Move 50mm to Z direction in the tool coordinates system of P1 by Joint interpolation movement.
2 Mvs P1 WthIf M_Ratio > 50, M_Hnd(1) = 0   ' Close the hand of the hand number 1 if it comes to 50% of distance of the purpose position during the movement to P1.
3 *Label : If M_Hnd(1) = 1 Then GoTo *Label  ' Wait until the hand of the hand number 1 closes.
```

[Explanation]

(1) Change and refer to the hand open/close states.
(2) Writing to “M_Hnd” is treated as the processing equal to the HOpen instruction /HClose instruction.
(3) You can make a statement on <Additional condition>/<Processing> of accompanying instruction to the operation instruction.
(4) Initial value just after the power supply obeys the setting value of the parameter “HANDTYPE” or “HANDINIT” (Output signal number 900 to 907),or “ORS***”(General-purpose output signal).
(5) If you appoint the hand number which is not specified by the parameter “HANDTYPE”, it becomes no processing at the time of writing, and -1 (Undefined hand) returns at the time of reading.
(6) If the hand of specified < hand number> is Double solenoid (D) setting, and output signal state is neither hand open(&B01) nor hand close($B10), return 1(hand open).
(7) You can omit the step to specify <Hand number>.When it is omitted, L3110 (Argument value range over) error occurs.
(8) Number which you can enter to specify <Hand number> is an integer in the range of "1" to "8." Entering anything else causes L3110 (Argument value range over) error to occur.
(9) Number which you can enter to specify <Value> is an integer “0” or “1”. Entering anything else causes L3110 (Argument value range over) error to occur.
(10) If you write “M_Hnd” by using the task slot which does not acquire a machine control rights, L3280 (Cannot execute without GETM) error occurs.
(11) If you read “M_Hnd” by using the task slot which does not acquire a machine control rights, return the robot hand open/close states of machine number 1.
(12) It is impossible to use for the electric hand with many functions made in TAIYO company.
Please refer to the explanation of “Usage of the electric hand with many functions”.
(13) “M_Hnd” does not correspond to the hand macro.

M_NvOpen

[Function]

Indicates the vision sensor line connection status.

[Format]

<Numeric value> = M_NvOpen(<Vision sensor number>)

[Terminology]

<Vision sensor number>

This specifies the number of the vision sensor to control.
Setting range: 1 - 8

<Numeric value>

Indicates the vision sensor line connection status.
-1 : Not connected
0 : Line connecting (Logon not complete)
1 : Logon complete

[Sample sentence]

```
If M_NVOpen(1)<>1 Then  ' If vision sensor number 1 is not connected
 NVOpen “COM2:” As #1    ' Connects with the vision sensor connected to COM2 and sets its number as number 1.
EndIf
Wait M_NVOpen(1)=1          ' Connects with vision sensor number 1 and waits for the logon state.
・・・・・
 :
NVClose #1                  'Cuts the line with the vision sensor connected to COM2.
```

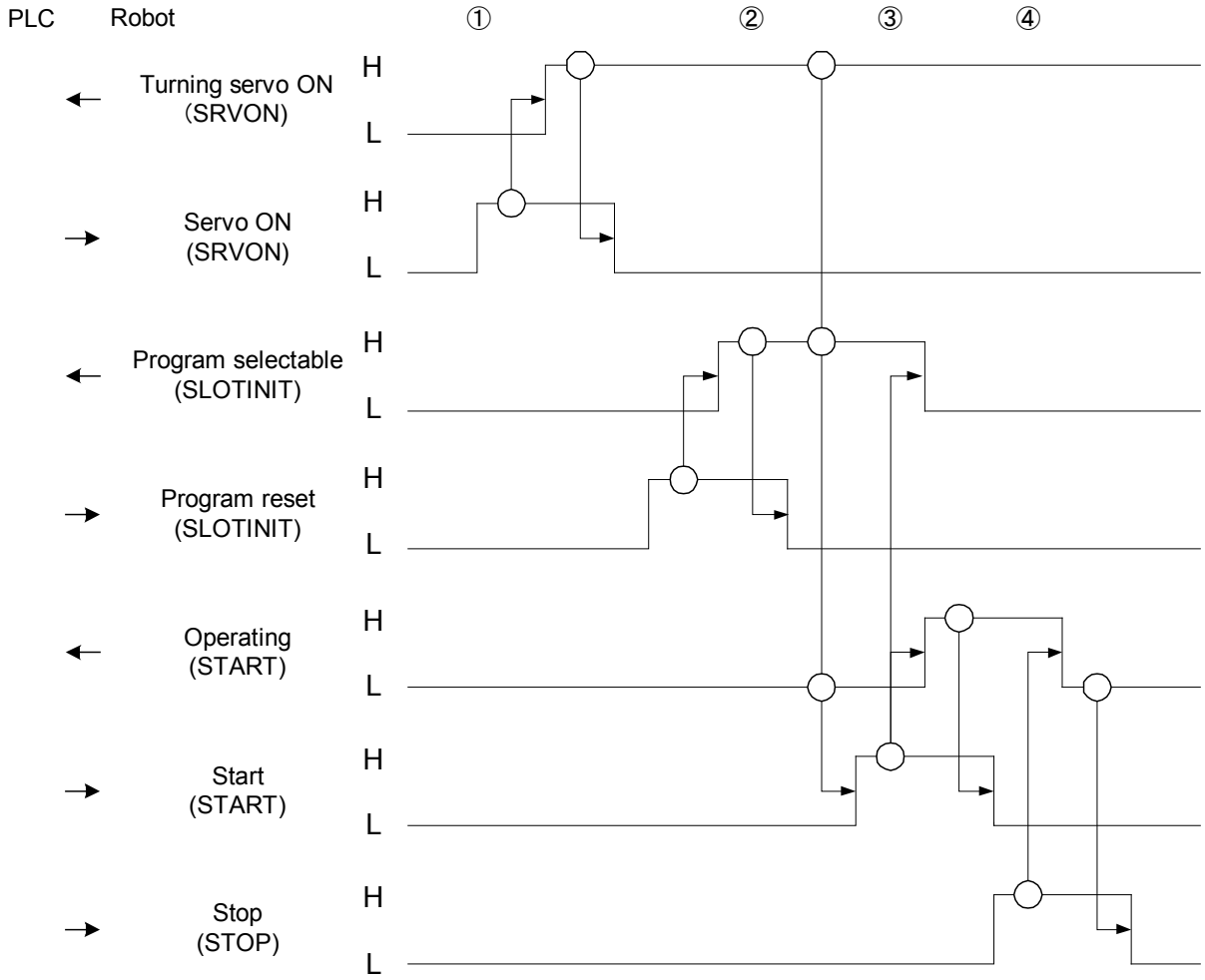
[Explanation]

(1) Indicates the status of a line connected with a network vision sensor with an NVOpen command when the line is opened.
