# FANUC Robot series

**R-30*i*B/R-30*i*B Mate/R-30*i*B Plus/R-30*i*B Mate Plus/**
**R-30*i*B Compact Plus/R-30*i*B Mini Plus CONTROLLER**

**Line Tracking**

# OPERATOR'S MANUAL

**B-83474EN/03**

## • Original Instructions

Thank you very much for purchasing FANUC Robot.

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



The products in this manual are controlled based on Japan's “Foreign Exchange and Foreign Trade Law”. The export from Japan may be subject to an export license by the government of Japan.
Further, re-export to another country may be subject to the license of the government of the country from where the product is re-exported. Furthermore, the product may also be controlled by re-export regulations of the United States government.
Should you wish to export or re-export these products, please contact FANUC for advice.

In this manual, we endeavor to include all pertinent matters. There are, however, a very large number of operations that must not or cannot be performed, and if the manual contained them all, it would be enormous in volume. It is, therefore, requested to assume that any operations that are not explicitly described as being possible are "not possible".

# SAFETY PRECAUTIONS

This chapter describes the precautions which must be followed to enable the safe use of the robot. Before using the robot, be sure to read this chapter thoroughly.

For detailed functions of the robot operation, read the relevant operator's manual to understand fully its specification.

For the safety of the operator and the system, follow all safety precautions when operating a robot and its peripheral equipment installed in a work cell.
For safe use of FANUC robots, you must read and follow the instructions in the “FANUC Robot series SAFETY HANDBOOK (**B-80687EN)**”.

# 1 PERSONNEL

Personnel can be classified as follows.

Operator:
- Turns the robot controller power ON/OFF
- Starts the robot program from operator panel

Programmer or Teaching operator:
- Operates the robot
- Teaches the robot inside the safeguarded space

Maintenance technician:
- Operates the robot
- Teaches the robot inside the safeguarded space
- Performs maintenance (repair, adjustment, replacement)

- The operator is not allowed to work in the safeguarded space.
- The programmer or teaching operator and maintenance technician are allowed to work in the safeguarded space. Work carried out in the safeguarded space include transportation, installation, teaching, adjustment, and maintenance.
- To work inside the safeguarded space, the person must be trained on proper robot operation.

Table 1 (a) lists the work outside the safeguarded space. In this table, the symbol “O” means the work allowed to be carried out by the specified personnel.

**Table 1 (a) List of work outside the Safeguarded Space**

| | Operator | Programmer or Teaching operator | Maintenance technician |
|---|---|---|---|
| Turn power ON/OFF to Robot controller | O | O | O |
| Select operating mode (AUTO/T1/T2) | | O | O |
| Select remote/local mode | | O | O |
| Select robot program with teach pendant | | O | O |
| Select robot program with external device | | O | O |
| Start robot program with operator’s panel | O | O | O |
| Start robot program with teach pendant | | O | O |
| Reset alarm with operator’s panel | | O | O |
| Reset alarm with teach pendant | | O | O |
| Set data on teach pendant | | O | O |
| Teaching with teach pendant | | O | O |
| Emergency stop with operator’s panel | O | O | O |
| Emergency stop with teach pendant | O | O | O |
| Operator’s panel maintenance | | | O |
| Teach pendant maintenance | | | O |

During robot operation, programming and maintenance, the operator, programmer, teaching operator and maintenance technician take care of their safety using at least the following safety protectors:

- Use clothes, uniform, overall adequate for the work
- Safety shoes
- Helmet

# 2 DEFINITION OF SAFETY NOTATIONS

To ensure the safety of users and prevent damage to the machine, this manual indicates each precaution on safety with "**WARNING**" or "**CAUTION**" according to its severity. Supplementary information is indicated by "**NOTE**". Read the contents of each **"WARNING**", "**CAUTION**" and "**NOTE**" before using the robot.

| Symbol | Definitions |
|---|---|
| ⚠WARNING | Used if hazard resulting in the death or serious injury of the user will be expected to occur if he or she fails to follow the approved procedure. |
| ⚠CAUTION | Used if a hazard resulting in the minor or moderate injury of the user, or equipment damage may be expected to occur if he or she fails to follow the approved procedure. |
| NOTE | Used if a supplementary explanation not related to any of WARNING and CAUTION is to be indicated. |

# TABLE OF CONTENTS

## APPENDIX

# 1 OVERVIEW

## 1.1 OVERVIEW

Tracking is an optional feature that enables a robot to treat a moving workpiece as a stationary object. The option is used in conveyor applications, where the robot must perform tasks on moving workpieces without stopping the assembly line. See Figure 1.1.
This user guide provides information for the installation and operation of Line Tracking option. This feature provides a complete stand-alone environment for all teach pendant-based line tracking, with teach pendant SETUP screen access to tracking parameters and teach pendant instructions for tracking program execution.

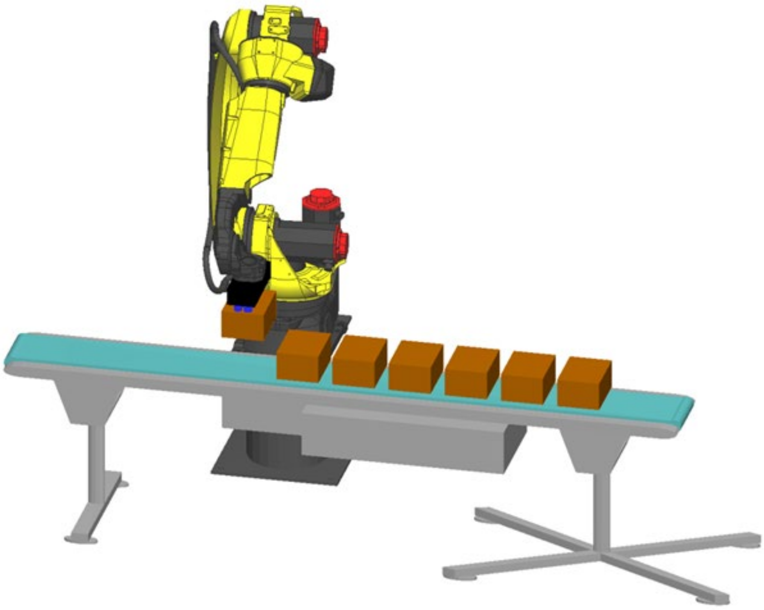


**Fig. 1.1 Typical tracking workcell**

## 1.2 GENERAL TRACKING DESCRIPTIONS

Tracking refers to the option for tracking an assembly line or workpieces moving on a conveyor. In this environment, the robot must track and manipulate a workpiece which is moving through its workspace on a conveyor, platform, or other mechanism.
Tracking saves production time by allowing workpieces to continue to move on the conveyor. Tracking can also increase the working volume of the robot's workspace if you carefully segment the program into various regions or windows. Each region lies within the robot's workspace, at some time, as the workpiece moves past the robot.
Tracking can be accomplished in two ways:
- Single axis line tracking
- Cartesian line or circular tracking

## 1.3 SINGLE-AXIS (RAIL) TRACKING

In single axis tracking, the position of the robot's extended axis (an integrated or non-integrated base axis) is adjusted to track the motion of a linear conveyor. The conveyor motion direction must be parallel to that of the tracking axis.

This single-axis tracking is known as rail tracking, since the typical application uses a rail or platform to perform the tracking motion. With rail tracking, the robot arm configuration (excluding the tracking axis) remains as programmed. All types of motion (Linear, Circular, and Joint) are allowed.
Rail tracking is a simple method of dealing with a constantly moving workpiece. Rail tracking is used in large systems that can occupy a large amount of floor space. It is easy to teach and works with almost any application. This option allows a large volume of work to be accomplished by one system.

# 1.4 CARTESIAN TRACKING

Cartesian tracking refers to a stationary robot whose Tool Center Point (TCP) position is adjusted to track the motion of a conveyor. You should use Cartesian tracking whenever floor space is a primary concern, or if you cannot install a rail axis for tracking. The robot can be mounted above or below the conveyor, or on a rail or other integrated extended axis depending on the needs of the application.
There are two kinds of Cartesian tracking: Line and Circular (not to be confused with Linear and Circular motions).

**NOTE**
With Cartesian tracking, the arm configuration of the main robot axes (not including any extended axes which might be present) is changed to achieve the tracking motion. Because of this, Cartesian tracking is restricted to Linear and Circular program motions. Joint motions are not supported.

**NOTE**
Cartesian tracking with extended axes motion group only supports integrated extended axes.

## 1.4.1 LINE Tracking

Cartesian line tracking consists of a robot and a linear conveyor which moves parts past a robot. The robot is usually mounted on a stationary pedestal beside the conveyor, where it can easily reach the parts as they move past it.

## 1.4.2 CIRCULAR Tracking

Cartesian circular tracking consists of a circular conveyor or rotary table which moves parts past a robot. The robot can be located either inside or outside the circle of the conveyor.

**NOTE**
Tracking boundaries when using circular tracking can only be used when bit4(bit 0x10) of $LNCFG.$COMP_SW2 is TRUE. If the software series is 7DF1(V9.10P) or later, this setting is TRUE by default.

**NOTE**
When using the WJNT instruction with circular tracking, if the conveyor speed is not constant or if the override is less than 100%, the robot TCP angle at the time of reaching the target position may differ from the teaching position. If you uses the WAIT command after WJNT motion, the robot TCP will track while maintaining the angle in the world frame.
In particular, 6-axis parallel link robots such as M-3iA/6A, M-2iA/3A, M-2iA/3A always make WJNT motion to protect the wrist mechanism, so this phenomenon is likely to occur.

# 2 HARDWARE AND SOFTWARE

## 2.1 HARDWARE

The line tracking system requires a line tracking interface board (option) within the controller, or uses the interface on main CPU board to connect an encoder(Pulsecoder). When a line tracking interface board is used, a fiber optic FSSB connection cable is required too.
Additionally, external hardware (a tracking encoder) and the associated interconnections (an encoder cable) are required to track the line (conveyor, platform, table, and so forth). See Fig. 2.1.1(a) - 2.1.1 (f) in using αA1000S Pulsecoder, A860-0372-T001 (which is adaptable to both incremental and absolute.). See Appendix C in using incremental Pulsecoder, A860-0301-T001 to T004.
Finally, another external mechanism (a sensor or part detect switch) must be installed to detect the presence of a part traveling on the conveyor as it approaches the robot workspace.

### Line Tracking Interface Board

The line tracking interface board should be inserted into the applicable slot of the power supply unit or main CPU board. See Fig. 2.1.1 (d) to Fig. 2.1.1 (e). If Separate Detector Interface units are used, they can be mounted in the cabinet separately from the main CPU board. See Fig. 2.1.1 (h) to Fig. 2.1.1 (p).

### Fiber Optic FSSB Connector

- When a line tracking interface board, A20B-8101-0421 (wide-mini slot) is used,
  The original fiber optic FSSB cable connector that connects to the COP10A connector of the main CPU board should be moved to COP10A of the line tracking interface board. The additional fiber optic FSSB cable should connect COP10A of the main CPU board to COP10B of the line tracking interface board.
- When a line tracking interface board, A20B-8101-0601 (mini slot) is used,
  The additional fiber optic FSSB cable should connect COP10A of the 6-axis servo amplifier or aux axis servo amplifier to COP10B of the line tracking interface board. See Fig. 2.1.1 (d) to Fig. 2.1.1 (e).
- When αA1000S Pulsecoder is connected to the main CPU board, there are no additional fiber optic FSSB cables.
- If Separate Detector Interface units are used, see Fig. 2.1.1 (p) for a connection diagram.

### Tracking Encoder

For R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus robots, encoders (Pulsecoders) that can be used are shown below.

- αA1000S Pulsecoder A860-0372-T001 (available as both incremental and absolute)
- Incremental Pulsecoder A860-0301-T001 to T004

For R-30*i*B Compact Plus / R-30*i*B Mini Plus robots, encoders (Pulsecoders) that can be used are shown below.

- αA1000S Pulsecoder A860-0372-T001 (available as both incremental and absolute)

Normally, use αA1000S Pulsecoder A860-0372-T001.
Make sure you use appropriate gear or reducer to get desirable resolution (typically 300 pulses or more per mm for line tracking).

> **NOTE**
> Incremental Pulsecoder A860-0301-T001 to T004 is interpolated 4 times on the software. For example, in the case of A860-0301-T004 (resolution is 4000 / rev.), 16000 pulses are read in one rotation.

### Part Detect Switch

A part detect switch must be installed, as a digital input, to monitor when a part on the conveyor is approaching the robot workspace. Refer to "Digital I/O" in R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus / R-30*i*B Compact Plus / R-30*i*B Mini Plus CONTROLLER OPERATOR'S MANUAL (Basic Operation) (B-83284EN) for more information on setting up a digital input. This switch might be one of numerous types including a contact switch, proximity switch, or optical beam device.

> **NOTE**
> You must be aware of the exact location along the conveyor, at which the part will trigger the switch. This location will be used for tracking. Tracking accuracy depends on the precision of the trigger switch. A faster part detect switch gives a more precise trigger value.

### Pulse Multiplexer

When using multiple robots, input the value of the encoder to each robot controller via the pulse multiplexer. Connect the line tracking cables and power cable to the pulse multiplexer as shown in Appendix C. When the pulse multiplexer is used, αA1000S Pulsecoder A860-0372-T001 can not be used. If you want to use multiple robots with αA1000S Pulsecoder, use Ethernet Encoder function (A05B-2600-R762, A05B-2660-R762) (option).

## 2.1.1 Requirements

R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus/ R-30*i*B Compact Plus / R-30*i*B Mini Plus line tracking system requires the items shown below.

**Table 2.1.1 (a) Requirements for each Controller and Pulsecoder**

| Pulsecoder | R-30*i*B / R-30*i*B Plus controller | R-30*i*B Mate / R-30*i*B Mate Plus controller | R-30*i*B Compact Plus / R-30*i*B Mini Plus controller |
|---|---|---|---|
| A860-0372-T001 αA1000S Pulsecoder (which is adaptable to both incremental and absolute.) | Subsection 2.1.1 Requirements Table 2.1.1 (b) | Subsection 2.1.1 Requirements Table 2.1.1 (c) | Subsection 2.1.1 Requirements Table 2.1.1 (d) |
| A860-0301-T001 to T004 incremental Pulsecoder | Appendix C Table C.1 (a) | Appendix C Table C.1 (b) | N/A |

**Table 2.1.1 (b) Requirements (R-30*i*B / R-30*i*B Plus: αA1000S Pulsecoder A860-0372-T001)**

| Required Components | R-30*i*B / R-30*i*B Plus Controller A-Cabinet | R-30*i*B / R-30*i*B Plus Controller B-Cabinet | Comments |
|---|---|---|---|
| **Hardware** | | | |
| Line Tracking Interface Board | A20B-8101-0421 (wide-mini slot)<br>or<br>A20B-8101-0601 (mini slot) | A20B-8101-0421 (wide-mini slot)<br>or<br>A20B-8101-0601 (mini slot) | • The Line tracking interface board in the left column is included in the following order specification.<br>A05B-2600-J035, A05B-2660-J035 (A/B-Cabinet, wide-mini slot)<br>A05B-2600-J036, A05B-2660-J036 (A-Cabinet, mini slot)<br>A05B-2600-J037 (B-Cabinet, mini slot)<br>• Separate Detector Unit (SDU) - A02B-0323-C205 can be used in place of Line tracking Interface board.<br>NOTE: SDU requires retrofit work to mount in the container. (See Fig. 2.1.1 (h) to Fig. 2.1.1 (o).)<br>• αA1000S Pulsecoder can be also connected to Encoder terminal (JD17) on Main CPU board. In this case, Line tracking interface board is made redundant, but Learning Vibration Control function (A05B-2600-J573) (option) cannot be used. |
| Fiber Optic (FSSB) Cable | A66L-6001-0023 | A66L-6001-0023 | • The Fiber Optic cable in the left column is included in the following order specification.<br>A05B-2600-J035, A05B-2660-J035 (A/B-Cabinet, wide-mini slot)<br>A05B-2600-J036, A05B-2660-J036 (A-Cabinet, mini slot)<br>A05B-2600-J037 (B-Cabinet, mini slot) |
| Pulsecoder Cable (In case of an αA1000S Pulsecoder as an incremental type encoder) | for A20B-8101-0421 (wide-mini slot) :<br>A05B-2601-J220 (7M)<br>A05B-2601-J221 (14M)<br>A05B-2601-J222 (20M)<br>A05B-2601-J223 (30M) | for A20B-8101-0421 (wide-mini slot) :<br>A05B-2603-J220 (7M)<br>A05B-2603-J221 (14M)<br>A05B-2603-J222 (20M)<br>A05B-2603-J223 (30M) | • In case of a αA1000S Pulsecoder as an absolute type encoder, cables in this table can not be used.<br>• When the pulse multiplexer is used, αA1000S Pulsecoder cannot be used. If you want to use multiple robots with αA1000S Pulsecoder, use Ethernet Encoder function (A05B-2600-R762) (option). |
| | for A20B-8101-0601 (mini slot) (one Pulsecoder) :<br>A05B-2601-J210 (7M)<br>A05B-2601-J211 (14M)<br>A05B-2601-J212 (20M)<br>A05B-2601-J213 (30M) | for A20B-8101-0601 (mini slot) (one Pulsecoder) :<br>A05B-2603-J210 (7M)<br>A05B-2603-J211 (14M)<br>A05B-2603-J212 (20M)<br>A05B-2603-J213 (30M) | |

| Required Components | R-30*i*B / R-30*i*B Plus Controller A-Cabinet | R-30*i*B / R-30*i*B Plus Controller B-Cabinet | Comments |
|---|---|---|---|
| Hardware | | | |
| Pulsecoder Cable<br>(In case of an αA1000S Pulsecoder as an incremental type encoder) | for A20B-8101-0601 (mini slot)<br>(two Pulsecoders) :<br>A05B-2601-J260 (7M)<br>A05B-2601-J261 (14M)<br>A05B-2601-J262 (20M)<br>A05B-2601-J263 (30M) | A20B-8101-0601 (mini slot)<br>(two Pulsecoders) :<br>A05B-2603-J260 (7M)<br>A05B-2603-J261 (14M)<br>A05B-2603-J262 (20M)<br>A05B-2603-J263 (30M) | • In case of a αA1000S Pulsecoder as an absolute type encoder, cables in this table can not be used.<br>• When the pulse multiplexer is used, αA1000S Pulsecoder cannot be used. If you want to use multiple robots with αA1000S Pulsecoder, use Ethernet Encoder function (A05B-2600-R762) (option). |
| | For Main CPU board :<br>A05B-2601-J270 (7M)<br>A05B-2601-J271 (14M)<br>A05B-2601-J272 (20M)<br>A05B-2601-J273 (30M) | For Main CPU board :<br>A05B-2603-J270 (7M)<br>A05B-2603-J271 (14M)<br>A05B-2603-J272 (20M)<br>A05B-2603-J273 (30M) | |

Refer to Fig. 2.1.1 (a) to Fig. 2.1.1 (c) for Pulsecoder signal information, and images containing the dimensions of the Pulsecoders.
Refer to Fig. 2.1.1 (d) to Fig. 2.1.1 (p) for information on dimensions, connections, and installation of the Detector Interface Units.