(2) The initial value is "-1". At the point in time that the NVOpen command is executed and the line is connected, the value becomes "0" (line connecting). At the point in time that the network vision sensor logon is completed, the value becomes "1" (logon complete).
(3) This variable strongly resembles the status of status variable M_OPEN, but whereas M_Open becomes "1" when the connection is verified, M_NVOpen becomes "1" when the vision sensor logon is complete.
(4) If the type of data specified as an array element is incorrect, L4220 (syntax error in input command statement) error occurs.
(5) If there is an abnormal number of array elements (too many or too few), L3810 (incorrect argument type) error occurs.
(6) If an array element other than "1" through "8" is specified, L4370 (array element mistake) error occurs.

13. 2. Timing Diagram of Dedicated Input/Output Signals

13. 2. 1. Robot Program Start Processing

The signal timing when a robot program is started from an external device is shown below.



1) PLC sets “servo ON H” when it detects “turning servo ON L.” The robot turns the servo power supply on and sets “turning servo ON H.” PLC acknowledges “turning servo ON H” and sets “servo ON L.”

2) PLC sets “program reset H” upon receiving “program selectable L.” The robot returns to the beginning of the program and sets “program selectable H” when the program becomes ready to be started. PLC sets “program reset L” when it detects “program selectable H.”

3) PLC acknowledges “turning servo ON H,” “program selectable H” and “operating L” and sets “start H.” The robot sets “program selectable L” and “operating H” when it detects “start H.” PLC confirms “operating H” and sets “start L.”

4) If a stop signal is input, the following processing is performed.
Upon receiving “stop H” from PLC, the robot sets “operating L.”

14. Troubleshooting

This section explains causes of error occurrence and actions to be taken.

14.1. Occurrence of errors of Tracking and Vision Sensor

Table 14-1 List of Tracking relation Errors

Error number	Error description	Causes and actions
L2500	Tracking encoder data error	[Causes] The data of the tracking encoder is abnormal. (The amount of the change is 1.9 times or more.) [Actions] 1) Check the conveyor rotates at the fixed velocity. 2) Check the connection of the encoder. 3) Check the earth of the earth wire.
L2510	Tracking parameter reverses	[Causes] Tracking parameter[EXCRGMN] and [EXCRGMX] Setting value reverses [Actions] 1) Check the value of [ENCRGMX] and [ENCRGMN] parameters.
L2520	Tracking parameter is range over	[Causes] The set value is outside the range parameter [TRBUF]. The first argument is 1 to 8, and the second argument is 1 to 64. [Actions] 1) Check the value of [TRBUF] parameter.
L2530	There is no area where data is written	[Causes] The data of the size or more of the buffer in which the TrWrt command was continuously set to the second argument of parameter [TRBUF] was written. [Actions] 1) Check the execution count of the TrWrt command is correct. 2) Check the value of the second argument of parameter [TRBUF] is correct. 3) Check that the X and Y coordinates of the position variable ”PCHK” in “CM1” program are not ”0.” If they are ”0,” change the difference from the theoretical value to an allowable value.
L2540	There is no read data	[Causes] The TrRd command was executed in state the data is not written in tracking buffer. [Actions] 1) Execute the TrRd command after confirming whether the buffer has the data with the state variable [M_Trbfct]. 2) Confirm whether the buffer number specified by the buffer number specified in TrWrt command and the TrRd command is in agreement.
L2560	Illegal parameter of Tracking	[Causes] The value set as the parameter [EXTENC] is outside the range. The ranges are 1-8. [Actions] Please confirm the value set to Parameter [EXTENC]. Please confirm whether the Q173DPX unit is installed in the slot specified for parameter "ENCUNITn" (n=1-3). Please confirm whether slot 0-2 of a basic base is not specified by setting the parameter. Please confirm whether the setting of "Management CPU" that exists in "I/O unit and intelligent function unit details setting" of the parameter of the sequencer and specification of parameter "ENCUNITn" (n=1-3) are corresponding. There is a possibility Q173DPX is not robot CPU management.

Error number	Error description	Causes and actions
L2570	Installation slot error.	[Causes] Q173DPX is installed in slot 0-2 of a basic base. [Actions] Slot 0-2 of the basic base is basically only for CPU. Please install Q173DPX since slot3.
L2580	No workpiece in the tracking area.	[Causes] There is no workpiece in the tracking buffer or “TrkMv On” command is executed Before the workpiece enters to the tracking area. [Actions] Execute “TrkMv On” command when the workpiece is in the tracking area.
L3982	Cannot be used (singular point)	[Causes] 1) This robot does not correspond to the singular point function 2) Cmp command is executed 3) A synchronous addition axis control is effective 4) Tracking mode is effective 5) Pre-fetch execution is effective 6) This robot is a setting of the multi mechanism 7) ColChk On command is executed [Actions] 1) Check the argument of Type specification 2) Invalidate a compliance mode (execute Cmp Off) 3) Invalidate a synchronous addition axis control 4) Invalidate a tracking mode (execute Trk Off) 5) Invalidate a pre-fetch execution 6) Do not use the function of passage singular point 7) Invalidate a collision detection (execute ColChk Off)
L6632	Input TREN signal cannot be written	[Causes] During the actual signal input mode, external output signal 810 to 817 (TREN signal) cannot be written. [Actions] 1) Use an real input signal (TREN signal)

Table 14-2 List of Vision Sensor relation Errors

Error number	Error description	Causes and actions
L3130	COM file is already opened	[Causes] The communications line that was the subject of the attempted opening is already open. [Actions] Check the COM number and vision sensor number and re-execute. Or check the communications parameters.
L3141	The NVOpen command is not executed.	[Causes] No NVOpen command was executed before execution of a command communicating with the vision sensor. [Actions] Revise the robot program to execute the NVOpen command.
L3142	The communication line can not be opened.	[Causes] The line for communication with the vision sensor can not be opened. [Actions] Check the communication cable or the communications parameters.
L3501	Illegal Receive data(EBREAD)	[Causes] The type of the data received by EBRead command and the type of specified variable are different. [Actions] Revise the program.

Error number	Error description	Causes and actions
L7810	Abnormal Ethernet parameter setting	[Causes] The parameter setting is incorrect. [Actions] Check the NETHSTIP, NETPORT, NETMODE, and other such parameters.