**Table 2.1.1 (c) Requirements (R-30iB Mate / R-30*i*B Mate Plus: αA1000S Pulsecoder A860-0372-T001)**

| Required Components | R-30*i*B Mate / R-30*i*B Mate Plus Controller | R-30*i*B Mate / R-30*i*B Mate Plus Controller (Open Air) | Comments |
|---|---|---|---|
| Hardware | | | |
| Line Tracking Interface Board | A20B-8101-0601 (mini slot) | A20B-8101-0601 (mini slot) | • The Line tracking interface board in the left column is included in the following order specification.<br>A05B-2650-J035, A05B-2661-J035 (R-30*i*B Mate / R-30*i*B Mate Plus, mini slot)<br>A05B-2655-J035, A05B-2662-J035 (R-30*i*B Mate / R-30*i*B Mate Plus (Open Air), mini slot)<br>• αA1000S Pulsecoder can be also connected to Encoder terminal (CRS41) on Main CPU board. In this case, Line tracking interface board is made redundant, but Learning Vibration Control function (A05B-2660-J573) (option) cannot be used. |
| Fiber Optic (FSSB) Cable | A66L-6001-0026 | A66L-6001-0023 | • The Line tracking interface board in the left column is included in the following order specification.<br>A05B-2650-J035, A05B-2661-J035 (R-30*i*B Mate / R-30*i*B Mate Plus, mini slot)<br>A05B-2655-J035, A05B-2662-J035 (R-30*i*B Mate / R-30*i*B Mate Plus (Open Air), mini slot) |

| Required Components | R-30*i*B Mate / R-30*i*B Mate Plus Controller | R-30*i*B Mate / R-30*i*B Mate Plus Controller (Open Air) | Comments |
|---|---|---|---|
| Hardware | | | |
| Pulsecoder Cable<br>(In case of an αA1000S Pulsecoder as an incremental type encoder) | for A20B-8101-0601<br>(mini slot)<br>(one Pulsecoder) :<br>A05B-2650-J205 (7M)<br>A05B-2650-J206 (14M)<br>A05B-2650-J207 (20M)<br>A05B-2661-J205 (7M)<br>A05B-2661-J206 (14M)<br>A05B-2661-J207 (20M) | for A20B-8101-0601<br>(mini slot)<br>(one Pulsecoder) :<br>A05B-2655-J205 (7M)<br>A05B-2655-J206 (14M)<br>A05B-2655-J207 (20M)<br>A05B-2662-J205 (7M)<br>A05B-2662-J206 (14M)<br>A05B-2662-J207 (20M) | • In case of a αA1000S Pulsecoder as an absolute type encoder, cables in this table can not be used.<br>• When the pulse multiplexer is used, αA1000S Pulsecoder cannot be used. If you want to use multiple robots with αA1000S Pulsecoder, use Ethernet Encoder function (A05B-2660-R762) (option).<br>• αA1000S Pulsecoder cannot be connected to Main CPU board A (A20B-8200-0790) for R-30*i*B Mate, Main CPU board D (A20B-8201-0420) for R-30*i*B Mate or Main CPU board A (A20B-8201-0750) for R-30*i*B Mate Plus. See Table 2.1.1 (e). |
| | for A20B-8101-0601<br>(mini slot)<br>(two Pulsecoders) :<br>A05B-2650-J215 (7M)<br>A05B-2650-J216 (14M)<br>A05B-2650-J217 (20M)<br>A05B-2661-J215 (7M)<br>A05B-2661-J216 (14M)<br>A05B-2661-J217 (20M) | A20B-8101-0601<br>(mini slot)<br>(two Pulsecoders) :<br>A05B-2655-J215 (7M)<br>A05B-2655-J216 (14M)<br>A05B-2655-J217 (20M)<br>A05B-2662-J215 (7M)<br>A05B-2662-J216 (14M)<br>A05B-2662-J217 (20M) | |
| | For Main CPU board :<br>A05B-2650-J220 (7M)<br>A05B-2650-J221 (14M)<br>A05B-2650-J222 (20M)<br>A05B-2661-J220 (7M)<br>A05B-2661-J221 (14M)<br>A05B-2661-J222 (20M) | For Main CPU board :<br>A05B-2655-J220 (7M)<br>A05B-2655-J221 (14M)<br>A05B-2655-J222 (20M)<br>A05B-2662-J220 (7M)<br>A05B-2662-J221 (14M)<br>A05B-2662-J222 (20M) | |

Refer to Fig. 2.1.1 (a) to Fig. 2.1.1 (c) for Pulsecoder signal information, and images containing the dimensions of the Pulsecoders.
Refer to Fig. 2.1.1 (d) to Fig. 2.1.1 (p) for information on dimensions, connections, and installation of the Detector Interface Units.

**Table 2.1.1 (d) Requirements (R-30*i*B Compact Plus /R-30*i*B Mini Plus: αA1000S Pulsecoder A860-0372-T001)**

| Required Components | R-30*i*B Compact Plus Controller | R-30*i*B Mini Plus Controller | Comments |
|---|---|---|---|
| Hardware | | | |
| Pulsecoder Cable<br>(In case of an αA1000S Pulsecoder as an incremental type encoder) | For Main CPU board<br>(one Pulsecoder) :<br>A05B-2690-J300 (7M)<br>A05B-2690-J301 (14M)<br>A05B-2690-J302 (20M | For Main CPU board<br>(one Pulsecoder) :<br>A05B-2696-J300 (7M)<br>A05B-2696-J301 (14M)<br>A05B-2696-J302 (20M) | • In case of a αA1000S Pulsecoder as an absolute type encoder, cables in this table can not be used.<br>• If you want to use multiple robots with αA1000S Pulsecoder, use Ethernet Encoder function (A05B-2660-R762) (option).<br>• αA1000S Pulsecoder cannot be connected to Main CPU board A (A17B-8100-0800) for R-30*i*B Compact Plus. See Table 2.1.1 (f). |
| | For Main CPU board<br>(two Pulsecoders) :<br>A05B-2690-J310 (7M)<br>A05B-2690-J311 (14M)<br>A05B-2690-J312 (20M) | For Main CPU board<br>(two Pulsecoders) :<br>A05B-2696-J310 (7M)<br>A05B-2696-J311 (14M)<br>A05B-2696-J312 (20M) | |

Refer to Fig. 2.1.1 (a) to Fig. 2.1.1 (c) for Pulsecoder signal information, and images containing the dimensions of the Pulsecoders.
Refer to Fig. 2.1.1 (d) to Fig. 2.1.1 (p) for information on dimensions, connections, and installation of the Detector Interface Units.

**Table 2.1.1 (e) Requirements for R-30*i*B Mate / R-30*i*B Mate Plus Main CPU board (αA1000S Pulsecoder)**

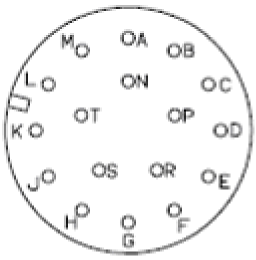
| Main CPU board | Board Specification | Connectable / Non-connectable for αA1000S Pulsecoder | Comments |
|---|---|---|---|
| Main CPU board A | A20B-8200-0790 (R-30*i*B Mate)<br>A20B-8201-0750 (R-30*i*B Mate Plus) | Non-connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H001, A05B-2661-H001 (R-30*i*B Mate)<br>A05B-2655-H001, A05B-2662-H001 (R-30*i*B Mate (Open Air))<br>A05B-2680-H001 (R-30*i*B Mate Plus) |
| Main CPU board B | A20B-8200-0791 (R-30*i*B Mate)<br>A20B-8201-0751 (R-30*i*B Mate Plus) | Connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H002, A05B-2661-H002 (R-30*i*B Mate)<br>A05B-2655-H002, A05B-2662-H002 (R-30*i*B Mate (Open Air))<br>A05B-2680-H002 (R-30*i*B Mate Plus) |
| Main CPU board C | A20B-8200-0792 (R-30*i*B Mate)<br>A20B-8201-0752 (R-30*i*B Mate Plus) | Connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H003, A05B-2661-H003 (R-30*i*B Mate)<br>A05B-2655-H003, A05B-2662-H003 (R-30*i*B Mate (Open Air))<br>A05B-2680-H003 (R-30*i*B Mate Plus) |
| Main CPU board D | A20B-8201-0420 (R-30*i*B Mate) | Non-connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H004, A05B-2661-H004 (R-30*i*B Mate)<br>A05B-2655-H004, A05B-2662-H004 (R-30*i*B Mate (Open Air))) |
| Main CPU board E | A20B-8201-0421 (R-30*i*B Mate) | Connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H005, A05B-2661-H005 (R-30*i*B Mate)<br>A05B-2655-H005, A05B-2662-H005 (R-30*i*B Mate (Open Air))) |

| Main CPU board | Board Specification | Connectable / Non-connectable for αA1000S Pulsecoder | Comments |
|---|---|---|---|
| Main CPU board F | A20B-8201-0422<br>(R-30*i*B Mate) | Connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H006, A05B-2661-H006 (R-30*i*B Mate)<br>A05B-2655-H006, A05B-2662-H006 (R-30*i*B Mate (Open Air))) |

**Table 2.1.1 (f) Requirements for R-30*i*B Compact Plus Main CPU board (αA1000S Pulsecoder)**

| Main CPU board | Board Specification | Connectable / Non-connectable for αA1000S Pulsecoder | Comments |
|---|---|---|---|
| Main CPU board A | A17B-8100-0800 | Non-connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2690-H001 |
| Main CPU board B | A17B-8100-0801 | Connectable | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2690-H002 |

A860-0372-T001




3102A20-29PW

| Signal Name | Pin No. α A1000S 3102A20-29P |
|---|---|
| SD | A |
| *SD | D |
| REQ | F |
| *REQ | G |
| +5V | J,K |
| 0V | N,T |
| Shield | H |
| +6VA | R |
| 0VA | S |

**Fig. 2.1.1 (a) αA1000S Pulsecoder (A860-0372-T001) connection signal information**

| Item | | Specification |
|---|---|---|
| Power voltage | | 5 [V]±5% |
| Current consumption | | Up to 0.3 [A] |
| Working temperature range | | 0 to +60 [℃] |
| Resolution | | 1 000 000 [/rev.]<br>(NOTE: The resolution that is used in Line tracking function is 65 536 [/rev.].) |
| Maximum speed of revolution | | 4000 [$min^{-1}$] |
| Input shaft inertia | | Up to $1 \times 10^{-4}$ [kg $m^2$] |
| Input shaft startup torque | | Up to 0.1 [N m] |
| Ratio loads | Radial | 98 [N] |
| | Axial | 49 [N] |
| Shaft diameter runout | | $0.02 \times 10^{-3}$ [m] |

| Item | Specification |
|---|---|
| Configuration | Dust proof and drip-proof<br>(equivalent to IP55: when using waterproof connector) |
| Vibration proof acceleration | 5 [G] (50 to 2,000[Hz]) |
| Weight | Approx. 0.75 [kg] |

**Fig. 2.1.1 (b) αA1000S Pulsecoder (A860-0372-T001) specifications**

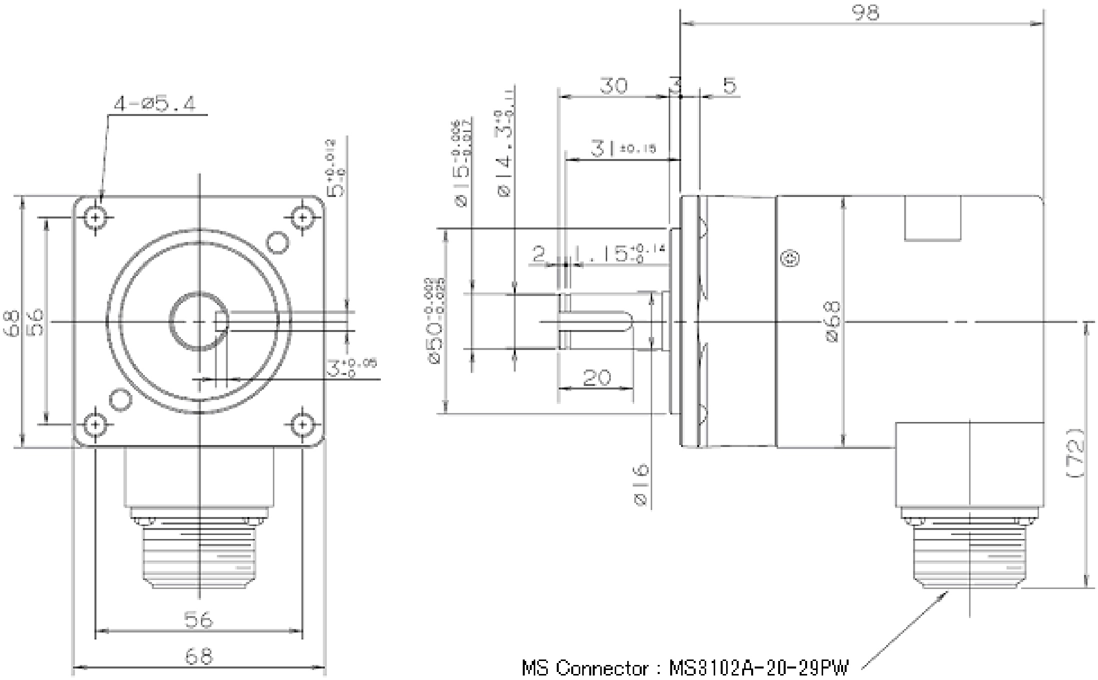



**Fig. 2.1.1 (c) αA1000S Pulsecoder dimensions (A860-0372-T001)**

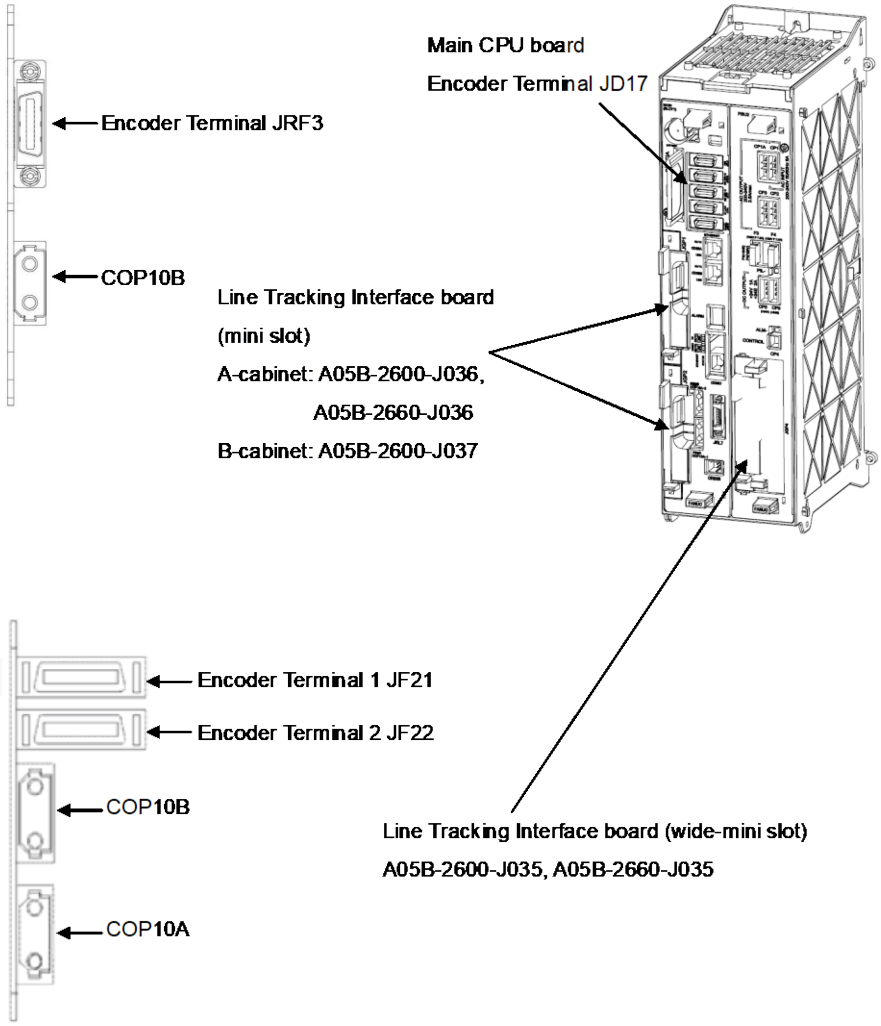



**Fig. 2.1.1 (d) R-30*i*B / R-30*i*B Plus Controller with line tracking connections**

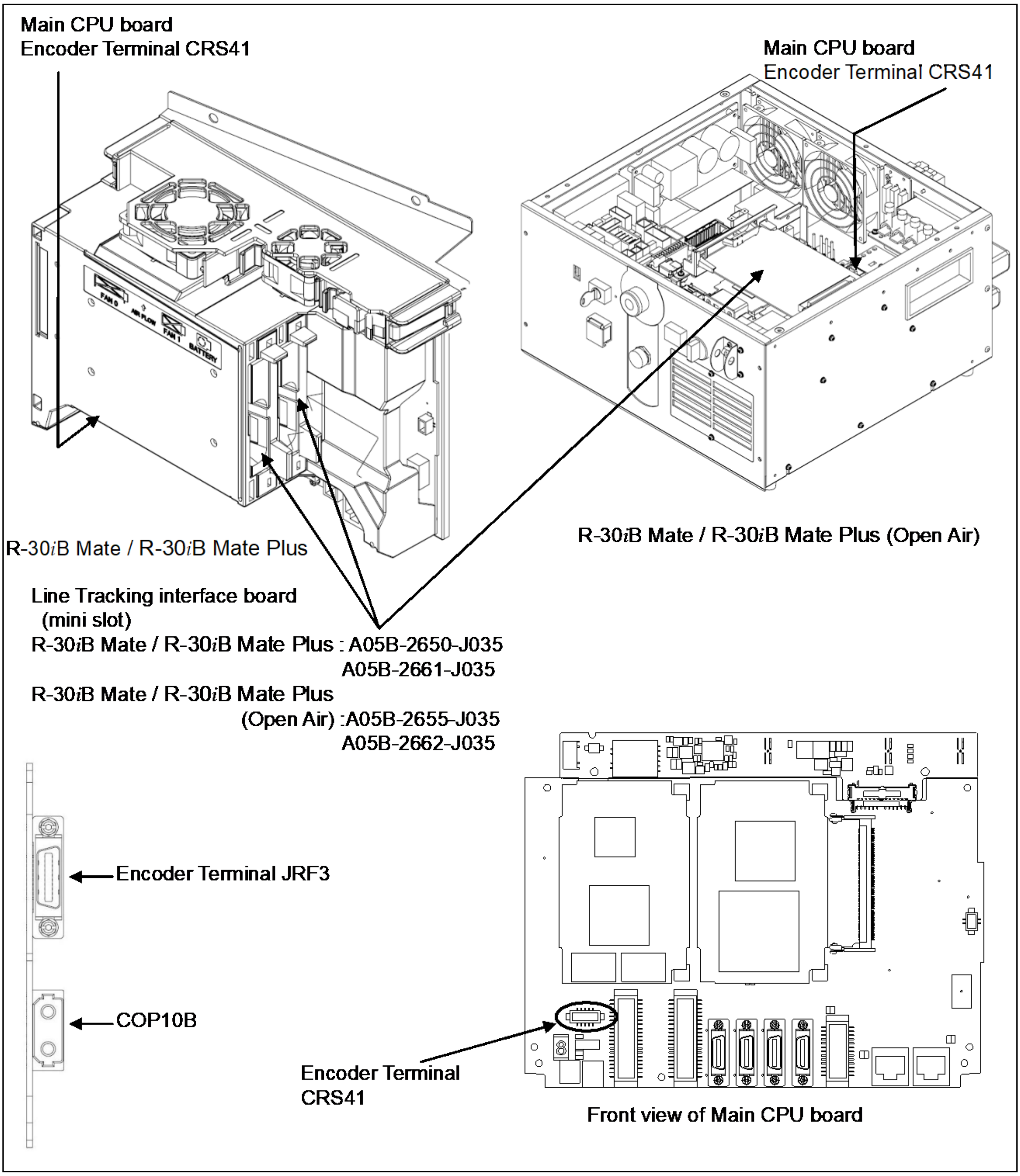



**Fig. 2.1.1 (e) R-30*i*B Mate / R-30*i*B Mate Plus Controller with line tracking connections**

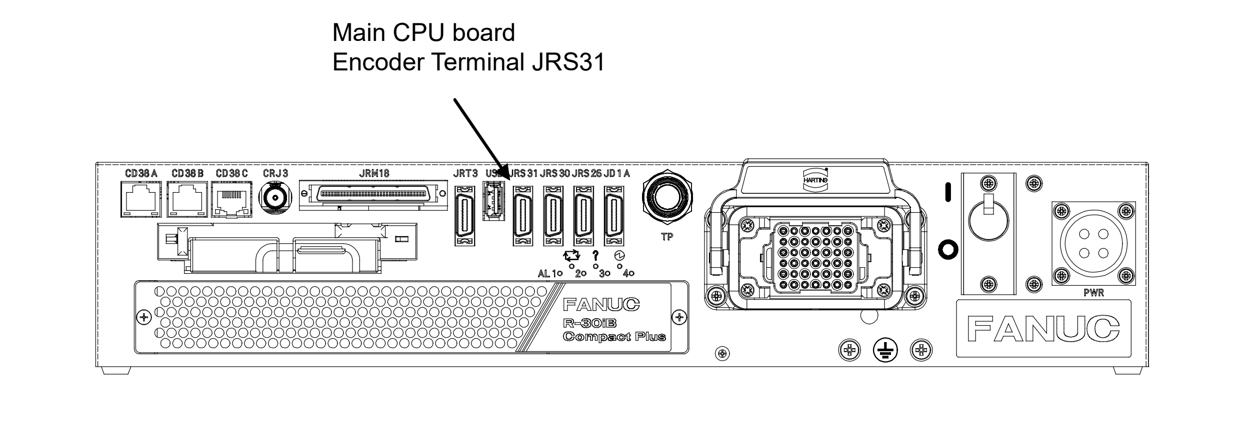



**Fig. 2.1.1 (f) R-30*i*B Compact Plus Controller with line tracking connections**

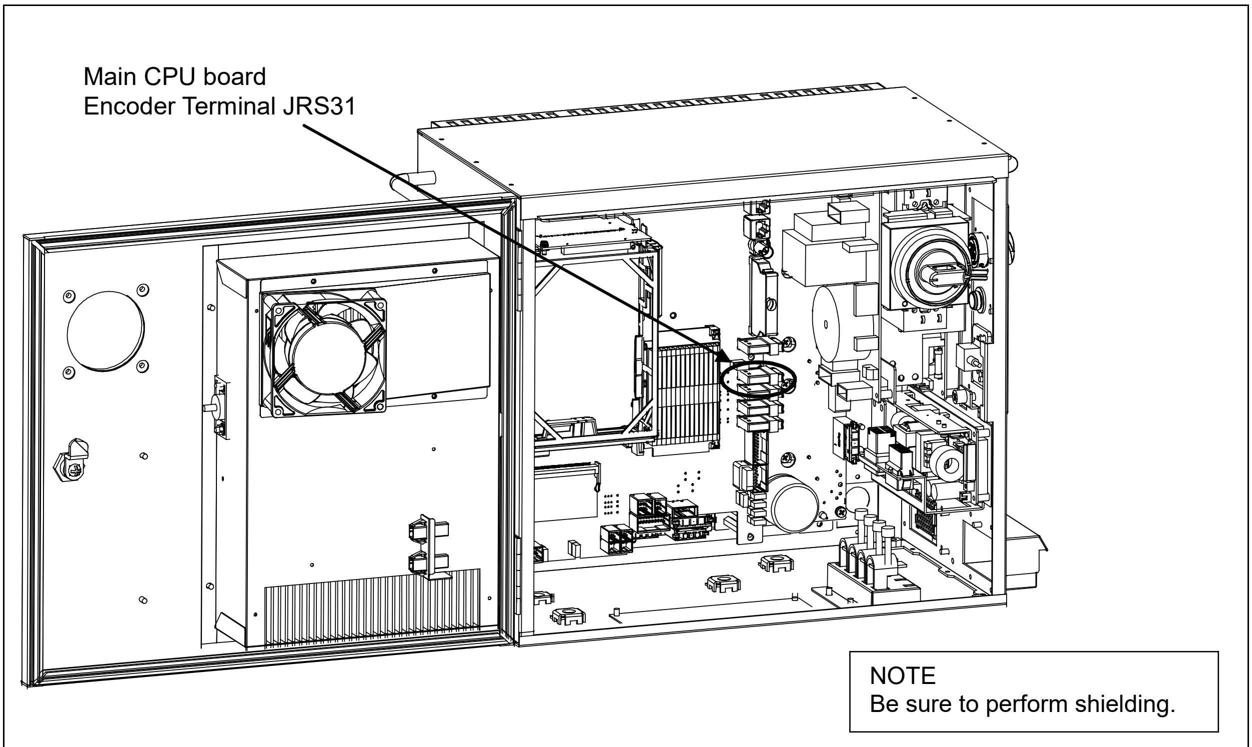



**Fig. 2.1.1 (g) R-30*i*B Mini Plus Controller with line tracking connections**

> **NOTE**
> If the line tracking interface board and main CPU board cannot be used or is not available, you can use the SDU shown in Fig. 2.1.1 (h) to Fig. 2.1.1 (p).

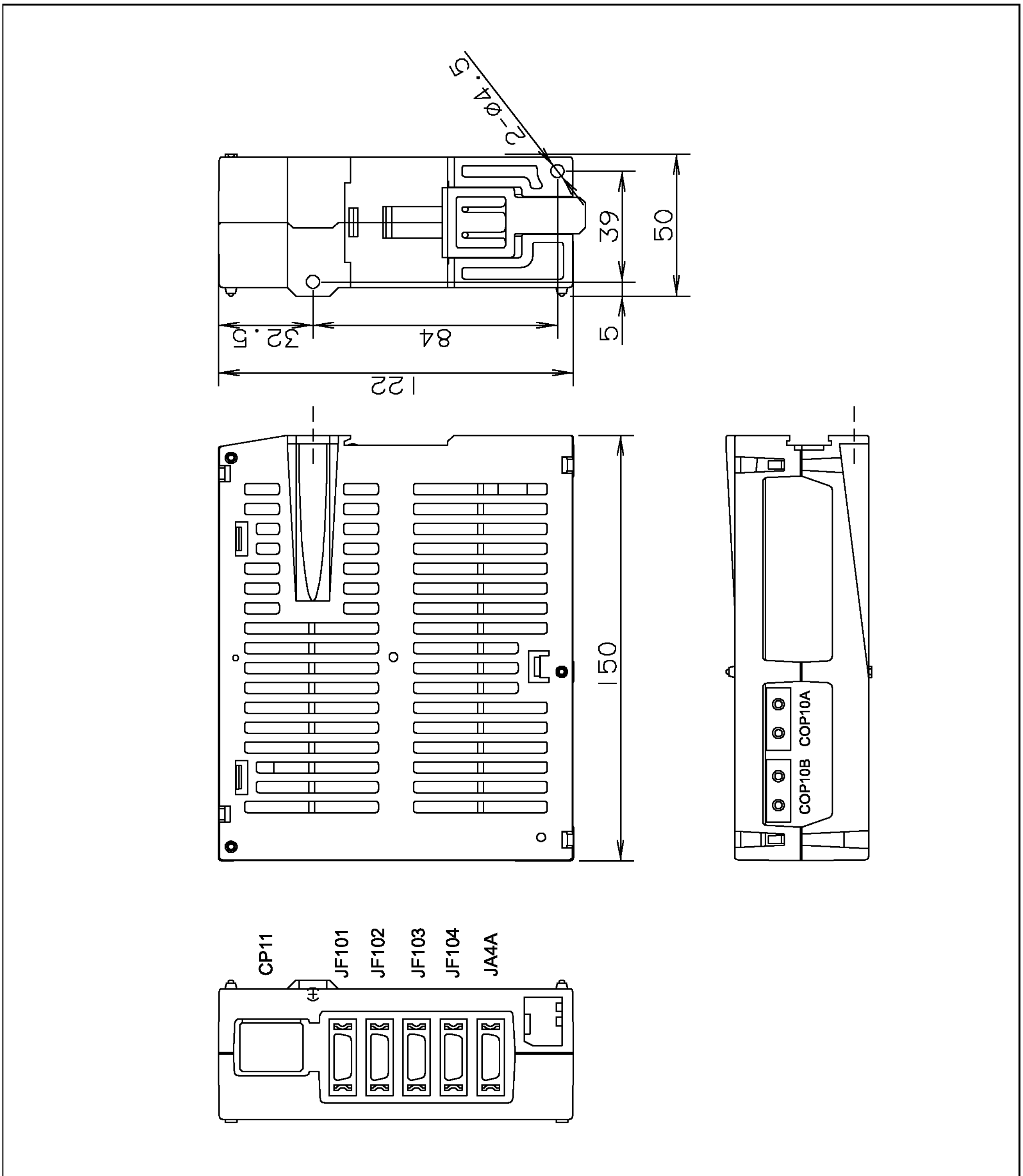



**Fig. 2.1.1 (h) External dimensions of separate detector interface unit**

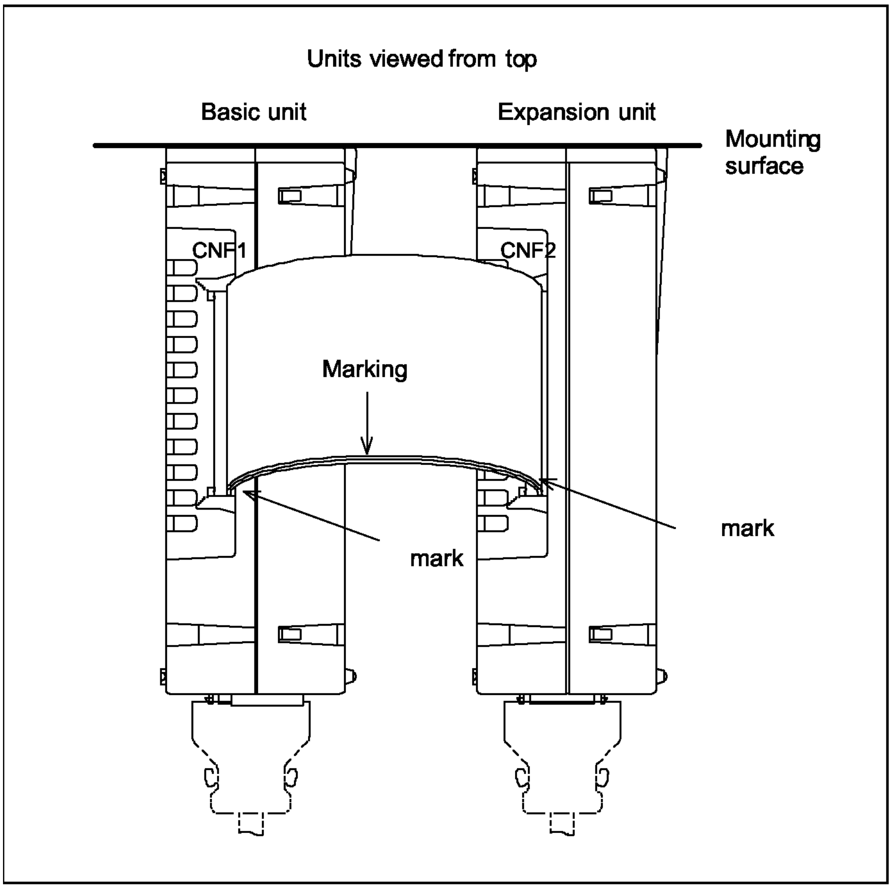



**Fig. 2.1.1 (i) Cable connection between basic unit and expansion unit**

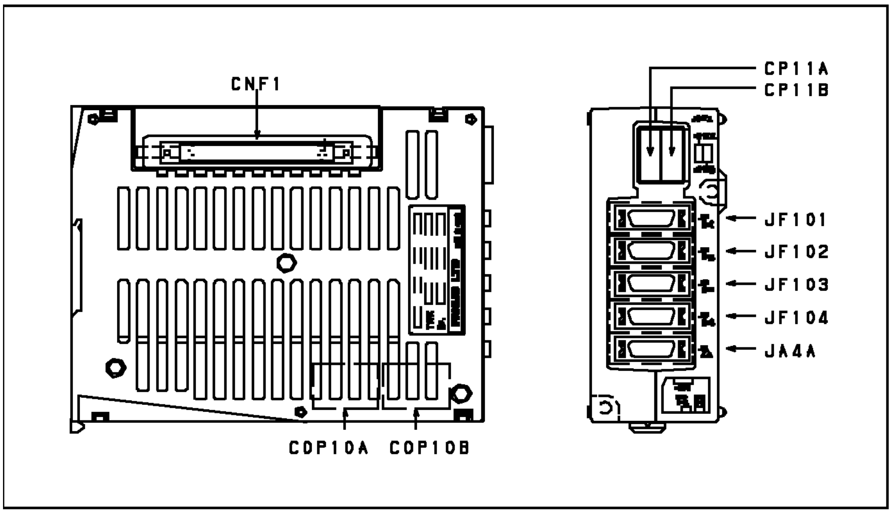



**Fig. 2.1.1 (j) Connector locations on the basic unit**

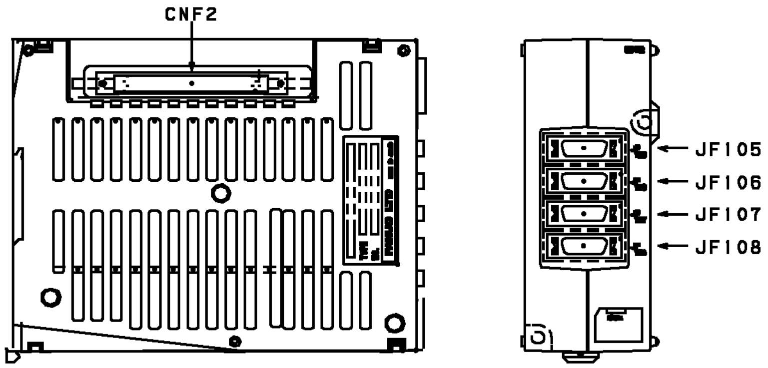



**Fig. 2.1.1 (k) Connector locations on the expansion unit**

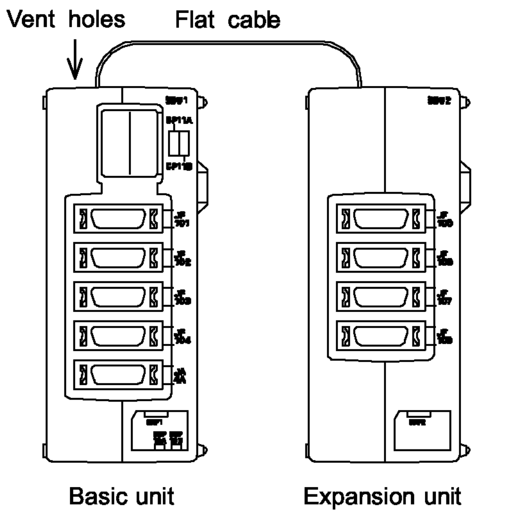



**Fig. 2.1.1 (l) Flat cable placement during installation**

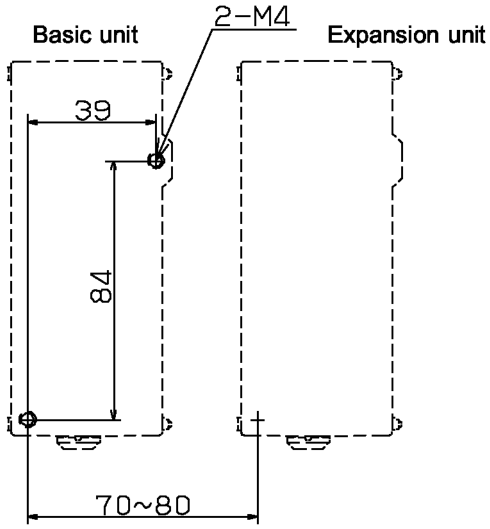
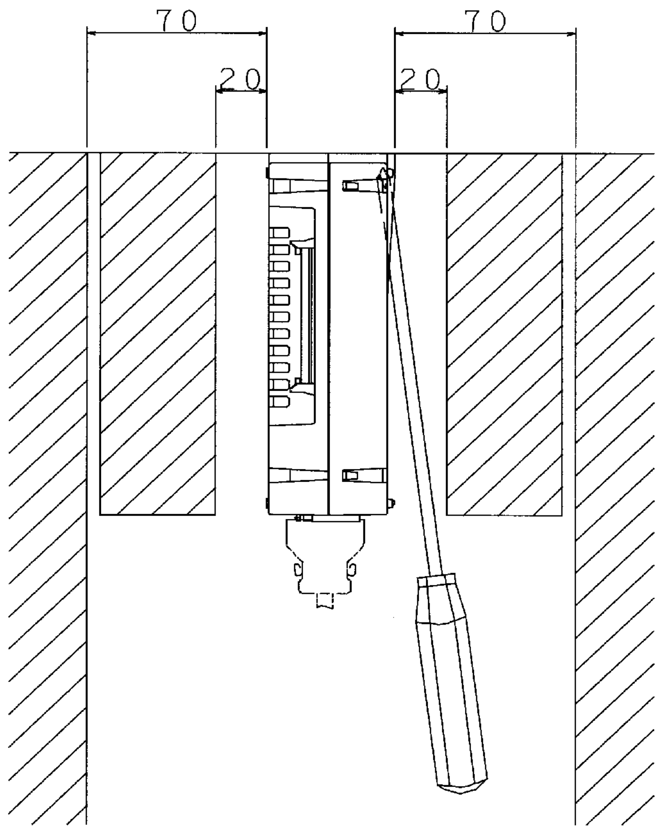



**Fig. 2.1.1 (m) Horizontal separation of mounting holes during installation**

**⚠CAUTION**

To install or remove the unit, you must insert a screwdriver obliquely. Therefore, you must have sufficient access clearance on both sides of the units. As a general guideline, if the front of an adjacent unit appears flush with the unit or slightly set back, allow a clearance of about 20 mm between the two units. If the front of an adjacent unit protrudes beyond the front of the unit, allow a clearance of about 70 mm between the two units. Also, when you are installing the unit near the side of a cabinet, you must allow a clearance of about 70 mm between the unit and the side of the cabinet.




Access clearance near a separate detector interface unit

**Fig. 2.1.1 (n) Accessing the unit**

**⚠CAUTION**

When you are removing the unit, be careful not to damage the lock by applying excessive force. When you are installing and removing the unit, hold the upper and lower ends of the unit so that stress is not applied to the side of the unit (the surface with slits).

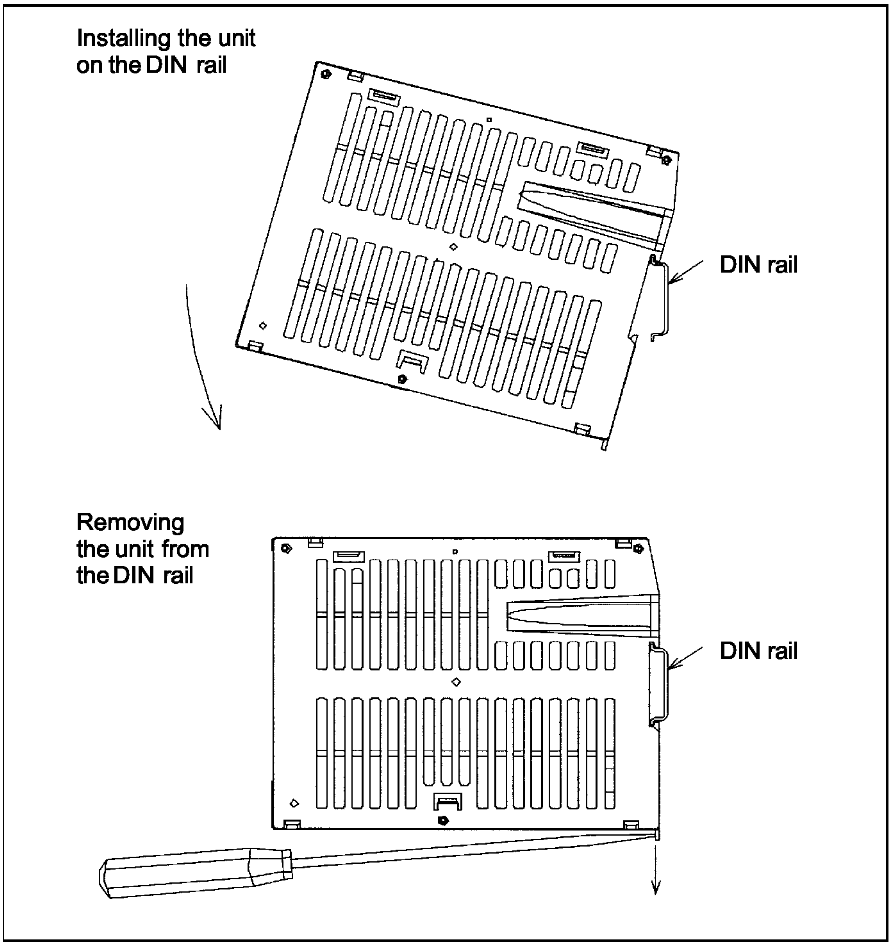



**Fig. 2.1.1 (o) Installing and removing the unit**

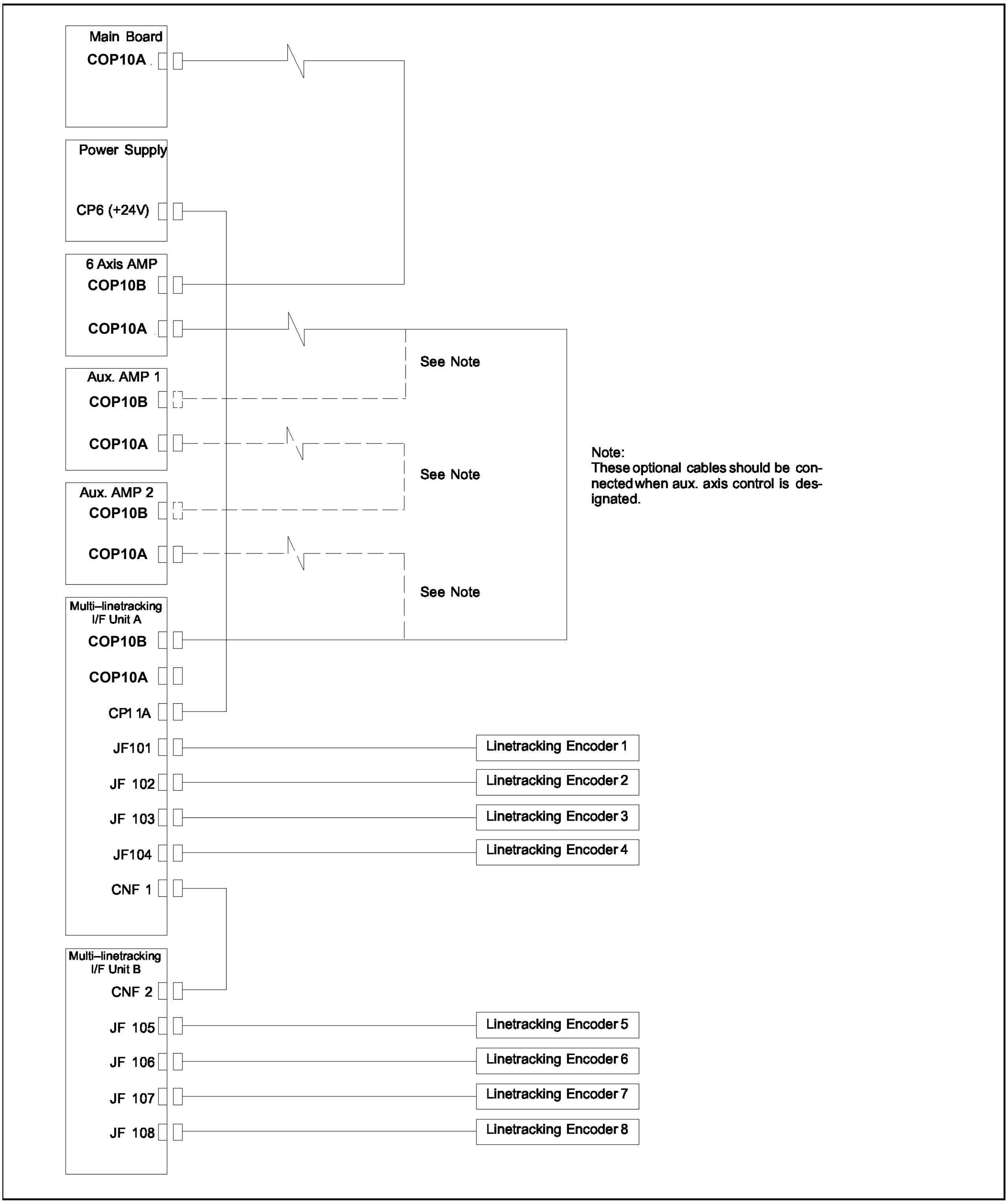



**Fig. 2.1.1 (p) Connection diagram**

## 2.1.2 Installation

The following encoder configuration example described in Table 2.1.2 is available for R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus / R-30*i*B Compact Plus / R-30*i*B Mini Plus Line tracking function.