L8600	Vision sensor not connected	[Causes] There is no vision sensor connected to the specified COM number. [Actions] Check the specified vision program number, "COMDEV" parameter, etc. settings.
L8601	Logon not possible	[Causes] The communication line was opened, but there is no response from the vision sensor. [Actions] Reset the program and start it again.
L8602	Wrong password	[Causes] The password for the user set with the "NVUSER" password is not set in the "NVPSWD" parameter. [Actions] Check the specified vision program number, "COMDEV" parameter, etc. settings.
L8603	Parameter abnormality	[Causes] There is no vision sensor connected to the specified COM number. [Actions] Check the NVUSER and NVPSWD parameters.
L8610	Abnormal communications	[Causes] Communication with the vision sensor was cut off before or during command execution. [Actions] Check the communication cable between the robot and vision sensor.
L8620	Abnormal vision sensor number specification	[Causes] The specified vision sensor number is not defined with an NVOpen command. [Actions] Check that the specified vision sensor number is correct. Also, check that that number is defined with an NVOpen command.
L8621	Abnormal vision program name	[Causes] The specified vision program name is more than 15 characters. [Actions] Specify a vision program name with no more than 15 characters.
L8622	Vision program not present.	[Causes] The specified program does not exist in the specified vision sensor. [Actions] Check whether the specified vision program exists in the specified vision sensor. Also check that the vision program name specified is correct.
L8623	SKIP number is already used.	[Causes] SKIP number is already used. [Actions] Confirm the SKIP number.
L8630	Incorrect value in recognition count cell	[Causes] The recognition count value was not in the cell specified as the recognition count cell. [Actions] Check that the correct cell is specified.

Error number	Error description	Causes and actions
L8631	Specified cell value out of range	[Causes] Corresponding to either the following. ・The values specified for the start cell and end cell are reversed. ・The range specified by Start Cell and End Cell exceeds line 30 and row 10. ・The number of data included in the cell which specifies it by Start Cell and End Cell exceeds 90. [Actions] ・Check that the correct cell is specified. ・Check the number of data acquired from the cell which specifies it by Start Cell and End Cell.
L8632	Vision sensor response timeout	[Causes] There is no response from vision sensor within the specified time or within a specific time. [Actions] Check that the specified time is correct. Or check that the vision sensor settings are correct.
L8636	Vision Tag name is abnormal	[Causes] The active specified symbolic tag does not exist in the vision program. [Actions] Please confirm the name of symbolic tag of Easy Builder is corresponding to the tag name specified by the robot program, and correct the tag name.
L8640	Abnormal image capture specification	[Causes] The image capture specification is other than "Network", "external", and "manual". [Actions] Specify an image capture specification of "Network", "external", or "manual".
L8650	Put online.	[Causes] The vision sensor is offline. [Actions] Put the vision sensor online to enable control from the outside.
L8660	Not permitted to control vision sensor	[Causes] The NVUSER and NVPSWD parameters set for logging on to the vision sensor do not have the right to full access to the vision sensor. [Actions] Check the vision sensor side user list registration and specify the name of a user with full access in NVUSER and their password in NVPSWD.

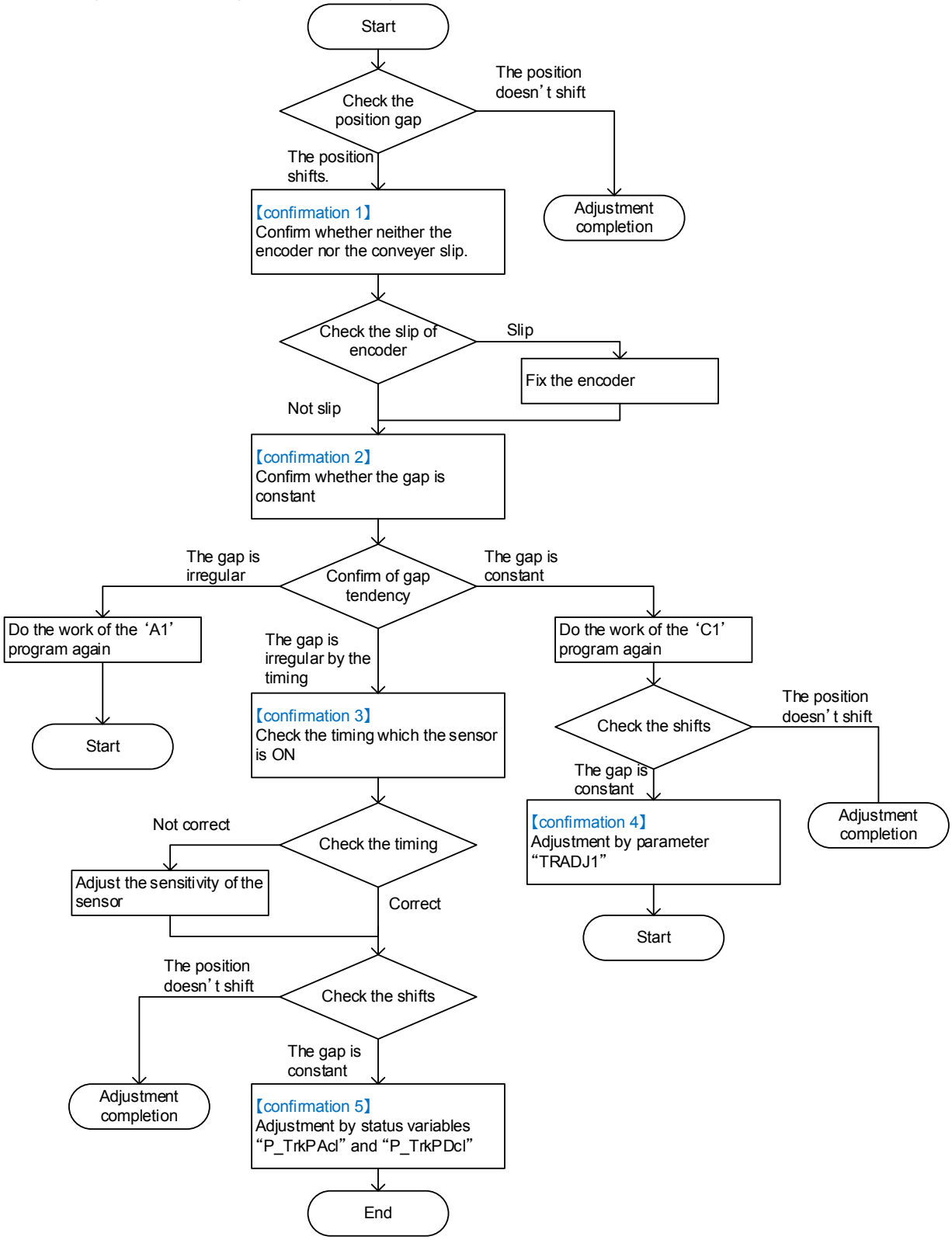
Please refer to separate manual “Troubleshooting”.