**NOTE**

Only αA1000S Pulsecoder (A860-0372-T001) can be connected to encoder terminal on Main CPU board.

**Table 2.1.2 Available encoder configuration example (when use αA1000S Pulsecoder)**

| Total Number of Encoders | Encoder configuration example | Comments |
|---|---|---|
| 1 | • **Encoder number 1 :** Connect αA1000S Pulsecoder to Encoder terminal of Main CPU board (R-30*i*B / R-30*i*B Plus: JD17, R-30*i*B Mate / R-30*i*B Mate Plus: CRS41(*1) / R-30*i*B Mate Plus: CRS41(*1) / R-30*i*B Compact Plus: JRS31(*2) / R-30*i*B Mini Plus: JRS31). | |
| 2 | • **Encoder number 1, 2 :** For R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus, connect αA1000S Pulsecoder to JF21-JF22 of Line tracking interface board<br>For R-30*i*B Compact Plus / R-30*i*B Mini Plus, connect αA1000S Pulsecoder to Encoder terminal of Main CPU board (JRS31(*2)). | |
| 3 | • **Encoder number 1, 2 :** Connect αA1000S Pulsecoder to JF21-JF22 of Line tracking interface board.<br>• **Encoder number 3 :** Connect αA1000S Pulsecoder to Encoder terminal of Main CPU board (R-30*i*B / R-30*i*B Plus: JD17, R-30*i*B Mate / R-30*i*B Mate Plus: CRS41(*1)). | • Make sure to set the encoder which is connected to Encoder terminal of Main CPU board as the last Encoder number when composed of a combination of Line tracking interface board and Main CPU board. In the left configuration example, set the Pulsecoder which is connected to the Line tracking interface board as Encoder number 1, 2 and set the Pulsecoder which is connected to the Main CPU board as Encoder number 3. |
| 4 | • **Encoder number 1-4 :** Connect αA1000S Pulsecoder to JF101-JF104 of Separate Detector Unit (SDU) Basic unit. | • SDU requires retrofit work to mount in the container. See Subsection 2.1.1 “Requirements”. |
| 5 | • **Encoder number 1-4 :** Connect αA1000S Pulsecoder to JF101-JF104 of Separate Detector Unit (SDU) Basic unit.<br>• **Encoder number 5 :** Connect αA1000S Pulsecoder to Encoder terminal of Main CPU board (R-30*i*B / R-30*i*B Plus: JD17, R-30*i*B Mate / R-30*i*B Mate Plus: CRS41(*1)). | • Make sure to set the encoder which is connected to Encoder terminal of Main CPU board as the last Encoder number when composed of a combination of Line tracking interface board and Main CPU board. In the left configuration example, set the Pulsecoder which is connected to the Separate Detector Unit (SDU) as Encoder number 1-4 and set the Pulsecoder which is connected to the Main CPU board as Encoder number 5.<br>• SDU requires retrofit work to mount in the container. See Subsection 2.1.1 “Requirements”. |

| Total Number of Encoders | Encoder configuration example | Comments |
|---|---|---|
| 6 | • **Encoder number 1-4 :** Connect αA1000S Pulsecoder to JF101-JF104 of Separate Detector Unit (SDU) Basic unit.<br>• **Encoder number 5, 6 :** Connect αA1000S Pulsecoder to JF105-JF106 of Separate Detector Unit (SDU) Expansion unit. | • SDU requires retrofit work to mount in the container. See Subsection 2.1.1 "Requirements". |
| 7 | • **Encoder number 1-4 :** Connect αA1000S Pulsecoder to JF101-JF104 of Separate Detector Unit (SDU) Basic unit.<br>• **Encoder number 5-7 :** Connect αA1000S Pulsecoder to JF105-JF107 of Separate Detector Unit (SDU) Expansion unit. | • SDU requires retrofit work to mount in the container. See Subsection 2.1.1 "Requirements". |
| 8 | • **Encoder number 1-4 :** Connect αA1000S Pulsecoder to JF101-JF104 of Separate Detector Unit (SDU) Basic unit.<br>• **Encoder number 5-8 :** Connect αA1000S Pulsecoder to JF105-JF108 of Separate Detector Unit (SDU) Expansion unit. | • SDU requires retrofit work to mount in the container. See Subsection 2.1.1 "Requirements". |

(*1) αA1000S Pulsecoder cannot be connected to Main CPU board A (A20B-8200-0790) for R-30*i*B Mate, Main CPU board D (A20B-8201-0420) for R-30*i*B Mate or Main CPU board A (A20B-8201-0750) for R-30***i***B Mate Plus. In such case, Line tracking interface board is required. About connectable Main CPU board of R-30*i*B Mate / R-30***i***B Mate Plus for αA1000S Pulsecoder, see Table 2.1.1 (e).

(*2) αA1000S Pulsecoder cannot be connected to Main CPU board A (A17B-8100-0800) for R-30***i***B Compact Plus. About connectable Main CPU board of R-30*i*B Compact Plus for αA1000S Pulsecoder, see Table 2.1.1(f).

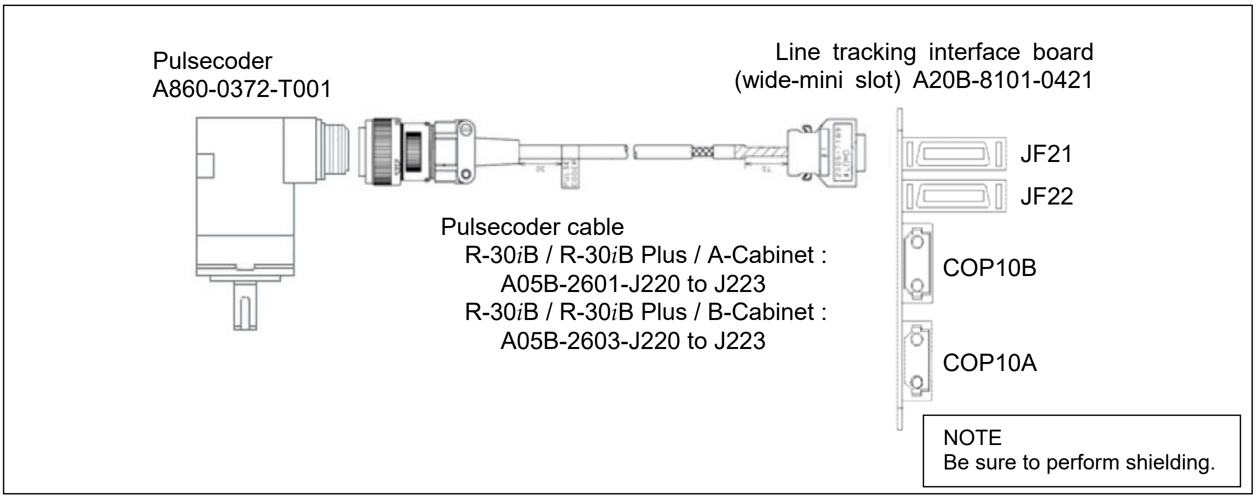



**Fig. 2.1.2 (a) Connecting cables with Line tracking interface board A05B-2600-J035 or A05B-2660-J035 (one αA1000S Pulsecoder)**

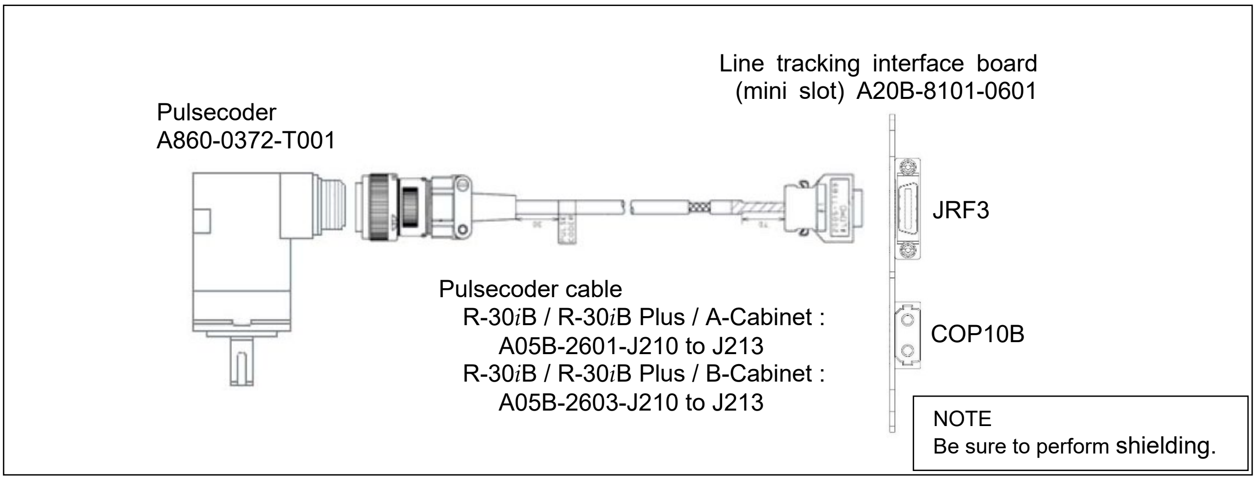



**Fig. 2.1.2 (b) Connecting cables with Line tracking interface board A05B-2600-J036, A05B-2660-J036 or A05B-2600-J037 (one αA1000S Pulsecoder)**

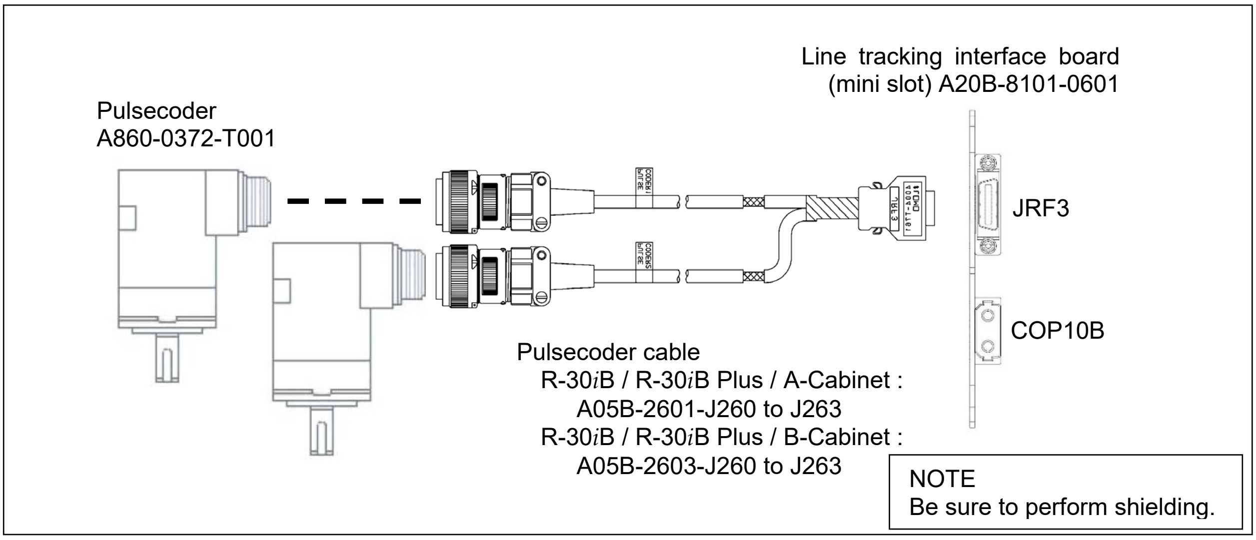



**Fig. 2.1.2 (c) Connecting cables with Line tracking interface board A05B-2600-J036, A05B-2660-J036 or A05B-2600-J037 (two αA1000S Pulsecoders)**

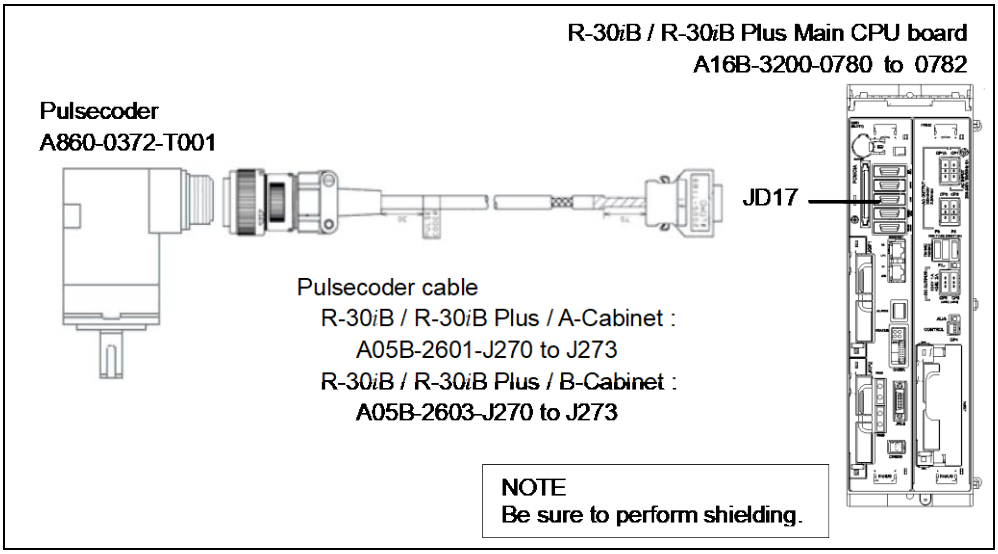



**Fig. 2.1.2 (d) Connecting cables with R-30iB / R-30iB Plus Main CPU board (one αA1000S Pulsecoder)**

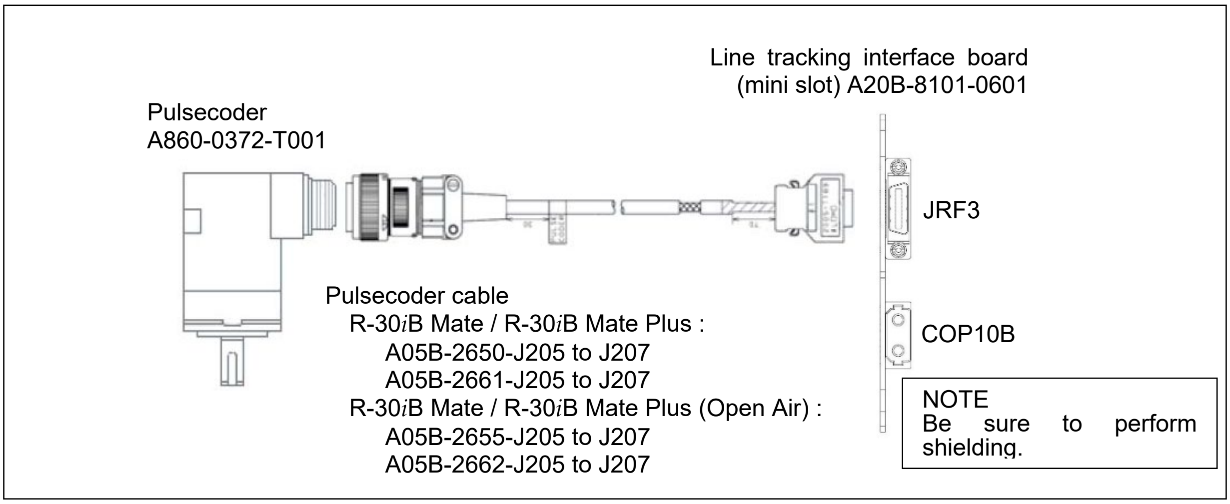



**Fig. 2.1.2 (e) Connecting cables with Line tracking interface board A05B-2650-J035, A05B-2655-J035, A05B-2661-J035 or A05B-2662-J035 (one αA1000S Pulsecoder)**

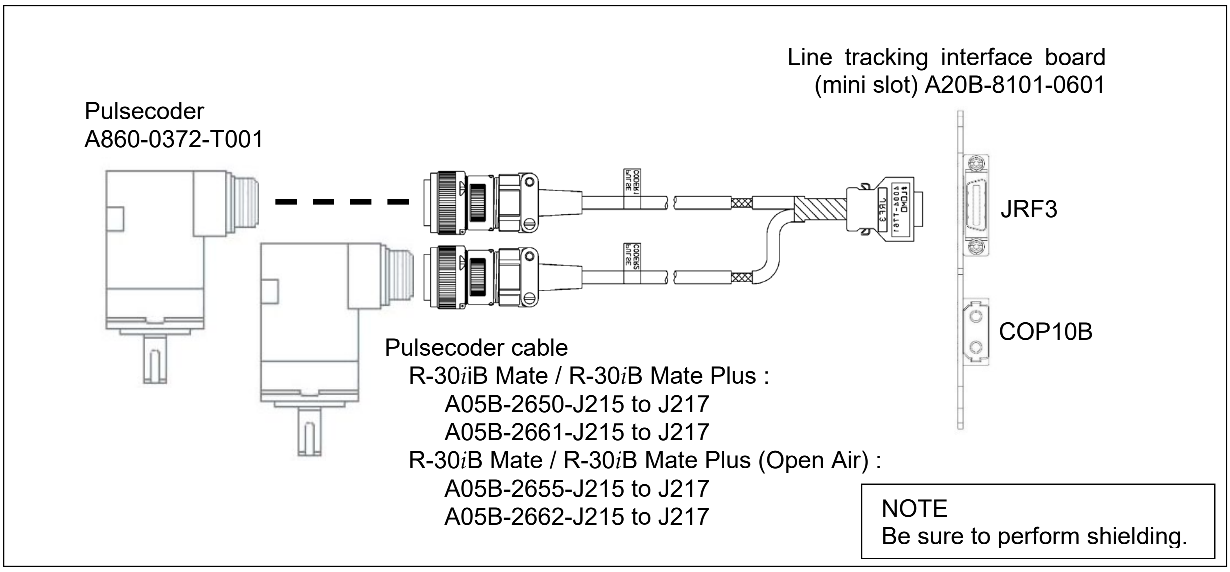



**Fig. 2.1.2 (f) Connecting cables with Line tracking interface board A05B-2650-J035, A05B-2655-J035, A05B-2661-J035 or A05B-2662-J035 (two αA1000S Pulsecoders)**

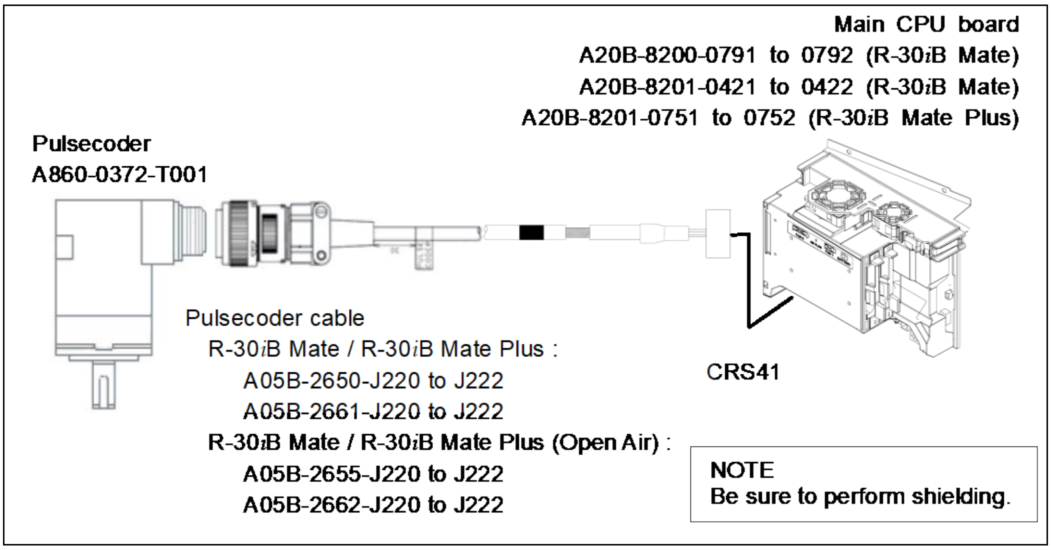



**Fig. 2.1.2 (g) Connecting cables with R-30iB Mate / R-30iB Mate Plus Main CPU board (one αA1000S Pulsecoder)**

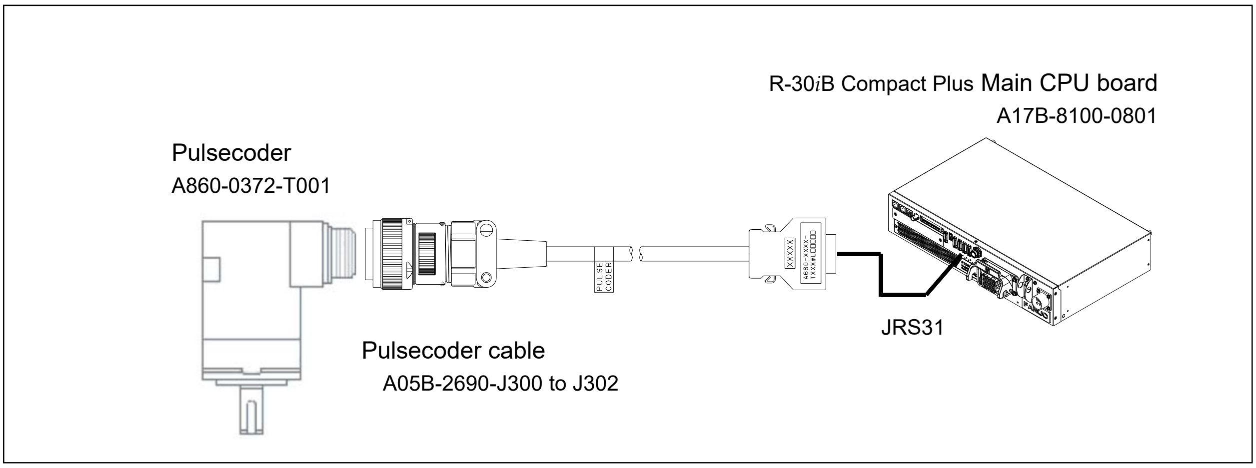



**Fig. 2.1.2 (h) Connecting cables with R-30*i*B Compact Plus Main CPU board (one αA1000S Pulsecoder)**

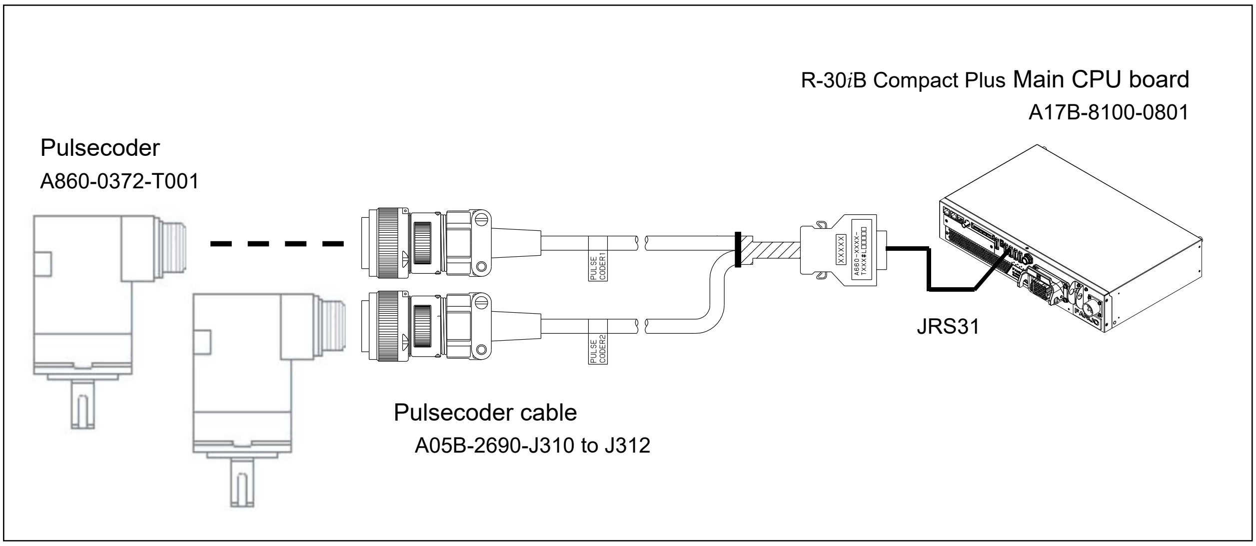



**Fig. 2.1.2 (i) Connecting cables with R-30*i*B Compact Plus Main CPU board (two αA1000S Pulsecoder)**

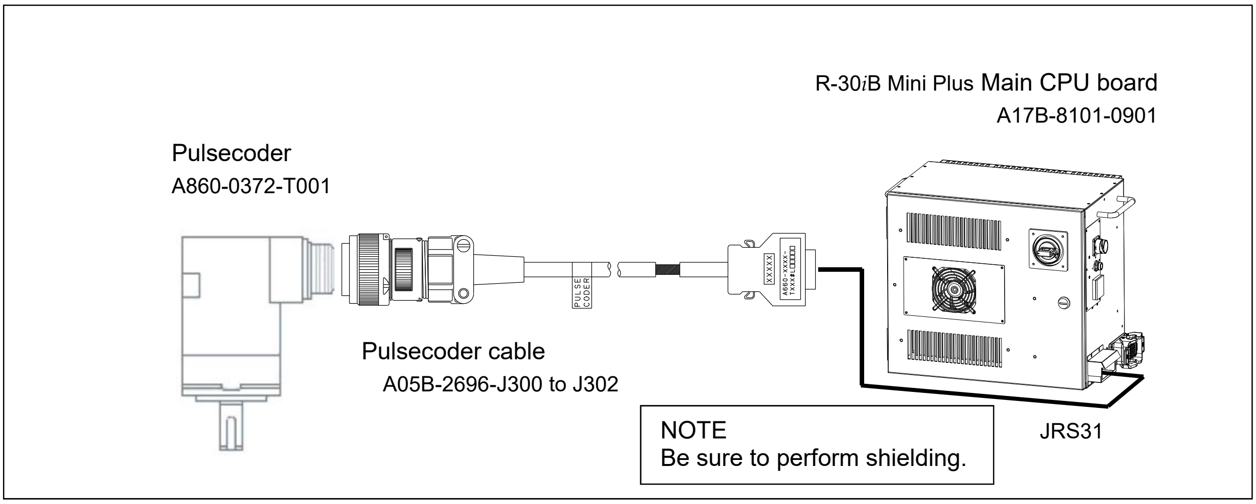



**Fig. 2.1.2 (j) Connecting cables with R-30*i*B Mini Plus Main CPU board (one αA1000S Pulsecoder)**

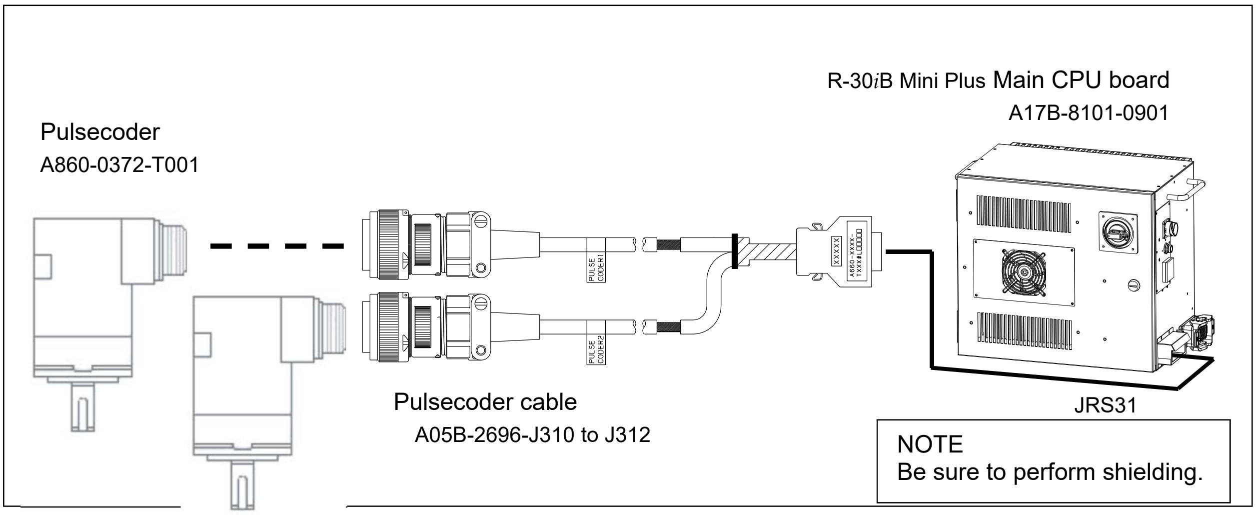



**Fig. 2.1.2 (k) Connecting cables with R-30*i*B Mini Plus Main CPU board (two αA1000S Pulsecoder)**

# 2.2 SOFTWARE

Line tracking software is distributed as an option.

## 2.2.1 Restriction

Line tracking function cannot be used with below functions.

- Palletizing
- Coordinated Motion
- Space Check
- Remote TCP
- Singularity Avoidance (Unavailable for tracking motion. Only available for normal motion)
- Finishing Function Package
- Servo Gun Change function
- Robot Link Function
- Basic/Intelligent Interference Check
- Arc Sensor (TAST)
- AVC
- RPM
- Touch Sensor
- Restart Position check function
- Weaving Function
- Continuous Rotation Function (This can be used together if the motion group is different)
- Softfloat Function
- Touch skip function
- MROT
- Distance before

**NOTE**

For North America Setting (R650), the Constant Path option (R663) must be loaded with the Tracking software option for all applications except painting. Not required for Standard Setting (R651).

**NOTE**

Restart position check function cannot be used with the Line Tracking function. Therefore, please set $USERTOL_ENB=FALSE in system variable screen in order to disable Restart position check function.

# 3 SETUP

"Encoder Setup" and "Tracking Setup" must be set up before a tracking program is created.

## 3.1 ENCODER SETUP

Set the encoder information. Refer to Procedure 3-1 "Encoder Setup". Table 3.1 (b) shows each setting item and system variables.
Once this setting is complete, you should test that the encoder is set up correctly. Refer to Subsection 3.1.2.

### Procedure 3-1 Encoder Setup

**Steps**

1. Press MENU key.
2. Select "6 SETUP".
3. Press F1 key "[ TYPE ]".
4. Select "Encoders". You will see a screen similar to the following.

```
SETUP Encoders
                                              1/8
            Encoder Number :   1
      1 Encoder Axis :                          0
      2 Encoder Type :         INCREMENTAL
      3 Encoder Enable :                      OFF
        Current Count (cnts) :                  0
      4 Multiplier (ITP/update) :               1
      5 Average (updates) :                     1
      6 Stop Threshold (cnt/updt) :             0
      7 Simulate :          Enable :          OFF
      8            Rate (cnt/updt) :            0
      9 Simulation speed Ramp Time (ms):240

   [ TYPE ]          ENCODER  [ CHOICE ]
```

5. To display the encoder information for another Encoder number, press F3 key "ENCODER". This is the schedule selection number of the encoder you are setting up. The default value is 1. For detail, see Subsection 3.1.1 "Encoder Number" which Encoder number you should set the encoder information.
6. Select Encoder Axis. Type the servo axis number to be used for the tracking encoder. Valid values for this field are 1 through 32. For single axis tracking, this field is set to 1. For dual axis tracking, set the Encoder Axis number of Encoder 1 to 1 and set the Encoder Axis number of Encoder 2 to 2. There is no need to setup Encoder Axis number when use αA1000S Pulsecoder which is connected to Main CPU board.

> **NOTE**
> You must perform a COLD start for this change to take effect.

7. Move the cursor to the Encoder Type. This specifies the type of tracking encoder that is to be used. Press F4 key "[CHOICE]" and then the selection screen such as the following is displayed. About each Encoder Type selection, see Table.3.1 (a).

```
SETUP Encoders
                                          1/8
     1           umber :   1
1    INCREMENTAL                            0
2     Serial ABS               INCREMENTAL
3      SERVOCONV :                        OFF
4        VIRTUAL (cnts) :                   0
5     Serial INC P/update) :                1
6 Main Serial INC es) :                     1
7 Main Serial ABS  (cnt/updt) :             0
                         Enable :         OFF
   8          Rate (cnt/updt) :             0
   9 Simulation speed Ramp Time (ms):240

 [ TYPE ]         ENCODER  [ CHOICE ]
```

> **NOTE**
> You must perform a COLD start for this change to take effect.

**Table 3.1 (a) Encoder Type**

| No | Encoder Type | COMMENTS |
|---|---|---|
| 1 | INCREMENTAL | Select "INCREMENTAL" when incremental Pulsecoder (A860-0301-T001 to T004) is used and connected to Line tracking interface board (A20B-8101-0421 (wide-mini slot), A20B-8101-0601 (mini slot)) or Separate Detector Unit (SDU). About the hardware, see Appendix C. |
| 2 | Serial ABS | Select "Serial ABS" when αA1000S Pulsecoder (A860-0372-T001) is used as absolute type encoder and connected to Line tracking interface board (A20B-8101-0421 (wide-mini slot), A20B-8101-0601 (mini slot)) or Separate Detector Unit (SDU). |
| 3 | SERVOCONV | "SERVOCONV" is used when "Servo Conveyor Line Tracking" or "Continuous Servo Conveyor Tracking" |
| 4 | VIRTUAL | This mode allows you to run the line tracking motion with simulated encoder without actual hardware (Line tracking interface board or Separate Detector Unit (SDU)). |
| 5 | Serial INC | Select "Serial INC" when αA1000S Pulsecoder (A860-0372-T001) is used as incremental type encoder and connected to Line tracking interface board (A20B-8101-0421 (wide-mini slot), A20B-8101-0601 (mini slot)) or Separate Detector Unit (SDU). |
| 6 | Main Serial INC | Select "Main Serial INC" when αA1000S Pulsecoder (A860-0372-T001) is used as incremental type encoder and connected to Encoder terminal of Main CPU board (R-30*i*B / R-30*i*B Plus: JD17, R-30*i*B Mate / R-30*i*B Mate Plus: CRS41(*1) / R-30*i*B Compact Plus: JRS31(*2) / R-30*i*B Mini Plus: JRS31). |
| 7 | Main Serial ABS | Select "Main Serial INC" when αA1000S Pulsecoder (A860-0372-T001) is used as absolute type encoder and connected to Encoder terminal of Main CPU board (R-30*i*B / R-30*i*B Plus: JD17, R-30*i*B Mate / R-30*i*B Mate Plus: CRS41(*1) / R-30*i*B Compact Plus: JRS31(*2) / R-30*i*B Mini Plus: JRS31). |

(*1) αA1000S Pulsecoder cannot be connected to Main CPU board A (A20B-8200-0790) for R-30*i*B Mate, Main CPU board D (A20B-8201-0420) for R-30*i*B Mate or Main CPU board A (A20B-8201-0750) for R-30*i*B Mate Plus. In such case, Line tracking interface board is required. About connectable Main CPU board of R-30*i*B Mate / R-30*i*B Mate Plus for αA1000S Pulsecoder, see Table 2.1.1 (e).

(*2) αA1000S Pulsecoder cannot be connected to Main CPU board A (A17B-8100-0800) for R-30*i*B Compact Plus. About connectable Main CPU board of R-30*i*B Compact Plus for αA1000S Pulsecoder, see Table 2.1.1 (f).

**NOTE**

If "1 INCREMENTAL", "2 Serial ABS", or "5 Serial INC" is selected with the controller has no Line tracking interface board (e.g. when connect αA1000S Pulsecoder to Main CPU board), "SRVO-056 FSSB com error 2 (G: Group number A: Axis number)" could occur. Set correct Encoder Type as directed in Table 3.1 (a).

**NOTE**

If "1 INCREMENTAL" is selected but encoder is actually not connected (Line tracking interface board is installed), "SRVO-082 DAL alarm (Track encoder: Encoder number)" warning could occur. If the Encoder Axis number is set to an axis number which is used for motor control, the controller might turn off servo with SRVO-082 warning. If you do not connect encoder, clear the Encoder Axis number to zero.

8. Move the cursor to Encoder Enable. This allows you to turn the specified tracking encoder ON or OFF.
   - To turn ON the encoder, press F4 key "ON". When turned ON, the encoder will update the count value. The encoder must be turned ON for use with both the actual encoder and under simulation.
   - To turn OFF the encoder, press F5 key "OFF".
   - Current Count (cnts) displays the current value for the specified encoder. You cannot modify this value.

**⚠CAUTION**

The Encoder Enable field will automatically reset to OFF after each COLD start. Verify it is set correctly before you run production. In most cases, the encoder ON instruction is used to enable the encoder before calling the tracking program.

9. Select Multiplier (ITP/update). Normally, there is no need to change from 1. Enter a value for the encoder update multiplier. This field allows you to specify how often the multiplier looks at the conveyor, which can save processor time. There will be one encoder update for every interpolation time increment (ITP_TIME: Usually 8msec).

   *Multiplier × ITP_TIME (msec) = encoder 1 update (msec)*

10. Select Average (updates). Enter a value that will help to smooth robot motion when tracking the conveyor. If you have a conveyor that does not move smoothly, set this field to a larger value to make robot motion smooth. A typical encoder average value is 10.
11. Select Stop Threshold (cnt/updt). Type the number of encoder counts per encoder update. If the encoder count per update goes below this number, the system will consider the conveyor stopped.
12. The items "Simulate: Enabled", "Rate (cnt/updt)", and "Simulation speed Ramp Time (ms)" are not used when using a real encoder. Use these items only when running in simulation mode.
    - To simulate the tracking encoder, press F4 key "ON". When turned ON, the encoder counts will be generated based upon the simulation rate value.
    - To use actual encoder counts, press F5 key "OFF". When turned OFF, the encoder counts will be read from the actual encoder when the conveyor is moved.

**NOTE**

The encoder itself must also be turned ON to allow encoder simulation.

**NOTE**

You do not have to plug in a real encoder to simulate. However, if you do not have a real encoder connected, you might get a “SRVO-082 DAL alarm (Track encoder: Encoder number)” error code. This error will not affect the operation of the robot or the simulated line tracking.

13. You must perform a COLD start if you changed Encoder Axis or Encoder Type.
14. Verify that you have set up the encoder correctly. Refer to Subsection 3.1.2.

You have completed Encoder Setup. You can now proceed to Section 3.2, Tracking Setup.

**Table 3.1 (b) Encoder setup items**

| ENCODER SETUP SCREEN PARAMETERS | DESCRIPTION | RELATED SYSTEM VARIABLE |
|---|---|---|
| **Encoder Number**<br>Value: 1 - 8<br>Default: 1 | This item is the schedule selection number of the encoder you are setting up. See Subsection 3.1.1 “Encoder Number” for detail. | N/A |
| **Encoder Axis**<br>Value: 0 – 32<br>Default: 0 | This item allows you to select the Encoder Axis number to set up.<br>NOTE: There is no need to setup Encoder Axis number when use αA1000S Pulsecoder which is connected to Main CPU board. | $SCR. $ENC_AXIS[Encoder number] |
| **Encoder Type**<br>Value: See Table 3.1 (a).<br>Default: INCREMENTAL | This item specifies the type of tracking encoder to be used. See Table 3.1 (a). | $SCR. $ENC_TYPE[Encoder number] |
| **Encoder Enable**<br>Value: OFF (0) / ON (1)<br>Default: OFF (0) | This item allows you to turn the specified tracking encoder ON or OFF. This should also be enabled in the simulation. | $ENC_STAT[Encoder number].<br>$ENC_ENABLE |
| **Current Count (cnts)**<br>Value: Integer | This item displays the current value for the specified encoder. This displays the value read based on the update period calculated by ITP_TIME and Multiplier. | $ENC_STAT[Encoder number].<br>$ENC_COUNT |
| **Multiplier (ITP/update)**<br>Value: 1 - 100<br>Default: 1 | This item allows you to specify how often the multiplier looks at the conveyor, which can save processor time. In most cases you do not need to set a value other than 1. | $ENC_STAT[Encoder number].<br>$ENC_MULTIPL |
| **Average (updates)**<br>Value: 1 - 100<br>Default: 1 | This item is a value that will help to smooth robot motion when tracking the conveyor. When a conveyor does not move smoothly and vibration occurs in movements of the robot, increasing this value to about 10 may improve it. | $ENC_STAT[Encoder number].<br>$ENC_AVERAGE |
| **Stop Threshold (cnt/updt)**<br>Value: Positive Integer<br>Default: 0 | This item is the number of encoder counts per encoder update. If the encoder counts per update go below this number, the system will consider the conveyor stopped. | $ENC_STAT[Encoder number].<br>$ENC_THRESH |
| **Simulate Enable**<br>Value: OFF (0) / ON (1)<br>Default: OFF (0) | This item allows you to turn simulation of the specified tracking encoder ON or OFF. | $ENC_STAT[Encoder number].<br>$ENC_SIM_ON |

| ENCODER SETUP SCREEN PARAMETERS | DESCRIPTION | RELATED SYSTEM VARIABLE |
|---|---|---|
| **Simulate Rate (cnt/updt)**<br>Value: Integer<br>Default: 0 | This item is the desired number of encoder counts per encoder update. This field is used when encoder simulation is enabled. | $ENC_STAT[Encoder number].<br>$ENC_SIM_SPD |
| **Simulation speed Ramp Time (ms)**<br>Value: Integer<br>Default: 240 | This item is the time used for speed change when encoder simulation is enabled. | $ENC_SRAMP_T[Encoder number] |

## 3.1.1 Encoder Number

There are some rules in setting of the Encoder number.