14. 2. In such a case (improvement example)

Explain the improvement example, when building the tracking system using the sample robot program.

14. 2. 1. The adsorption position shifts (Conveyer Tracking)

When the place that shifts from the specified adsorption position has been adsorbed, the cause is investigated according to the following procedures.



[Confirmation 1]
1) Stop the conveyer.
2) Confirm the disk installed in the rotary encoder has come in contact with the conveyer.
3) Confirm whether the disk installed in the encoder rotates when the conveyer is made to work.

[Confirmation 2]
1) Stop the conveyer.
2) Change X coordinates of PDly1 in '1' program to a big value like the "10" second etc.
3) Start '1' program, and start the conveyer in low-speed.
4) Stop the conveyer because it keeps following during the "10" second in the place where the robot moved to the adsorption position. And, stop '1' program.
5) Confirm whether the position in which the robot adsorbs workpiece is correct.
6) Confirm the tendency to a positional gap repeating this work several times.

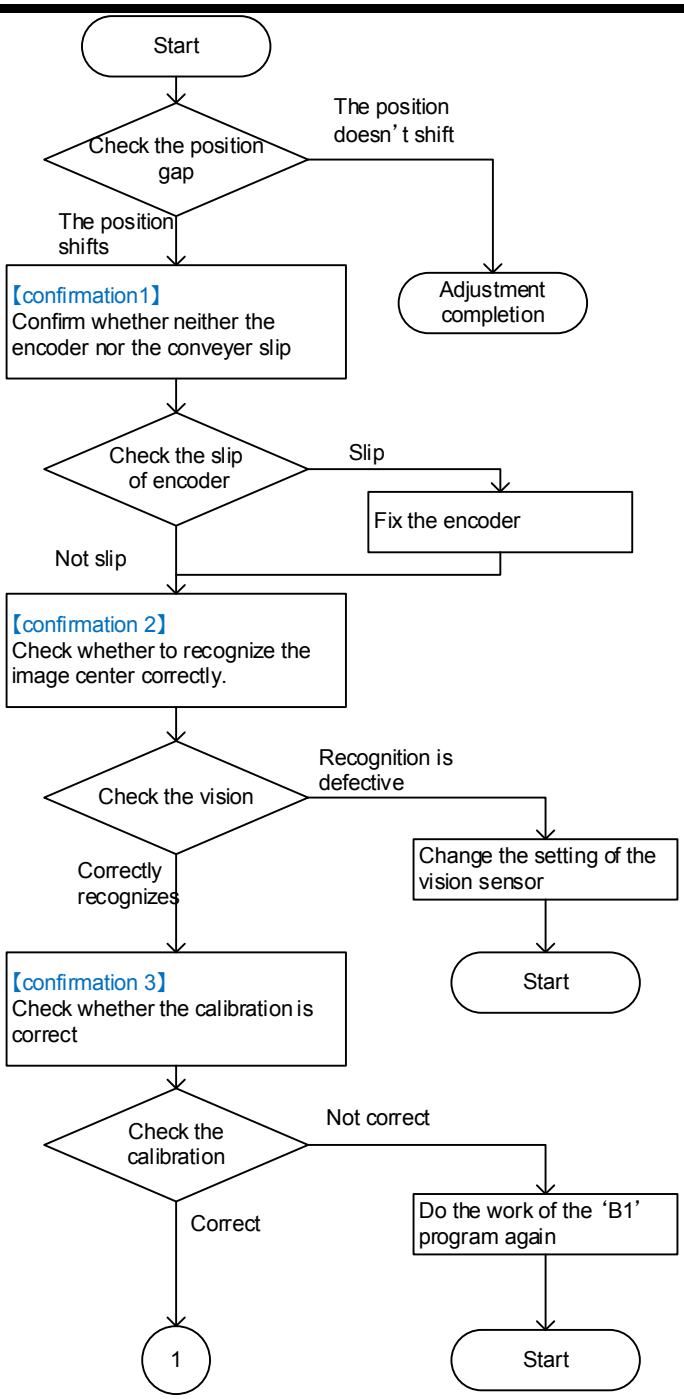
[Confirmation 3]
1) Stop the conveyer.
2) Put a workpiece at the front to which a sensor reacts.
3) Move the conveyer manually and confirm whether the timing to which a sensor reacts is correct.