- Rules when connecting to Line tracking interface board or Separate Detector Unit (SDU)
  Make sure to set the encoder information to the Encoder number which is the same as the Encoder terminal number. If the Encoder number differs from the Encoder terminal number, the controller cannot read the encoder data. The following Table 3.1.1 (a) shows correct Encoder number and Encoder terminal number.

**Table 3.1.1 (a) Correct Encoder number and Encoder terminal number**

| Encoder Number | Line tracking interface board | | Separate Detecor Unit (SDU) |
|---|---|---|---|
| | **A20B-8101-0421 (wide-mini slot)** | **A20B-8101-0601 (mini slot)** | **A02B-0323-C205 (Basic unit)<br>A02B-0323-C204 (Expansion unit)** |
| 1 | Encoder which is connected to **JF21**. | Encoder which is connected to **PULSECODER1**. | Encoder which is connected to **JF101**. |
| 2 | Encoder which is connected to **JF22**. | Encoder which is connected to **PULSECODER2**. | Encoder which is connected to **JF102**. |
| 3 | | | Encoder which is connected to **JF103**. |
| 4 | | | Encoder which is connected to **JF104**. |
| 5 | | | Encoder which is connected to **JF105**. |
| 6 | | | Encoder which is connected to **JF106**. |
| 7 | | | Encoder which is connected to **JF107**. |
| 8 | | | Encoder which is connected to **JF108**. |

- Rules when connecting αA1000S Pulsecoder to Main CPU board
  1. When connecting one αA1000S Pulsecoder to Main CPU board
     There is no limitation in setting of the Encoder number. In this case, usually set the encoder information to the Encoder number 1.

  2. When connecting one αA1000S Pulsecoder to Main CPU board, and also connecting one or more encoders to Line tracking interface board or Separate Detector Unit (SDU)
     Make sure to set the encoder information which is connected to Main CPU board to the Encoder number greater than the encoders which are connected to Line tracking interface board or Separate Detector Unit (SDU). If this rule is not kept, the controller cannot read the encoder data. For example, when use Main CPU board in combination with Line tracking interface board, set the encoder information which is connected to Line tracking interface board to Encoder number 1, 2 and set the encoder information which is connected to Main CPU board to Encoder number 3.

3. When connecting two αA1000S Pulsecoders to Main CPU board (R-30*i*B Compact Plus / R-30*i*B Mini Plus)
   Set information of the encoder connected to **PULSECODER1** to Encoder number 1, and set information of the encoder connected to **PULSECODER2** to Encoder number 2.

### 3.1.2 Verify Encoder Setup is Correct

Follow the flowchart below to confirm that the encoder settings are correct.

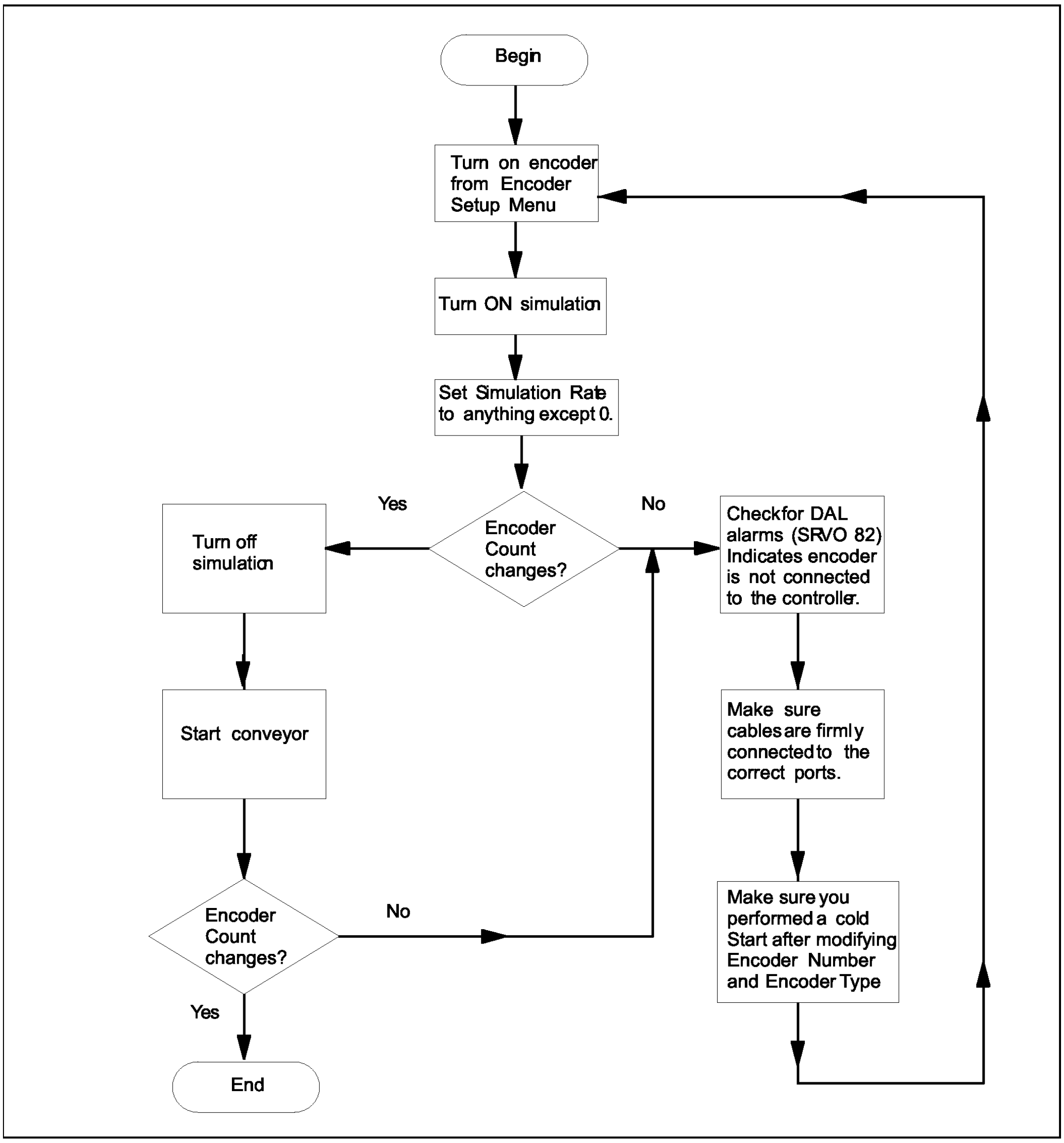



**Fig. 3.1.2 Verify encoder setup**

## 3.2 TRACKING SETUP

This section describes how to set up tracking parameters for your line tracking application.
Tracking setup allows you to set the parameters, listed on the Tracking Setup Screen, for up to eight different schedules.
Refer to Table 3.2 for an overview of each tracking setup item and its related system variable.

## Procedure 3-2 Tracking Setup

### Conditions

- Encoder setup has been performed. Refer to Procedure 3-1.

### Steps

1. Press MENU key.
2. Select SETUP.
3. Press F1, [TYPE].
4. Select 0 --NEXT--.
5. Select Tracking. You will see a screen similar to the following.

```
SETUP Tracking
                                              1/24
            Track Schedule Number :   1
      1 Robot Tracking Group :              1
      2 Tracking Type :                  Line
      3 Visual Tracking :                  NO
      4 Use Vision part queue:             NO
      5 Use Tracking Uframe:               NO
      6 Nominal Track Frame:  Stat:  WORLD
      7 Track (Ext) Axis Num:                0
      8 Track Axis Direction:         POSITIVE
      9 Tracking Encoder Num:                1
     10  Enc Scale Factor (cnt/mm):      500.0

    [ TYPE ]          SCHED
```

6. To display the tracking information for another track schedule number, press F3, SCHED. This specifies which one of the eight schedules is displayed.
7. Encoder number for all tracking schedules is set to 1 by default. If you need to change, select Tracking Encoder Num. Enter a number that specifies the encoder which will be used for all tracking programs that use the current Tracking Schedule Number.
8. Select Robot Tracking Group. Type a number that specifies the robot motion group associated with the current tracking schedule.
9. Move the cursor to Tracking Type. This specifies the type of tracking application.
10. Press F4, [CHOICE].
11. Select the type of tracking for your application. If you are using Rail Tracking, go to Step 15
    - Line Tracking.
    - Rail Tracking.
    - Circular Tracking.

> **NOTE**
> Changing the tracking type changes the values of the Nominal Tracking Frame, Track Axis Number, and Track Axis Direction. The previous values will be stored until either another schedule number is selected, or this SETUP menu is exited. If the Tracking Type is returned to its previous value, before you select another schedule number or exit the SETUP menu, the previous values will be restored.

12. If you are using Tracking User frame, set Use Tracking Uframe to YES. Otherwise, set it to NO.
13. If you are using Line or Circular tracking, move the cursor to Nominal Track Frame.

> **⚠CAUTION**
> Do not set the nominal tracking frame for any schedule that specifies RAIL tracking. The nominal tracking frame is automatically set to the (0,0,0,0,0,0) WORLD frame for RAIL tracking systems.

14. Press F2, DETAIL.
    - If you are using Line tracking, you will see a screen similar to the following.

```
SETUP Frames
      Track Frame Setup  (Line)           4/5
      Track Frame of Schedule:          1
       X:    0.00  Y:    0.00  Z:    0.00
       W:    0.00  P:    0.00  R:    0.00
      Teach Data :
       Origin:UINIT       Enc_cnt:            0
       X:    0.00  Y:    0.00  Z:    0.00
       +X dir:UINIT       Enc_cnt:            0
       X:    0.00  Y:    0.00  Z:    0.00
       +Y dir:UINIT
       X:    0.00  Y:    0.00  Z:    0.00
      Scale (cnt/mm):     500.00

          TEACH  COMPUTE  SCALE
```

    - If you are using Circular tracking, you will see a screen similar to the following.

```
SETUP Frames
      Track Frame Setup  (Circ)           4/5
      Track Frame of Schedule:          1
       X:    0.00  Y:    0.00  Z:    0.00
       W:    0.00  P:    0.00  R:    0.00
      Teach Data :
       +X dir:UINIT       Enc_cnt:            0
       X:    0.00  Y:    0.00  Z:    0.00
       +Y dir:UINIT       Enc_cnt:            0
       X:    0.00  Y:    0.00  Z:    0.00
       Assist:UINIT
       X:    0.00  Y:    0.00  Z:    0.00
      Scale (cnt/mm):      8.73

          TEACH  COMPUTE  SCALE
```

    The Track Frame SETUP menu provides a means for you to specify the nominal tracking frame used within Cartesian tracking systems. You can either enter a value for the nominal tracking frame directly, or teach the frame using the three-point method
    - If you are using the three-point method to set the nominal tracking frame, use Procedure 3-3 or Procedure 3-4 (Subsection 3.2.1) .
    - If you are using the direct entry method to set the nominal tracking frame, use Procedure 3-5 (Subsection 3.2.1).
15. If you are using Rail tracking, select Track (Ext) Axis Num. Enter a number that specifies the extended axis which will be used for tracking the conveyor within RAIL tracking systems. This number will automatically be set to 0 for Line and Circular tracking systems. Valid values are 1-3.
16. If you are using Rail tracking, move the cursor to Track Axis Direction. This specifies the normal forward motion of the conveyor, by comparing it to the motion of the extended axis.
    - If motion is the same as the extended axis, press F4, POSITIVE.
    - If motion is opposite the extended axis, press F5, NEGATIVE.

**NOTE**

The extended axis is used for tracking the conveyor within RAIL tracking systems. The Track Axis Direction is automatically set to POSITIVE for Line and Circular tracking systems.

17. Move the cursor to Encoder Scale Factor. Press F2, TEACH. You will be taken to the Scale Factor Setup screen. Refer to Procedure 3-6 of Subsection 3.2.2 for detailed information about teaching the Scale Factor.
    - For Line and Rail tracking, this specifies the number of encoder counts per millimeter (counts/mm) of conveyor motion.
    - For Circular tracking, this specifies the number of encoder counts per degree (counts/degree) of conveyor motion.
    - This number can be any real number except (0.0). You can also set this directly.
18. Move the cursor to Part Detect Dist. Enter the distance (in millimeters for Line and Rail tracking and in degrees for Circular tracking) from the part detect switch to a user-chosen location relative to the robot world frame. Refer to Fig. 3.2 (a). For example, if you use multiple robots for one conveyor, you can copy the tracking program from one robot to another when you set Part Detect Dist for all the robots. If the position of the trigger switch is changed after creating the tracking program and the current target position shifts to the direction of the conveyor, it can be corrected by changing this value.

**NOTE**
This parameter relies on a correct value for the Encoder Scale Factor Step 17.

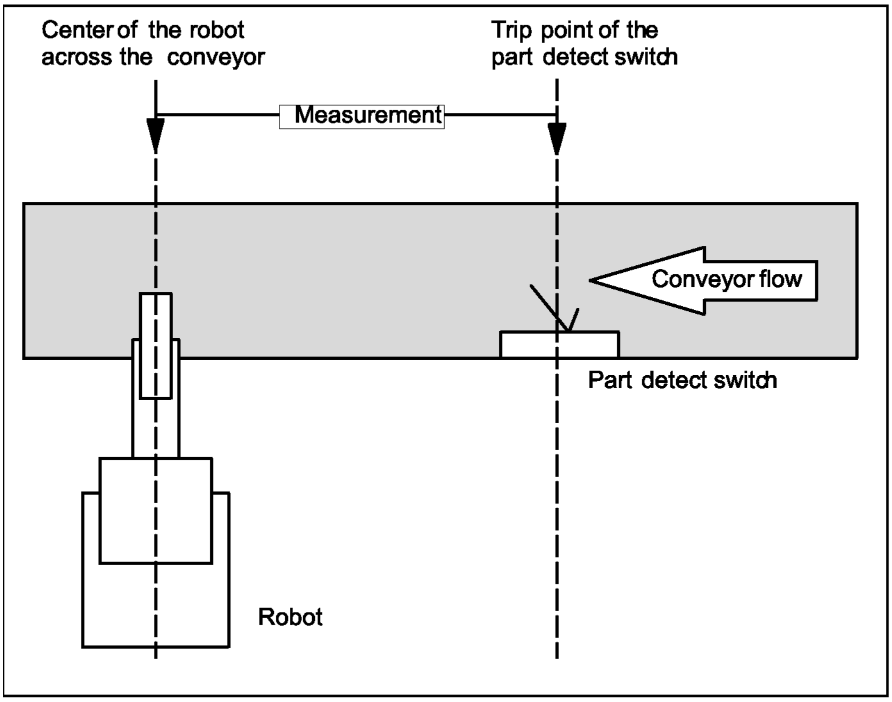



**Fig. 3.2(a) Part detect switch (line and rail tracking)**

19. Move the cursor to Trigger INPUT Number. Type a number to specify the digital input (DI[n] where "n" is a number), which is to be used for the part detect switch input signal. This input is monitored during conveyor synchronization for programs which specify the current Tracking Schedule Number. Valid values range from 0-4096.
    Trigger Value displays the value of the encoder count at the time of the last part detect (as stored by SETTRIG instruction). You cannot modify this value.
20. Encoder Count displays the current count value for the specified encoder. You cannot modify this value.
21. Select Selected Boundary Set. Enter a number of the Boundary Set that this tracking schedule will enable. The Boundary Set makes a area between the two boundaries, the upstream side (Up) and the downstream side (Dn). Refer to Fig. 3.2(b). If a target position of the next motion instruction executed by the tracking program using this tracking schedule is in this area, the robot can execute the instruction.

> **NOTE**
> This section describes how to set up a general and simple Boundary. Chapter 5 describes how to set special Boundaries such as CIRCULAR BOUNDARY and SKEW BOUNDARIES.

> **NOTE**
> Up to 10 Boundary Sets can be set. SETBOUND instruction can be used to change Selected Boundary from within a teach pendant program. Refer to Section 4.3 for more information about SETBOUND instruction.

> **NOTE**
> $LNCFG.$COMP_SW2 bit 0x10 must be enabled in order to use BOUNDARIES for circular tracking. This flag is enabled by default on R-30iB Plus and later controllers.

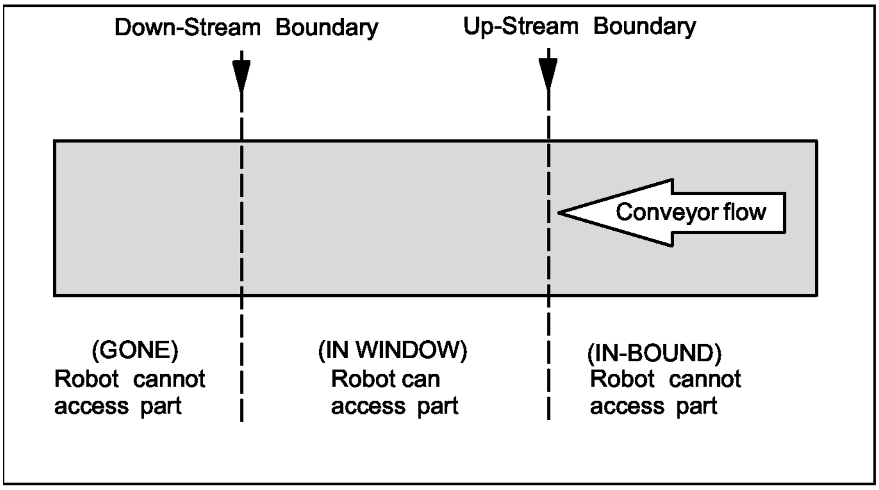



**Fig. 3.3(b) Boundary window**

22. Move the cursor to Boundary Set Up. This specifies the up-stream (IN-BOUND) location of a boundary window set. To record the current position of the robot tracking axis (for rail tracking) or TCP (for line tracking and circular tracking), press SHIFT and F2, RECORD simultaneously. The appropriate boundary value will be extracted and stored as the selected boundary.
    - Conveyor positions further up-stream of this position is considered IN-BOUND. The robot cannot work on the part.
    - Conveyor positions further down-stream of this position are either IN-WINDOW or GONE.
    - This item can also be set by entering a value directly. To initialize the currently selected boundary value to 0.0 (regardless of the value of the corresponding down-stream boundary value), press F4, INIT-BND.

> **NOTE**
> During production, the system will wait until the part travels past the up-stream boundary before the robot will start processing the part. If the conveyor stops at this time, the robot will not execute the next motion command until the conveyor restarts.

> **NOTE**
> This value must be further up-stream than the value of the corresponding down-stream boundary, otherwise a warning message will be displayed.

23. Move the cursor to Boundary Set Down. This specifies the down-stream (OUT-BOUND) location of a boundary window set, where the location is a position along the direction of the conveyor relative to the nominal tracking frame. To record the current position of the robot TCP (relative to the nominal tracking frame), press SHIFT and F2, RECORD simultaneously. The appropriate boundary value will be extracted and stored as the selected boundary.
    - Conveyor positions further up-stream of this position is considered either IN-WINDOW or IN-BOUND.
    - Conveyor positions further down-stream of this position is considered GONE. The robot cannot work on the part.
    - This item can also be set by entering a value directly. To initialize the currently selected boundary value to 0.0 (regardless of the value of the corresponding up-stream boundary value), press F4, INIT-BND.

**NOTE**
If the part travels past the down-stream boundary, an alarm "TRAK-005 Track destination gone error" will be displayed and the robot will stop.

**NOTE**
SETBOUND instruction sets “Boundary Set Up” and “Boundary Set Down”. Refer to Subsection 4.3.

24. Teach and run a simple tracking program to verify. Verify that you have set up tracking correctly. Refer to Subsection 4.1. If you find any problems, refer to Subsection 3.3.

**⚠CAUTION**
Verification should be done before you run production.

This completes the tracking setup. Table 3.2 lists the tracking setup items.