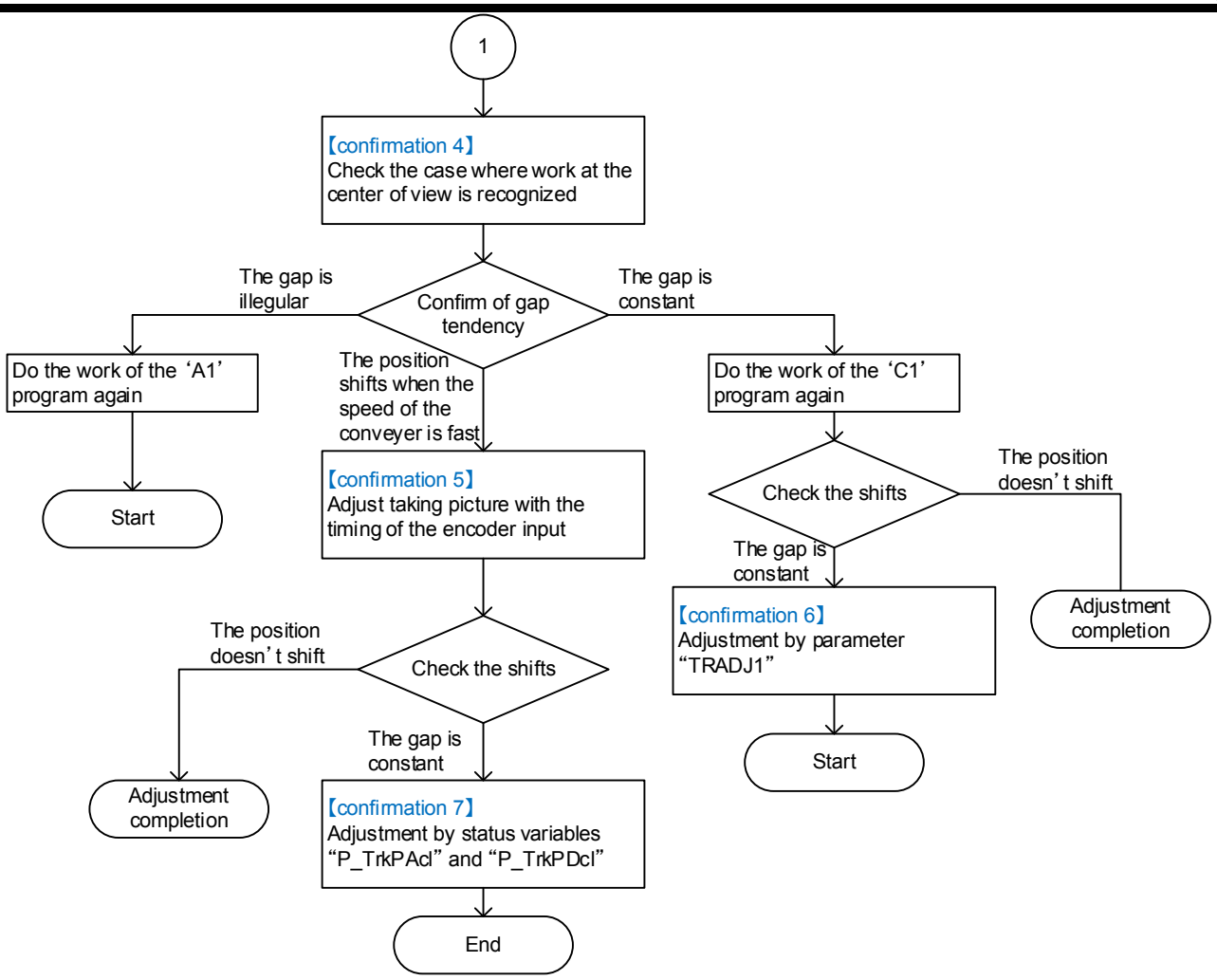
[Confirmation 4]
1) Change parameter "TRADJ1", and adjust a positional gap.

[Confirmation 5]
1) Change robot status variable "P_TrkPAcl" and "P_TrkPDcl" to make the follow speed of the tracking fast.
   Note it though the load factor of each axis of the robot goes up.
   Confirm the state of the load of each axis by "Load factor monitor" of RT ToolBox2.

14. 2. 2. The adsorption position shifts (Vision Tracking)

When the place that shifts from the specified adsorption position has been adsorbed, the cause is investigated according to the following procedures.


Start
Check the position gap
The position doesn't shift
The position shifts
Adjustment completion
【confirmation1】
Confirm whether neither the encoder nor the conveyer slip
Check the slip of encoder
Slip
Fix the encoder
Not slip
【confirmation 2】
Check whether to recognize the image center correctly.
Check the vision
Recognition is defective
Change the setting of the vision sensor
Start
Correctly recognizes
【confirmation 3】
Check whether the calibration is correct
Check the calibration
Not correct
Do the work of the 'B1' program again
Start
Correct
1



[Confirmation 1]
1) Stop the conveyer.
2) Confirm the disk installed in the rotary encoder has come in contact with the conveyer.
3) Confirm whether the disk installed in the encoder rotates when the conveyer is made to work.

[Confirmation 2]
1) Stop the conveyer.
2) Put workpiece on the center of the vision view.
3) In In-Sight Explorer(EasyBuilder), click the "Set Up Image" from the "Application Steps". And, set "Calibration Type" displayed in the lower right of the screen to "None".
4) Confirm workpiece is recognized by starting the job, and the recognition result (pixel level) is correct.
   (example)
   When the center of view is recognized, the result of (320,240) is displayed when pixels are 640×480 vision sensors.
5) Arrange workpieces on four corners.
6) Confirm whether the workpieces put on four corners of the image is recognized similar and correctly.

[Confirmation 3]
1) Stop the conveyer.
2) Put workpiece on the center of the vision view.
3) In In-Sight Explorer(EasyBuilder), click the "Set Up Image" from the "Application Steps".
   Set "Calibration Type" displayed in the lower right of the screen to "Import".
   Specify the file that exported when the calibration is done to "File Name".
4) Confirm workpiece is recognized by starting the job, and the recognition result (robot coordinate) is correct.

(example)
(+0, +0) is displayed as a recognition result when assuming that the robot coordinates are set as follows when the calibration is done by using the calibration seat, and using a ○ sign in four corners.
(the first point xy) (the second point xy)(the third point xy)(the fourth point xy)
= (+100,+100), (+100,-100), (-100,+100), and (-100,-100)

5) Arrange workpieces on four corners.
6) Confirm whether the workpieces put on four corners of the image is recognized similar and correctly.
The recognition result becomes (+100,+100), (+100,-100), (-100,+100), and (-100,-100).

[Confirmation 4]
1) Stop the conveyer.
2) Put workpiece on the center of the vision view.
3) Change X coordinates of PDly1 in '1' program to a big value like the "10" second etc.
4) Start '1' program, and start the conveyer in low-speed.
5) Stop the conveyer because it keeps following during the "10" second in the place where the robot moved to the adsorption position. And, stop '1' program.
6) Confirm whether the position in which the robot adsorbs workpiece is correct.
7) Confirm the tendency to a positional gap repeating this work several times.

[Confirmation 5]
1) Stop the conveyer.
2) Start the '1' program, and start the conveyer in the speed that you want.
3) Flow workpiece.
4) Stop the conveyer because it keeps following during the "10" second in the place where the robot moved to the adsorption position. And, stop '1' program.
5) Confirm the position in which the robot adsorbs workpiece.

<The position shifts in shape to adsorb the rear side of work >
Please adjust the encoder value specified by the TrWrt command as < delay time > "0".
For instance, the 'CM1' program is changed as follows and the numerical value (for instance, following "500") is adjusted.

```
MENCDATA#=MTR1#+500
TrWrt PRW, MENCDATA#, MWKNO,1,MENCNO
```

[Confirmation 6]
1) Change parameter "TRADJ1", and adjust a positional gap.

[Confirmation 7]
1) Change robot status variable "P_TrkPAcl" and "P_TrkPDcl" to make the follow speed of the tracking fast.
Note it though the load factor of each axis of the robot goes up.
Confirm the state of the load of each axis by "Load factor monitor" of RT ToolBox2.

14.2.3. Make adsorption and release of the work speedy

Adjust the adjustment variable "PDly1", and the value of X coordinates of "PDly2" of the program 1. Refer to "**Table** 11-1 List of adjustment variables in the program" for the adjustment method.