**Table 3.3 Tracking setup items**

| TRACKING SETUP PARAMETERS | DESCRIPTION | RELATED SYSTEM VARIABLES |
|---|---|---|
| Schedule Number<br>Value: 1 - 8<br>Default: 1 | This item is the schedule number for a tracking program. | N/A |
| Robot Tracking GroupValue: 1 - 5<br>Default: 1 | This item specifies the robot motion group associated with the current tracking schedule. | $LNSCH[Schedule_Number].$TRK_GRP_NUM |
| Tracking Type<br>Value: 0 = LINE<br>1 = RAIL<br>2 = CIRC<br>Default: 0 | This item specifies the type of tracking application. | $LNSCH[Schedule_Number].$TRK_TYPE |
| Visual Tracking | This item indicates whether the vision system will be used as the trigger mechanism. | Only used when the vision system is loaded. |
| Use Vision Part Queue | This item indicates whether the vision system will be used to set up the part queue. | Only used when the vision system is loaded. |

| TRACKING SETUP PARAMETERS | DESCRIPTION | RELATED SYSTEM VARIABLES |
|---|---|---|
| Use Tracking Uframe | This item indicates whether Tracking User Frame (Section 5.7) will be used in the current tracking schedule. | $LNSCH[Schedule Number].$USE_TRK_UFM |
| Nominal Tracking Frame<br>Value: Position (status)<br>Default: Uninit. | Set the frame to move according to the conveyor. Boundary data and teaching point data are recorded relative to this frame. The Z direction should match the direction directly above the conveyor. For line tracking, set the X direction of the tracking frame and the direction of the conveyor move to be the same. For Circular tracking, set the origin to the center of the conveyor.<br>This item is not used in Rail tracking. | $LNSCH[Schedule_Number].$TRK_FRAME |
| Track (Ext) Axis Num<br>Value: 0 - 3<br>Default: 0 | This item specifies the extended axis which will be used for tracking the conveyor within RAIL tracking systems. | $LNSCH[Schedule_Number].$TRK_AXS_NUM |
| Track Axis Direction<br>Value:<br>1 = TRUE (positive direction)<br>0 = FALSE (negative direction)<br>Default: 1 | This item is used in RAIL tracking. This item specifies the normal forward motion of the conveyor by comparing it to the motion of the extended axis. For line tracking, set to 1 (TRUE). | $LNSCH[Schedule_Number].$TRK_AXS_DIR |
| Tracking Encoder NumValue: 1 - 8<br>Default: 1 | This item specifies the encoder which will be used for all tracking programs that use the current Tracking Schedule Number. | $LNSCH[Schedule_Number].$TRK_ENC_NUM |
| Encoder Scale Factor(cnt/mm) or (cnt/deg)<br>Value: -999999.0 to 999999.0<br>Default: 500.0<br>Must not = 0.0 | For line and rail tracking, this item specifies the number of encoder counts per millimeter (counts/mm) of conveyor motion. For circular tracking, this item specifies the number of encoder counts per degree (counts/degree) of conveyor motion. | $LNSCH[Schedule_Number].$SCALE |

| TRACKING SETUP PARAMETERS | DESCRIPTION | RELATED SYSTEM VARIABLES |
|---|---|---|
| Part Detect Dist./Degrees(mm) or (deg)<br>Value: Integer<br>Default: 0 | Enter the distance to define the reference position of the teaching point. In most cases, it is the distance from the robot's zero position to the position detected by the sensor along the conveyor. Be sure to enter this value after setting the above Encoder Scale Factor. | $LNSCH[Schedule_Number].$TEACH_DIST |
| Vision Uframe Distance | This item allows you to enter a distance (in millimeters for Line and Rail tracking) from the part detect switch to a location you select where the snap shot of the part is taken. Enter a value only when you use Tracking User Frame (Section 5.7). | $LNSCH[Schedule_Number].$VISUFM_DIST |
| Trigger INPUT Number<br>Value: 0 - 4096<br>Default: 0 | This item allows you to enter a number to specify the digital input (DI[n] where "n" is a number), which is to be used for the part detect switch input signal. | $LNSCH[Schedule_Number].$TRG_DIN_NUM |
| Trigger Value (cnts)<br>Value: Integer<br>Default: 0 (uninit) | This item displays the value of the encoder count at the time of the last part detect (as stored by SETTRIG instruction). | $LNSCH[Schedule_Number].$TRIG_VALUE |
| Encoder Count (cnts)<br>Value: Integer | This item displays the current count value for the specified encoder. | $ENC_STAT[Encoder_Number].$ENC_COUNT |
| Selected Boundary Set<br>Value: 1 - 10<br>Default: 1 | This item specifies which of the boundary window sets are used for all position boundary checking, within programs using the current Tracking Schedule Number. | $LNSCH[Schedule_Number].$SEL_BOUND |
| Bndry Set n Up (mm) or (deg)<br>Bndry Set n Dn (mm) or (deg)<br>Value: -99999.0 to 99999.0<br>Default: 0.0 | This item specifies the upstream location and the downstream location, of a boundary window set. For Rail Tracking, this item specifies a location relative to the zero position of the tracking axis. For Cartesian tracking, this item specifies a position relative to the tracking frame. | $LNSCH[Schedule_Number].$BOUND1[Selected Boundary]<br>$LNSCH[Schedule_Number].$BOUND2[Selected Boundary] |

| TRACKING SETUP PARAMETERS | DESCRIPTION | RELATED SYSTEM VARIABLES |
|---|---|---|
| Bndry Set n Up Skew (deg)<br>Bndry Set n Dn Skew (deg) | This item specifies the angle used by the Up or Down boundary. When this value is 0, the boundary is perpendicular to the conveyor direction. Refer to section 5.2. | $LNSCH[Schedule_Number].$BOUND1_ANG[Selected Boundary]<br>$LNSCH[Schedule_Number].$BOUND2_ANG[Selected Boundary] |

## 3.2.1 Nominal Tracking Frame Setup

The nominal tracking frame is used in a tracking application to provide a coordinate reference frame for all positions and motions referenced with respect to the conveyor.

For line tracking applications:

- Use Procedure 3-3 if you are using the three-point method to set the nominal tracking frame for line tracking.
- Use Procedure 3-5 if you are using the direct entry method to set the nominal tracking frame for line tracking. This method is used when copying from another schedule.

For circular tracking applications:

- Use Procedure 3-4 if you are using the three-point method to set the nominal tracking frame for circular tracking.
- Use Procedure 3-5 if you are using the direct entry method to set the nominal tracking frame for circular tracking. This method is used when copying from another schedule.

For rail tracking applications the system automatically sets this value to be the WORLD (0,0,0,0,0,0) frame.

> ⚠**CAUTION**
> Do not set any USER frame (UFRAME) values for tracking programs. Setting a UFRAME could cause unexpected motion during tracking. If you try to set a UFRAME, you will receive an error message when you try to record a tracking position. The Tracking frame is used (instead of the UFRAME) for all tracking motions.

### Three Point Method

The three-point method is used to teach the nominal tracking frame. During teaching, you move the cursor to each of the three data positions listed under Teach Method Data. A status value is displayed for each of these positions, and will be one of three values:

- UNINIT - indicates that the position is un-initialized
- RECORDED - indicates that the position has been recorded but not yet used during processing
- PROCESSED - indicates that the position has been recorded and already used to compute a new nominal tracking frame

When any of these positions is selected, the word RECORD appears above the F2 function key. Pressing SHIFT and RECORD simultaneously will record the current robot TCP position (to be used during later processing) and will update the position status to RECORDED.

**⚠WARNING**
Be sure the robot UTOOL is properly defined before performing this procedure. The tracking frame cannot be calculated correctly and the robot may make unexpected motions. Refer to “Setting a Tool Coordinate System” in R-30*i*B / R-30*i*B Mate CONTROLLER OPERATOR’S MANUAL (Basic Operation) (B-83284EN) for more information of UTOOL.

## For Line Tracking Applications

When setting the nominal tracking frame for a line tracking application you must be aware of the following:

- The x-axis of this frame must point in the direction of conveyor FORWARD motion. Use Procedure 3-3 to set this, and all other axes of the nominal tracking frame. The y and z-axes are user-definable, but are typically set so that the z-axis points upward from the surface of the conveyor. Refer to Fig. 3.2.1(a).
- The origin location of the nominal tracking frame is arbitrary. You might prefer to set this to the World origin (0,0,0). However, the orientation is very important and should be left as taught using Procedure 3-3. After you have set this value and recorded either boundary or motion positions, do not change this value.

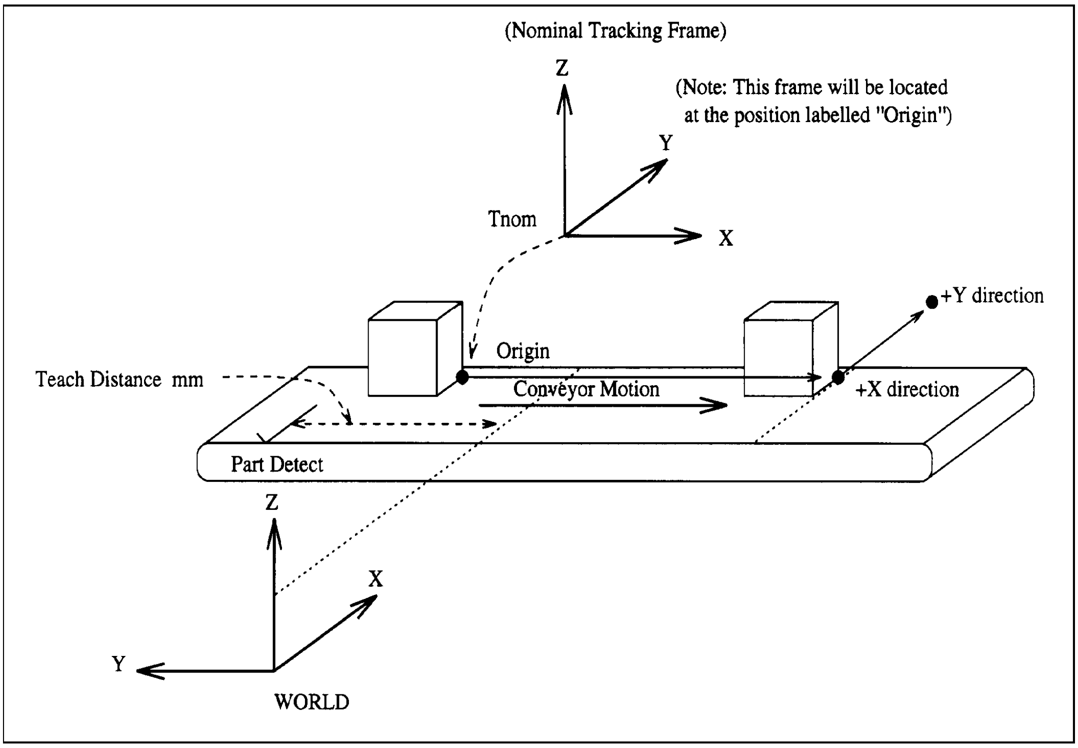



**Fig. 3.2.1(a) Nominal track frame - line conveyor motion**

## Procedure 3-3 Three Point Method to Teach the Nominal Tracking Frame for Line Tracking

## Conditions

- You have selected an encoder number on the tracking schedule setting screen.
- Track Frame Setup screen is displayed. (On tracking schedule setting screen, move the cursor to "Nominal Track Frame:" and then press DETAIL.)

```
SETUP Frames
      Track Frame Setup  (Line)           4/5
      Track Frame of Schedule:          1
       X:    0.00  Y:    0.00  Z:    0.00
       W:    0.00  P:    0.00  R:    0.00
      Teach Data :
       Origin:RECORDED    Enc_cnt:        2356
       X: 2241.80  Y:  754.09  Z:   30.00
       +X dir:RECORDED    Enc_cnt:        2356
       X: 2241.80  Y:  954.09  Z:   30.00
       +Y dir:UINIT
       X:    0.00  Y:    0.00  Z:    0.00
      Scale (cnt/mm):     500.00

          TEACH   COMPUTE   SCALE
```

## Steps

1. Move the cursor to the ORIGIN Teach Data position.
2. Move the robot TCP to a convenient position along the conveyor. (This position should be an easily distinguishable location either on the conveyor or on a part riding on the conveyor.)
3. Record this position by pressing SHIFT and TEACH simultaneously. The status of the ORIGIN position should change to RECORDED. The screen will also update X, Y, Z and Enc_cnt data with current robot TCP position and encoder count value.
4. Move the cursor to select the +X Direction Teach Method Data position.
5. Move the robot away from the part so that the conveyor (and the part) can be moved without running into the robot.
6. Move the conveyor FORWARD (in the direction of normal part flow) for a distance of at least several hundred millimeters (the farther, as long as the robot will still be able to reach the new location of the part.)
7. Stop the conveyor.
8. Move the robot to the same location relative to the conveyor (or part) that was used for the ORIGIN position.
9. Record this position by pressing SHIFT and TEACH. (The status of the +X Direction position should change to RECORDED.) The screen will also update X, Y, Z and Enc_cnt data with current robot TCP position and encoder count value.
10. Move the cursor to select the +Y Direction Teach Method Data position.
11. Without moving the conveyor (or the part), move the robot at least 50mm in the direction perpendicular to the conveyor.
    Typically this is toward the left side of the conveyor, when viewing along the direction of forward conveyor flow such that the resulting z-axis of the nominal tracking frame will point upward from the conveyor.
12. Record this position by pressing SHIFT and TEACH simultaneously. (The status of the +Y Direction position should change to RECORDED.)
13. To process all of the data positions and compute a new nominal tracking frame, press F3, COMPUTE. When the processing is complete, the status of the three Teach method Data positions will be set to PROCESSED, and the Frame Components data values will be updated to display the new nominal tracking frame. See the following screen for an example.

```
SETUP Frames
      Track Frame Setup  (Line)           3/5
      Track Frame of Schedule:          2
       X:  471.39 Y: -133.14  Z:  295.30
       W:    -0.03 P:    1.03  R:   88.24
      Teach Data :
       Origin:RECORDED    Enc_cnt:      109924
       X:  471.39  Y: -133.14  Z:   295.30
       +X dir:RECORDED    Enc_cnt:      203560
       X:  485.22  Y:  316.53  Z:   287.23
       +Y dir:RECORDED
       X:  331.23  Y:  316.53  Z:   287.23
      Scale (cnt/mm):       500.00
      Track Frame computed successfully

         TEACH   COMPUTE   SCALE
```

14. You can setup the encoder scale for this line tracking schedule here or at scale item in the Tracking Setup main menu. If you want to setup it at this time, follow the steps here: Move the cursor to Origin or +X dir. The SCALE function will be display. Calculate the encoder scale by pressing SHIFT and SCALE. The scale value will be updated. If you do not want to setup the encoder scales at this time, refer to Subsection 3.2.2, and set it at next time.

You have completed setup of the nominal tracking frame using the three point method. You can now go back to Tracking Setup at Procedure 3-2.

## For Circular Tracking Applications

When setting the Nominal Tracking Frame for a Circular Tracking application, you must be aware of the following.

- The three points are used to compute the CENTER of the circular conveyor, which is then used as the origin of the Nominal Tracking Frame for Circular tracking.
- The +y position relative to the +x position, must point in the direction of forward conveyor motion. This establishes the orientation of the Nominal Tracking Frame.
- The Assistant position of the nominal tracking frame is arbitrary, but should be located as shown in Fig. 3.2.1(b) or Fig. 3.2.1(c).
- For counter clockwise conveyor motion, the z-axis of the Nominal Tracking frame must point up. Refer to Fig. 3.2.1(b).
- For clockwise conveyor motion, the z-axis must point down. Refer to Fig. 3.2.1(c).
- The x-axis of the Nominal Tracking Frame always points to the +x position used to teach the frame.

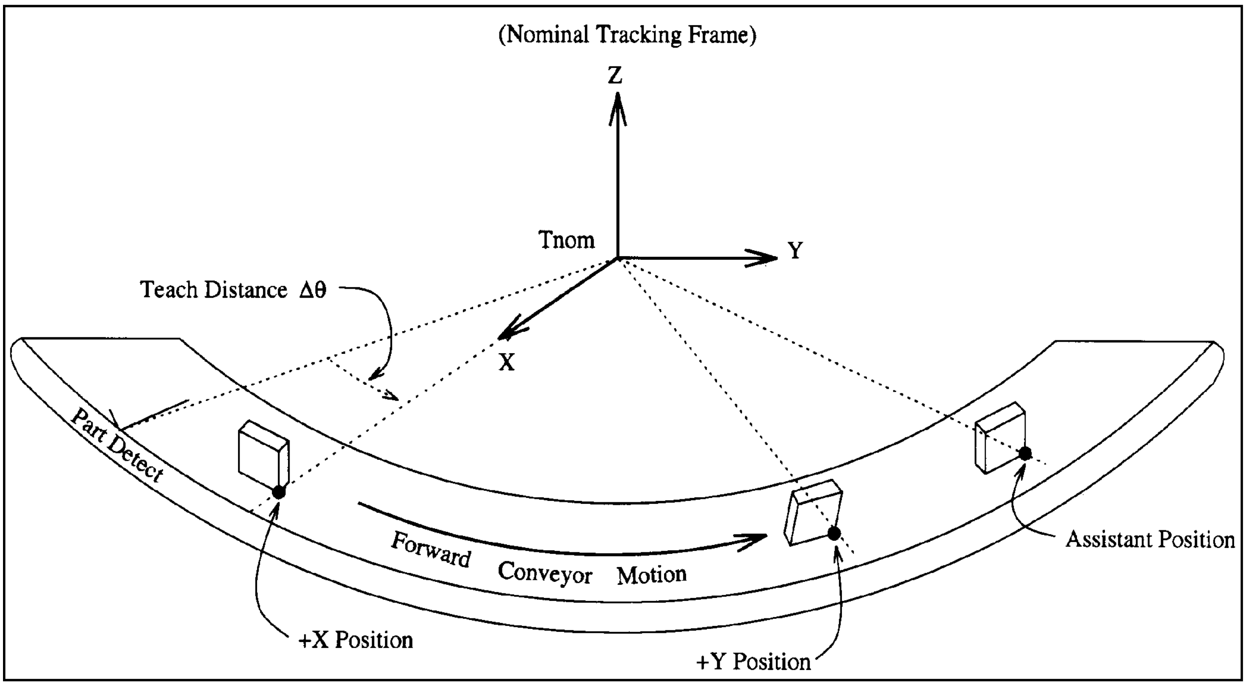



**Fig. 3.2.1(b) Nominal tracking frame - counter clockwise circular tracking**

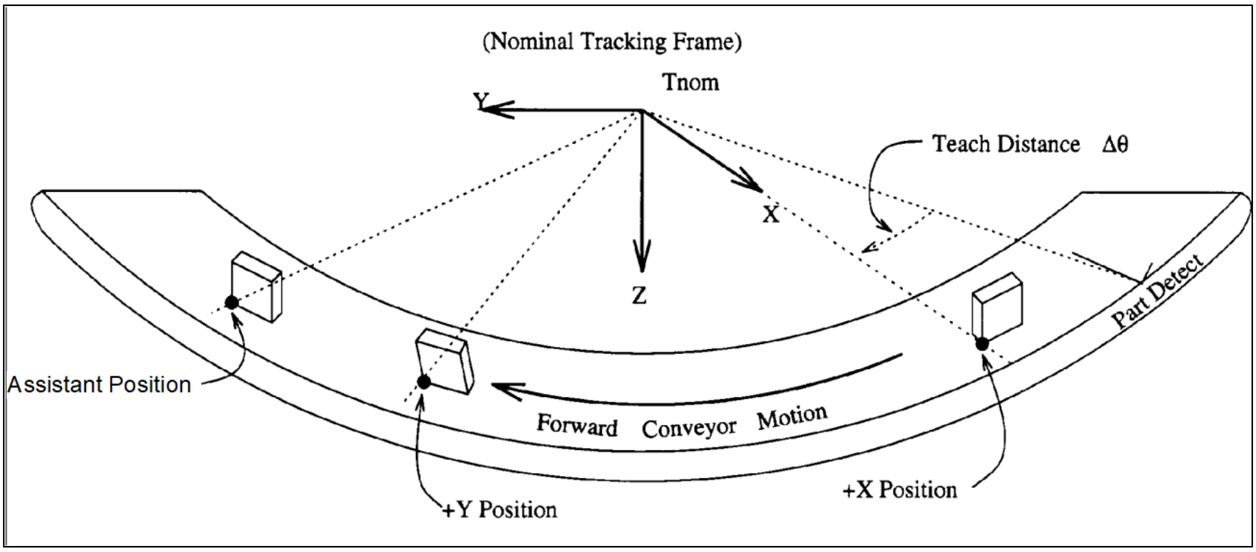



**Fig. 3.2.1(c) Nominal tracking frame - clockwise circular tracking**

## Procedure 3-4 Three Point Method to Teach the Nominal Tracking Frame for Circular Tracking

## Conditions

- You have selected an encoder number on the tracking schedule setting screen.
- Track Frame Setup screen is displayed. (On tracking schedule setting screen, move the cursor to "Nominal Track Frame:" and then press DETAIL.)

```
SETUP Frames
      Track Frame Setup  (Circ)           4/5
      Track Frame of Schedule:          1
       X:    0.00  Y:    0.00  Z:    0.00
       W:    0.00  P:    0.00  R:    0.00
      Teach Data :
       +X dir: UINIT      Enc_cnt:        2356
        X:    0.00  Y:    0.00  Z:   0.00
       +Y dir: UINIT      Enc_cnt:        2356
        X:    0.00  Y:    0.00  Z:   0.00
       Assist:UINIT
       X:    0.00  Y:    0.00  Z:    0.00
      Scale (cnt/deg):     500.00

          TEACH   COMPUTE   SCALE
```

## Steps

1. Move the cursor to select the +X Direction Teach Method Data position.
2. Move the robot TCP to a convenient position along the conveyor. (This position should be an easily distinguishable location either on the conveyor or on a part riding on the conveyor.)
3. Record this position by pressing SHIFT and TEACH simultaneously. The status of the +X Direction position should change to RECORDED.
4. Move the cursor to select the +Y Direction Teach Method Data position.
5. Move the robot away from the part so that the conveyor (and the part) can be moved without running into the robot.
6. Move the conveyor FORWARD (in the direction of normal part flow) for a distance of at least 30 to 40 degrees (the farther the better, as long as the robot will still be able to reach the new location of the part.)
7. Stop the conveyor.
8. Move the robot to the same location relative to the conveyor (or part) that was used for the +X Direction position.
9. Record this position by pressing SHIFT and TEACH. The status of the +Y Direction position should change to RECORDED.
10. Move the cursor to select the Assistant position.
11. Move the conveyor FORWARD (in the direction of normal part flow) for a distance of at least 30 to 40degrees. (The farther the better, as long as the robot will still be able to reach the new location of the part.)
12. Record this position by pressing SHIFT and TEACH simultaneously. The status of the Assistant position should change to RECORDED.
13. To process all of the data positions and compute a new nominal tracking frame, press F4, COMPUTE. When the processing is complete, the status of the three Teach method Data positions will be set to PROCESSED, and the Frame Components data values will be updated to display the new nominal tracking frame. See the following screen for an example.

```
SETUP Frames
      Track Frame Setup  (Circ)           4/5
      Track Frame of Schedule:          1
       X: 1541.80  Y:  564.09  Z:   30.00
       W:    0.00  P:    0.00  R:    0.00
      Teach Data :
       +X dir:RECORDED    Enc_cnt:        2356
       X: 2241.80  Y:  754.09  Z:   30.00
       +Y dir:RECORDED    Enc_cnt:        2356
       X: 2241.80  Y:  954.09  Z:   30.00
       Assist:RECORDED
       X: 2241.80  Y: 1035.09  Z:   30.00
      Scale (cnt/deg):      8.73

          TEACH   COMPUTE   SCALE
```

14. Move the cursor to Origin or +X dir. The SCALE function will be display. Calculate the encoder scale by pressing SHIFT and SCALE. The scale value will be updated.