14.2.4. Make movement of the robot speedy

Adjust the following setting to make movement of the robot speedy.

1) Adjustment of the optimal acceleration-and-deceleration setting
Set mass, size, and center of gravity of the hand installed in the robot as the parameter "HNDDAT1."
And, set mass, size, and center of gravity of the work as the parameter "WRKDAT1."

By this setting, the robot can move with the optimal acceleration and deceleration and speed.
Refer to "Table 6-2 List of Operation Parameter" for setting method.

2) Adjustment of carrying height
By making low distance at adsorption and release of robot, the moving distance decreases and motion time can be shortened as a result. Refer to the adjustment variable of "Pup1"and "Pup2" in the "**Table** 11-1 List of adjustment variables in the program" for change of rise distance.

14. 2. 5. Restore backup data to another controller

The status variable "P_EncDlt" is not saved in the backup data from tracking system robot controller. To generate the value of "P_EncDlt", execute the "P_EncDlt(MENCNO) =PY10ENC" command of "Program A" by step forward. (Moving distance per one pulse)

14. 2. 6. Circular arc movement in Tracking

Screw fastening and decoration on the work, etc are available in the tracking system. Here, explain the example which draws the circle on the basis of the adsorption position.

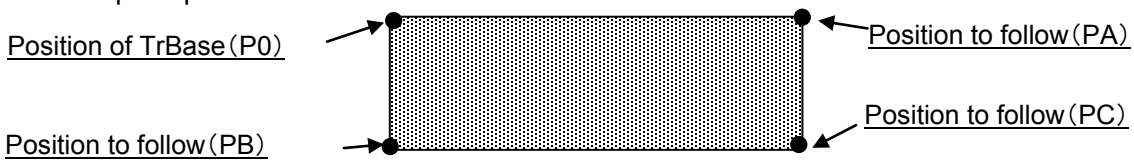
<Conditions>
*The adsorption position taught by Program C is the starting point of the circle.
*The offset from the adsorption position of pass and end position of circle decided as follows.
*Create PS1 (pass point) and PS2 (end point) from the relative distance.
*Use the Mvr command (circle command) and move on the circle of PGet->PS1 ->PS2.

The example of program change of the above <conditions> is shown in the following.

Before sample program change	After sample program change
TrkMv On, PGetUp, 1, *S91STOP Mov PGet Type 0,0 Dly PDly1.x Mov PGetUp Type 0,0 TrkMv Off ・・・・・ P_TrkBase(MWrkNo) = P_107(MWrkNo) PGet = P_TrkBase(MWrkNo)	TrkMv On, PGetUp, 1, *S91STOP Mov PGet Type 0,0 Dly PDly1.x **'<Add>->** **Mvr PGet,PS1,PS2** **Mvs PGet** **Dly PDly1.X** **'<-<Add>** Mov PGetUp Type 0,0 TrkMv Off ・・・・・ P_TrkBase(MWrkNo) = P_107(MWrkNo) PGet = P_TrkBase(MWrkNo) **'<Add>->** **PS1 = PGet * (+5.00,+5.00,+0.00,+0.00,+0.00,+0.00,+0.00)** **PS2 = PGet * (+0.00,+10.00,+0.00,+0.00,+0.00,+0.00,+0.00)** **'<-<Add>**

14. 2. 7. Draw the square while doing the Tracking

Here, explain the example which draws the outline of the following square workpiece on the basis of the adsorption position.



Before sample program change	After sample program change
TrkMv On, PGetUp, 1, *S91STOP Mov PGet Type 0,0 Dly PDly1.x Mov PGetUp Type 0,0 TrkMv Off ・・・・・ P_TrkBase(MWrkNo) = P_107(MWrkNo) PGet = P_TrkBase(MWrkNo)	TrkMv On, PGetUp, 1, *S91STOP Mov PGet Type 0,0 Dly PDly1.x **'<Add>->** **Mvs PA** **Mvs PC** **Mvs PB** **Mvs PGet** **Dly PDly1.X 'Adsorption confirmation** **'<-<Add>** Mov PGetUp Type 0,0 TrkMv Off ・・・・・ P_TrkBase(MWrkNo) = P_107(MWrkNo) PGet = P_TrkBase(MWrkNo) **'<Add>->** **PA = PGet * (+0.00,-50.00,+0.00,+0.00,+0.00,+0.00,+0.00)** **PC = PGet * (-20.00,-50.00,+0.00,+0.00,+0.00,+0.00,+0.00)** **PB = PGet * (-20.00,+0.00,+0.00,+0.00,+0.00,+0.00,+0.00)** **'<-<Add>**

15. Appendix

This appendix provides a list of parameters related to tracking and describes Expansion serial interface connector pin assignment as well as sample programs for conveyer tracking and vision tracking.

15.1. List of Parameters Related to Tracking

Table 15-1 List of Parameters Related to Tracking

Parameter	Parameter name	Number of elements	Description	Setting value at factory shipment
Tracking buffer	TRBUF	2 integers	Number of tracking buffers and their sizes (KB) <Buffer number> Specify the number of buffers where the tracking data is stored. Mainly the tracking data for each conveyors is saved at the buffer. Change the set value, when the conveyor for tracking is increased. However, if the value is enlarged, the memory area where the tracking data is saved will be secured. Be careful because the program number which can be saved decreases. Setting range: 1 to 8 <Buffer size> Specify the size in which the tracking data is preserved. Change this element when there is larger tracking data saved by TrWrt command than reading by TrRd command. Be careful because the memory is secured like the above-mentioned [Buffer number]. Setting range: 1 to 200	2 , 64
Minimum external encoder value	ENCRGMN	8 integers	The minimum external encoder data value (pulse) The range of the encoder value which can be acquired in state variable “M_Enc" (minimum value side)	0,0,0,0,0,0,0,0
Maximum external encoder value	ENCRGMX	8 integers	The maximum external encoder data value (pulse) The range of the encoder value which can be acquired in state variable “M_Enc" (maximum value side)	1000000000, 1000000000, 1000000000, 1000000000, 1000000000, 1000000000, 1000000000, 1000000000

Parameter	Parameter name	Number of elements	Description	Setting value at factory shipment
Tracking adjustment coefficient 1	TRADJ1	8 real numbers (X,Y,Z, A,B,C, L1,L2)	Tracking adjustment coefficient 1 Set the amount of delay converted to the conveyer speed. Convert to 100 mm/s. Example) • If the delay is 2 mm when the conveyer speed is 50 mm/s: Setting value = 4.0 (2 / 50 * 100) • If the advance is 1 mm when the conveyer speed is 50 mm/s: Setting value = -2.0 (-1 / 50 * 100)	0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00
Tracking acceleration	TRPACL	8 real numbers (X,Y,Z, A,B,C, L1,L2)	Tracking acceleration. Acceleration during execution of tracking movement.	1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0
Tracking deceleration	TRPDCL	8 real numbers (X,Y,Z, A,B,C, L1,L2)	Tracking deceleration. Deceleration during execution of tracking movement.	1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0

15.2. List of Parameters Related to Vision Sensor

Table 15-2 List of Parameters Related to Vision Sensor

Parameter	Parameter name	Number of elements	Description	Setting value at factory shipment
User name	NVUSER	Character string 1	The user name to log on the vision sensor is set. (no more than 15 characters)	"admin"
Password	NVPSWD	Character string 1	The password to log on the vision sensor is set.(no more than 15 characters)	""
Network Vision Job load timeout [sec]	NVJBTOUT	integer 1	Set up a timeout time of network vision sensor used with NVLoad command and TrkTrg command.	90
Initial value of tag name specified by EBRead command	EBRDTAG	Character string 1	Sets up an initial value of the "symbolic tag name" used with EBRead command (it is 128 characters or less) When the tag name of EBRead command is omitted, the value of this parameter is specified.	"Job.Robot. FormatString"
TimeOut for OPEN Command(sec)	OPNTOUT	real number 1	Set up a timeout time used with NVOpen command.	3.00

15.3. Scene of changing parameter

When the tracking function is used, the parameter need to be changed depends on operation phase. List of the parameter is shown as follow.

Table **15-3 List of the user scene of changing parameter**

No.	Operation phase	Model		Parameter name	Example	Explanation
		CR750-Q CR751-Q	CR750-D CR751-D			
1	Power on Setting origin JOG operation	—	—	—	—	
2	Attach option Connection with peripherals	●	—	ENCUNIT1 ENCUNIT2 ENCUNIT3	0, 5 -1, 0 -1, 0	It is set to have installed Q173DPX unit into 5 I/O slot of the base unit. By setting it, incremental three encoders connected with Q173DPX unit are recognized physical encoder number 1 to 3.
3		●	●	TRMODE	1	It makes tracking function valid. By being valid, incremental encoder value can be got.
4	In case of robot programming	●	●	EXTENC	1, 2, 3, 1, 2, 3, 1, 2	About EXTENC, because initial value is 1,2,1,2,1,2,1,2, physical encoder number 1 and 2 are allocated to logic encoder(physical encoder number3) number 1 to 8. At this time, the encoder connected with CH3 of Q173DPX unit is not allocated to logic encoder number. So by changing this parameter to 1,2,3,1,2,3,1,2, the encoder of CH3 is allocated to logic encoder number 3 and 6. Also it is possible in following case. 3 pcs encoder are connected with Q173DPX unit and attach each encoder to conveyer 1 to 3. If conveyer1 connect to encoder3, conveyer 3 connect to encoder 1, it is not effective to change encoder, so by changing this parameter to 3,2,1,3,2,1,1,2, encoder attached with conveyer 1 becomes logic encoder1.

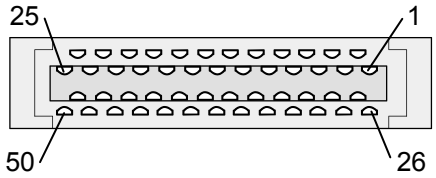
No.	Operation phase	Model		Parameter name	Example	Explanation
		CR750-Q CR751-Q	CR750-D CR751-D			
5	In case of system debag	•	•	TRCWDST	20.0	In case of vision tracking, if there is a workpiece not recognized well by vision sensor, it might reply over one recognition results to one workpiece. In this case, it makes possible to get only one recognition result excluding the results with the distance which is shorter than the distance set by this parameter. For example, it is recognized that 3 vision sensors exist for 1 workpieces. This one workpiece is got and another 2 workpieces are not got because the distance of result is shorter than it set 20mm.
6	In case of system debug	•	•	TRADJ1	+0.00, +4.00, +0.00, +0.00, +0.00, +0.00, +0.00, +0.00, +0.00	It is possible to adjust the gap by using this parameter when this gap is caused every time in the same direction when the tracking operates. For example, the speed of conveyer is 50mm/s and there is +2mm gap (＋Y direction) +2mm, Set value = 4.0 (2 / 50 * 100) +4.0 is set to the second element that shows Y coordinates.
7		•	•	TRBUF	3, 100	When three kinds of workpieces flow respectively on the three conveyers for one robot controller, three tracking buffers where workpiece information is preserved are needed. In this case, the first element of this parameter is changed to three. Moreover, when TrWrt command is frequently executed and TrRd command is slow, workpiece information collects in the tracking buffer. Because the error occurs when 64 workpieces information or more on an initial value collects, it is necessary to increase the number in which work information is preserved. Then, the second element of this parameter is changed to 100.

No.	Operation phase	Model		Parameter name	Example	Explanation
		CR750-Q CR751-Q	CR750-D CR751-D			
8	Others	●	●	ENCRGMN	0,0,0,0, 0,0,0,0	This parameter is a parameter that sets the range of the value of state variable M_Enc. M_Enc becomes the range of 0-100000000, and next to 100000000, it becomes 0 encoder rotates in case of an initial value. Though this range is changed by this parameter, tracking sample program is made on the assumption that it is used within this range, so do not change this parameter.
9		●	●	ENCRGMX	100000000 , 100000000 , 100000000 , 100000000 , 100000000 , 100000000 , 100000000 , 100000000	

15.4. Expansion serial interface Connector Pin Assignment

"Figure15-1 Connector Arrangement" shows the connector arrangement and "Table 15-4 Connectors: CNENC/CNUSR Pin Assignment" shows pin assignment of each connector.

CNUSR2(CR750-D/CR751-D)
Encoder

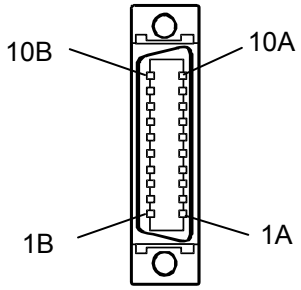


Connector: CNUSR2

CNUSR11/12/13(CR750-D)
Encoder

Connector: CNUSR11/12/13

CNENC(CRnD-700)
Encoder × 2CH



Connector: CNENC

Figure15-1 Connector Arrangement

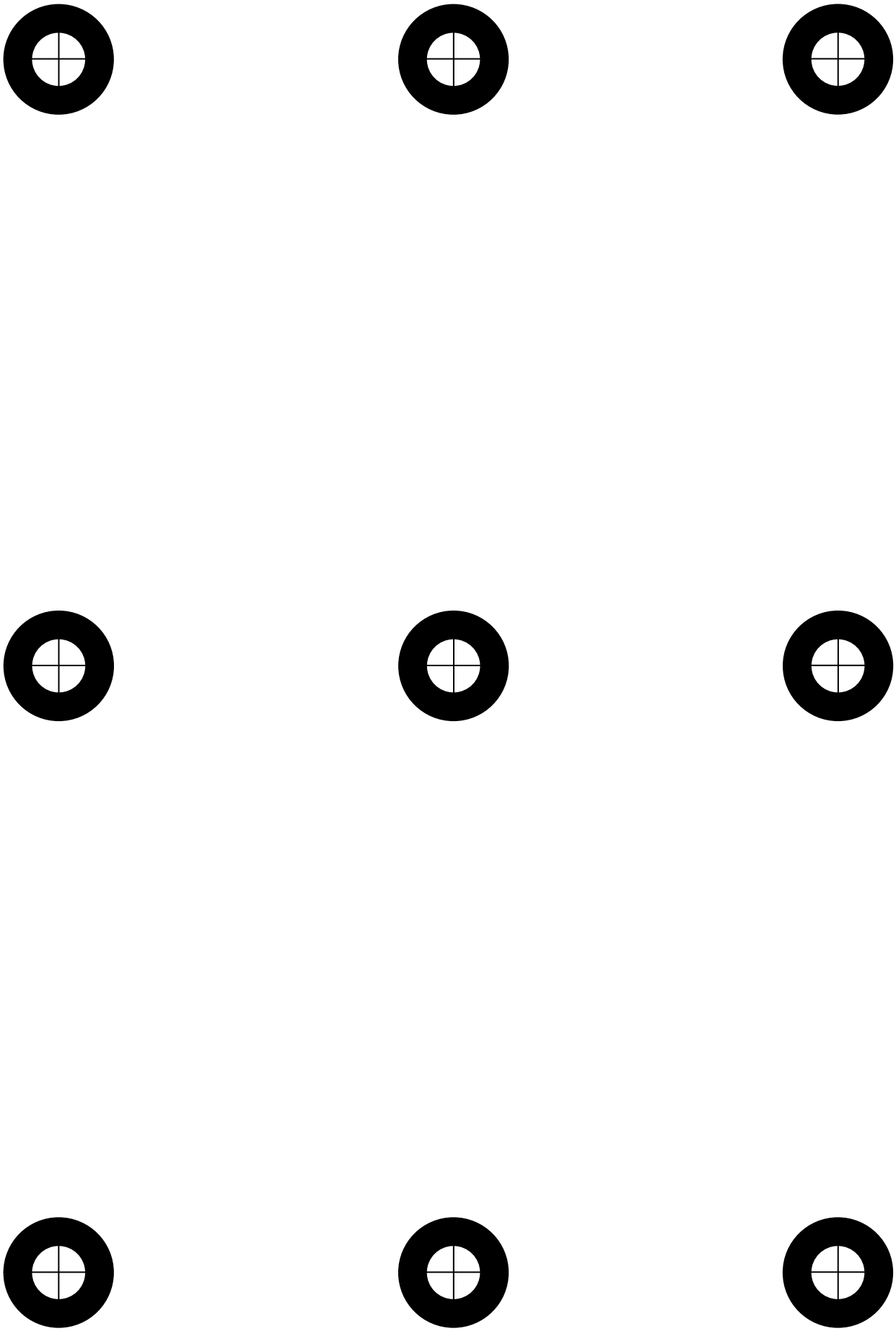
Table 15-4 Connectors: CNENC/CNUSR Pin Assignment

Pin NO. Connector name – Pin name		Signal name	Explanation	Input/output	Remark
CR751-D controller	CR751-D controller				
CNUSR1-28	CNUSR**11**-6	SG	Control power supply 0 V	GND	CH1
CNUSR1-21	CNUSR13-3	LAH1	+ terminal of differential encoder A-phase signal	Input	
CNUSR1-22	CNUSR13-5	LBH1	+ terminal of differential encoder B-phase signal	Input	
CNUSR1-23	CNUSR13-8	LZH1	+ terminal of differential encoder Z-phase signal	Input	
CNUSR1-33	CNUSR**12**-6	SG	Control power supply 0 V	GND	CH2
CNUSR2-21	CNUSR2-21	LAH2	+ terminal of differential encoder A-phase signal	Input	
CNUSR2-22	CNUSR2-22	LBH2	+ terminal of differential encoder B-phase signal	Input	
CNUSR2-23	CNUSR2-23	LZH2	+ terminal of differential encoder Z-phase signal	Input	
-	-	-	Empty	–	
-	-	-	Empty	–	
CNUSR2-15	CNUSR2-15	SG	Control power supply 0 V	GND	CH1
CNUSR1-46	CNUSR13-4	LAL1	- terminal of differential encoder A-phase signal	Input	
CNUSR1-47	CNUSR13-6	LBL1	- terminal of differential encoder B-phase signal	Input	
CNUSR1-48	CNUSR13-10	LZL1	- terminal of differential encoder Z-phase signal	Input	
CNUSR2-40	CNUSR2-40	SG	Control power supply 0 V	GND	CH2
CNUSR2-46	CNUSR2-46	LAL2	- terminal of differential encoder A-phase signal	Input	

Pin NO. Connector name – Pin name		Signal name	Explanation	Input/output	Remark
CR751-D controller	CR751-D controller				
CNUSR2-47	CNUSR2-47	LBL2	- terminal of differential encoder B-phase signal	Input	
CNUSR2-48	CNUSR2-48	LZL2	- terminal of differential encoder Z-phase signal	Input	
-	-	-	Empty	–	
-	-	-	Empty	–	

15.5. Calibration sheet

This is a calibration sheet. Please use this sheet in your calibration work.
Enlarge or reduce it as necessary to match the size of the field of vision of the image.
When changing the size of the sheet, or calibrating in more points, you can photocopy the sheet.



MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE: TOKYO BUILDING, 2-7-3, MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS: 5-1-14, YADA-MINAMI, HIGASHI-KU NAGOYA 461-8670, JAPAN

Authorised representative:
MITSUBISHI ELECTRIC EUROPE B.V. GERMANY
Gothaer Str. 8, 40880 Ratingen / P.O. Box 1548, 40835 Ratingen, Germany

Mar., 2014 MEE Printed in Japan on recycled paper.