You have completed setup of the nominal tracking frame using the three point method. You can now go back to Tracking Setup at Procedure 3-2.

### Direct Entry

This method allows you to modify any of the frame component values (x, y, z, w, p, r) directly. This method is usually used when you copy data from another schedule.

### Procedure 3-5 Directly Entering the Nominal Tracking Frame

### Conditions

- You have selected Nominal Track Frame from the Tracking Setup Screen.
- You are currently at the Track Frame Screen.

### Steps

1. Move the cursor to one of the Frame Component values.
2. Press ENTER to select a Frame Component.
3. Enter a new value.
4. Repeat Step 1 - Step 3 for each value (x, y, z, w, p, r) you want to set.

You have finished the nominal tracking frame setup using the direct entry method. You can now go back to Tracking Setup at Procedure 3-2.

## 3.2.2 Scale Factor Setup

The encoder scale factor is the conversion value used to correlate conveyor encoder count value information with conveyor motion.

This value is a real number (in units of encoder counts per millimeter or degrees) representing FORWARD conveyor motion. The sign (+/-) of this value is EXTREMELY important, since the encoder might be wired into the controller in such a way as to provide either increasing or decreasing count values for conveyor FORWARD motion. The sign of this value should not be confused with the value of the Track Axis Direction used for RAIL tracking systems.
The encoder scale factor can be taught instead of computed manually. Use Procedure 3-6 to teach the encoder scale factor.

**NOTE**
This procedure can be done in Track Frame setup procedure for line tracking and circular tracking. If it has been done, you can skip Procedure 3-6.

### Teaching Hints

During Procedure 3-6, the two robot positions (the same position relative to the conveyor or part at two different conveyor positions) and the two corresponding conveyor positions, are recorded internally. The following equation is computed by the controller to determine the encoder scale factor value.

$$scale = \frac{change\ in\ encoder\ counts}{change\ in\ robot\ location}$$

Both conveyor distance and robot positioning accuracy are very important in the above computation. The conveyor should begin at the farthest up-stream end of the robot workspace, positioned so that the robot can still reach the part or marked location on the conveyor, and move to the farthest down-stream end of the robot workspace which meets the same constraints.

You should be very careful to position the robot TCP at the marked position on the part or conveyor, and should be equally precise when repositioning the robot at the second conveyor location. This will provide the highest possible resolution and accuracy for the encoder scale factor computation.

### Procedure 3-6 Teaching the Scale Factor

## Conditions

- Be sure that the robot's tool frame is properly defined before performing this procedure.

## Steps

1. You have selected Scale Factor then F2, TEACH and are currently at the Scale Factor screen. See the following screen (for line/rail tracking) for an example.

```
SETUP Encoder Scale
                                              3/3
       Track Schedule:            1
        Track Scale (cnt/mm):           500.000

         Start Point:    RECORDED
          TCP X:2241.80 Y:  754.09 Z:   30.00
          Encoder Count:                    2356
         End Point:      RECORDED
          TCP X:2241.80 Y:  954.09 Z:   30.00
          Encoder Count:                13567

            TEACH  COMPUTE
```

2. Move cursor to Start Point. Jog the robot TCP to a marked location on the part. Press SHIFT and TEACH simultaneously. The status of the Start Point will change from UNIINT to RECORDED. The TCP location and Encoder Count will be updated.

**NOTE**
After teaching the Start Point and before moving the conveyor, move the TCP out of way so that conveyor can be moved without interference.

**NOTE**
For rail tracking systems that use a non-integrated external axis (rail), only the rail position should be changed during this procedure. Otherwise, the result will be inaccurate.

3. Move the conveyor FORWARD to position that part at the DOWN-STREAM end of the robot workspace.
4. Move cursor to End point. Jog the robot TCP to a marked location on the part. Press SHIFT and TEACH simultaneously. The status of the End Point will change from UNIINT to RECORDED. The TCP location and Encoder Count will be updated.
5. Press SHIFT and COMPUTE simultaneously. The Encoder Scale will be calculated and updated.
6. Move the TCP out of way so that conveyor can be moved without interference.

You have finished teaching the scale factor. You can now go back to Tracking Setup at Procedure 3-2.

# 3.3 VERIFY TRACKING SETUP

## 3.3.1 Robot Does Not Move as Planned

If the robot does not move as planned,

1. Ensure the following items on the tracking schedule setting screen are set correctly.
   - Nominal Track Frame
   - Enc Scale Factor
   - Part Detect Dist
2. Confirm the "Trigger value" on the tracking schedule setting screen is updated correctly when the program is executed. If it has not been updated, change the program.
3. Confirm the tracking system was correctly synchronized before the positions were taught.
4. Ensure encoder simulation is OFF.
5. Check if "SRVO-171 MotorSpd lim / DVC" is displayed in the alarm history. If it is displayed, the motor speed is limited. Please reduce the teaching speed.

## 3.3.2 Robot Does Not Move to Tracking Positions

If the robot fails to move to tracking positions,

1. Ensure the encoder scale factor for schedule has been taught correctly.
2. While the program is running, verify the trigger value displayed updates once at the beginning of the program.
3. While the program is running, verify the encoder currency count is updating and that the value is valid.
4. Ensure encoder simulation is OFF.
5. Ensure the boundary values for schedule are correct, particularly if the error message TRAK-005 Track Destination Gone is displayed.

## 3.3.3 Tracking Motion Vibrates

Causes of robot movement vibration during tracking include low encoder resolution, unstable conveyor speed, and short tracking filters.

### The Solution for low encoder resolution

Add gear to increase encoder resolution (recommended value is 300 pulses / mm).

> **NOTE**
> If hardware improvement is difficult, enable software gear.
> Enable the software gear by turning on bit 3 of $LNCFG.$COMP_SW ( Add value 8 if it was not turned on before). Then cycle the power. After that because the gear ratio would not be the same any more, you need to re-teach the scale of the tracking schedule using F2, TEACH on the SETUP Tracking menu. Refer to Section 3 for more information. Or re-enter the scale (cnt / mm or cnt / deg) in the tracking schedule that uses the encoder by multiplying the previous value of the teach scale by the value of $ENC_SCALE[x] where x is the encoder number in the tracking schedule.

### The Solution for Unstable conveyor speed

The tracking motion may vibrate when the conveyor speed is not stable. Increase the "Average" value on the encoder setting screen.

## The Solution for Short Tracking Filter

If the robot vibrates at the beginning or end of the tracking motion, increase the length of the tracking filter. The filter length is the time robot has to catch up with the parts on the conveyor. When the filter length is short, the robot is heavy and the conveyor speed is high, the robot will be commanded to accelerate too fast. The motion would look aggressive if the robot can accelerate. A collision alarm would occur if it is beyond the robot's capability.

Therefore, for a large robot and high conveyor speed you should adjust $LNSCH[schedule number].$TRK_FLTR_LN (R-30iB Plus : Maximum 80, R-30iB : Maximum 40) accordingly to a higher value. Cycle power after setting.

> **NOTE**
> When using servo conveyor tracking function, Set $SLTK_GRP[group].$SL_FLTR_LEN instead of $LNSCH[schedule number].$TRK_FLTR_LN.

# 4 PROGRAM

A typical tracking system program consists of a main program and a tracking program (a subprogram that executes tracking motion). Before teaching or running the tracking program, trigger value must be updated to synchronize the robot with the conveyor. The trigger value is an encoder count value when a sensor detects a target moving on the conveyor. There are two ways to synchronize, one is to use tracking instructions when a tracking program is executed, and the other is to execute a synchronization process when program edit screen is opened. Teaching position of the tracking program records results of calculating positions relative to the trigger position.

> **NOTE**
> Tracking programs need to be set tracking schedule number from program details screen before teaching. If the tracking schedule is set to 0, it means that it is not a tracking program.

## 4.1 Program creation and confirmation

A typical tracking system consists of a conveyor, an encoder that measures the amount of movement, a sensor that detects parts, and a robot. This section explains how to create a program by taking a program in which a robot moves as shown in Fig. 4.1 as an example. After moving to the home position P [1], the robot waits for the part to cross an upstream boundary. The movement to P[2] and P[3] is executed by the tracking program and follows the movement of the conveyor.

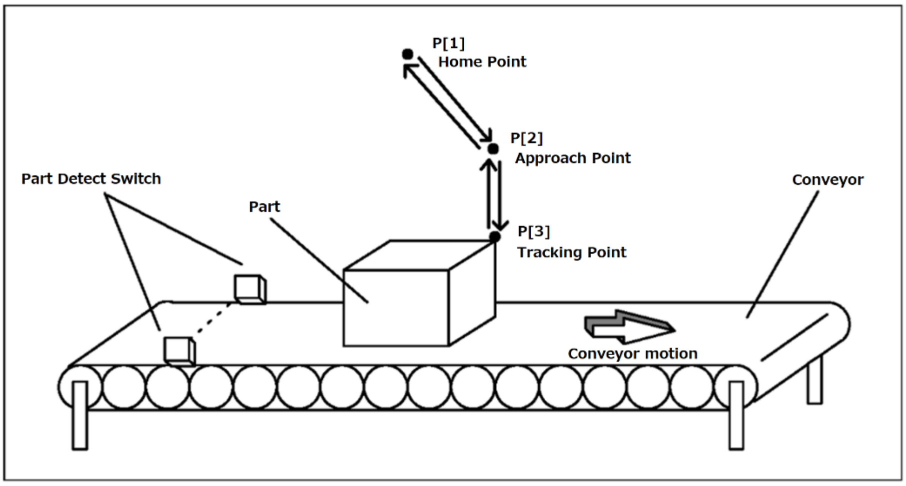



**Fig. 4.1 An example of a typical tracking system**

> **NOTE**
> For more information on tracking instructions, refer to Section 4.3.

> **NOTE**
> The sample program described here assumes a system in which the sensor detects a next part after the robot has completed all motions. For systems where the switch needs to detect the next part before the robot completes a tracking program, it is necessary to consider how to set the trigger value. The system can use Tracking Part Queues to buffer trigger values. Please refer to section 5.4

## Procedure 4-1 Creation and Confirmation of Trigger Value Setting Program (TRIGGER)

### Conditions

- The hardware used for line tracking is properly connected.
- Encoder information and tracking schedule parameters have already been set.
- The encoder is enabled and simulation is turned off.

### Steps

1. Create a new program "TRIGGER".
2. Edit TRIGGER by referring to Example 4.1(a). If necessary, change the encoder number, parts detection signal, register number, and tracking schedule number according to the system configuration.

**Example 4.1 (a) A program to update the trigger value (TRIGGER)**

```
1: WAIT DI[1]=ON          -- Wait for the sensor (DI [1]) to detect the part
2: LINECOUNT[1] R[1]      -- Save the count value of encoder[1] to R[1]
3: SETTRIG LNSCH[1] R[1]  -- Set trigger value of schedule[1] to value of R[1]
[End]
```

3. Place a part upstream on the conveyor from the trigger switch.
4. Run TRIGGER and wait on the first line.
5. Start the conveyor and the trigger switch will detect the part. When the work is within the operating range of the robot, stop the conveyor.
6. Display the tracking setting screen and confirm that the value of "Trigger Value (cnts)" has been updated.

> **NOTE**
> There is no problem even if the instructions for updating the trigger value are executed in the main program.

> **NOTE**
> The LINECOUNT instruction must be executed immediately after the sensor detects parts.

## Procedure 4-2 Creation and confirmation of Tracking Program (TRACK)

### Conditions

- The trigger value can be set correctly according to step 4-1.
- The encoder is enabled and simulation is turned off.

### Steps

1. Create a new program "TRACK". At this time, the tracking program details screen is displayed. (Program list screen → F2"CREATE" → Enter program name → F2"DETAIL" → F3"NEXT")

```
        Line Track Program Header     1/3

      1 Line track schedule number: 0
      2 Continue track at prog end: TRUE
      3 Selected Boundary         : 0

     END      PREV     NEXT
```

> **NOTE**
> The number of tracking program groups must be one. Please enable the group mask setting only for a group set in the schedule.

2. For “Line track schedule number”, enter a schedule number used by this program.
3. For "Continue track at prog end", select whether to continue the tracking motion even after the program ends.

> **NOTE**
> If "Continue track at prog end" is FALSE, when the tracking program ends and returns to the main program, the last instruction of the tracking program moves like CNT0 even if it is CNT100.

4. For Selected Boundary, enter the Boundary number used by this tracking program. If "Selected Boundary" is 0, the tracking area of the number written in "Selected Boundary Set" on the setting screen of schedule will be used by this program.

> **NOTE**
> When the tracking program is running, the "Selected Boundary Set" on the schedule setting screen is automatically changed to the number written on the program details screen. If $LNCFG.$RSTR_BNDS is FALSE, it will not be restored.

> **NOTE**
> Boundary position checking is enforced during program execution or single-stepping, as determined by the value of $LNSCH[1].$SEL_BOUND within the line track schedule associated with the program. This might cause the robot to pause motionless if a position is not within the selected boundary window. However setting $LNSCH[1].$SEL_BOUND = -1 will disable the boundary checking and facilitate program editing.

5. When the edit screen of the tracking program is opened, a pop-up message of the synchronization process is displayed. If the schedule is already synchronized with the target part by Procedure 4-1, it is not necessary to synchronize again, so select "NO". If the trigger value is incorrect, synchronize as described in Section 4.2.
6. Move the conveyor until the parts are close to the robot, and with the conveyor stopped, teach by referring to Figure 4.1 and Example 4.1 (b). To make it easier to teach, it is okay to move the conveyor and readjust the perch position during teaching. Even if the conveyor rotates in the reverse direction,

there is no problem. When you add a teaching point, the position data automatically corrected according to the current position of the conveyor is recorded.

> **NOTE**
> Cartesian must be selected as teaching positions in tracking programs. Do not use UFRAMES when recording positions in a line tracking program. If you do, the point will remain uninitialized and an error will occur.

> **NOTE**
> Do not use a PAUSE instruction in your TRACK program or in a subprogram that is called by the TRACK program. Doing so could result in unexpected motion when the TRACK program resumes.

**Example 4.1 (b) Tracking Program (TRACK)**

```
1: L  P[2]  500mm/sec FINE    -- Move to Approach Point
2: L  P[3]  500mm/sec FINE    -- Move to Tracking Point
3:   WAIT    5.00(sec)         -- Wait 5 seconds on Tracking Point
4: L  P[2]  500mm/sec FINE    -- Move to Approach Point
[End]
```

7. Move the conveyor a short distance and stop it again. At that time, execute the tracking program from the first line. Make sure that the motion instruction can be executed considering the movement amount of the conveyor.

## Procedure 4-3 Creation and confirmation of Main Program (MAIN)

### Conditions

- TRIGGER and TRACK work correctly according to Procedure 4-1 and Procedure 4-2

### Steps

1. Create a new program “MAIN”.
2. The tracking program details screen does not need to be changed. The tracking schedule number is 0(No tracking) by default.
3. Edit the program by referring to Example 4.1.(c). P[1] is Home point (safe position that serves as the reference position for the robot).

**Example 4.1 (c) Main Program (MAIN)**

```
 1:   LINE[1]  ON          -- Enable The Encoder
 2:
 3:   LBL[1]
 4:
 5:J  P[1]  50%  FINE      -- Move to Home Point
 6:
 7:   CALL TRIGGER         -- Call trigger value update program
 8:   CALL TRACK           -- Call tracking program
 9:
10:   JMP LBL[1]           -- Process the next part
[End]
```

4. Place a part upstream on the conveyor from the trigger switch.
5. Execute MAIN from the first line and start to move the conveyor, and confirm that the robot works as follows. When the robot finishes processing one part, the robot waits for the next part by JMP LBL.
    1) Move to P [1] (Home Point).
    2) Wait until the part is detected.
    3) Move to P [2] (Approach Point).
    4) Move to P [3] (Tracking Point).

5) Wait for 5 seconds while tracking.
6) Move to P [2] (Approach Point).
7) Move to P [1] (Home Point).

**NOTE**
If a tracking program is currently being executed at any override speed, and the conveyor begins to move, the robot motion will immediately stop; and the error "LNTK-041 Encoder is moved in T1 mode" will be displayed.

This completes the steps to create and confirm a typical tracking program.

## 4.2 Tracking program synchronization

If the tracking system synchronizes during program execution, use the LINECOUNT and SETTRIG instructions within the program. When editing a tracking program, if the trigger value is incorrect, the system must synchronize before editing the tracking program. Procedure 4-4 describes a procedure for updating the trigger value without executing SET TRIG when teaching the target position of tracking programs.

**Procedure 4-4 Re-synchronizing the Robot and Conveyor**

### Conditions

- Encoder and tracking schedule settings have already been completed.
- The encoder is enabled and simulation is turned off.

### Steps

1. When an editing screen of the tracking program opens from the program list screen, the following pop-up is displayed.

```
The current part detect
trigger value MAY NOT
be valid.
Resynchronize the
tracking system?

NO          YES
```

2. On the above screen, select YES for resynchronize process. If the current trigger value is correct, you can select NO to skip resynchronize process.
3. The following screen will be displayed. Follow the instructions on the screen to detect a part on the conveyor with the Part Detect Switch.

```
Move a new part past the
part detect switch and
into the robot workspace.
Or Press Enter to cancel.
```

4. The following screen will be displayed.

```
Stop the part when it is
at a convenient location
in the robot workspace.

Press ENTER when done.

          [OK]
```

5. Stop the conveyor when the part is in a teaching position.
6. Press ENTER. Program edit screen is displayed.

The robot and conveyor synchronization is complete. If there is a problem, an error message will be displayed depending on the cause. If a different screen is displayed in step 4-4, refer to the following to solve the problem.

## • While conveyor was moving BACKWARDS

If the conveyor was running in the reverse direction when the trigger was detected, you will see a screen similar to the following.

```
WARNING: A trigger was just
generated while conveyor
was moving BACKWARDS!

  Exit the program and
  re-enter it to re-sync!
            OK
```

## • The encoder was invalid

If the encoder associated with the specified line tracking schedule is not enabled you will see a screen similar to the following.

```
WARNING: The selected
 encoder is currently
  turned OFF!

Enable the encoder then
return and re-sync!
          OK
```

### • The tracking schedule has not been properly initialized.

If one or more of the parameters in the specified line tracking schedule are not set properly, you will see a screen similar to the following.

```
WARNING: Selected tracking
   schedule has not been
   properly initialized.

   Initialize the schedule
   then return and re-sync!
              OK
```

### • Part Detect Not Found

If a part detect is not encountered within a certain time period (approximately two minutes), you will see a screen similar to the following. This error can occur due to a stopped or slow-moving conveyor, or because the digital input is not functioning properly.

```
PART DETECT TIMEOUT!!!
A part detect was not
observed within the time-
out period (120 seconds).
Exit the program and
re-enter it to re-sync!
           OK
```

### • Simulation mode is enabled

When the simulation mode is enabled, the following screen is displayed. Disable simulation mode.

```
   Tracking simulation is
   currently enabled.

   Press <ENTER> to
   simulate a new part
   detect trigger value.
            OK
```

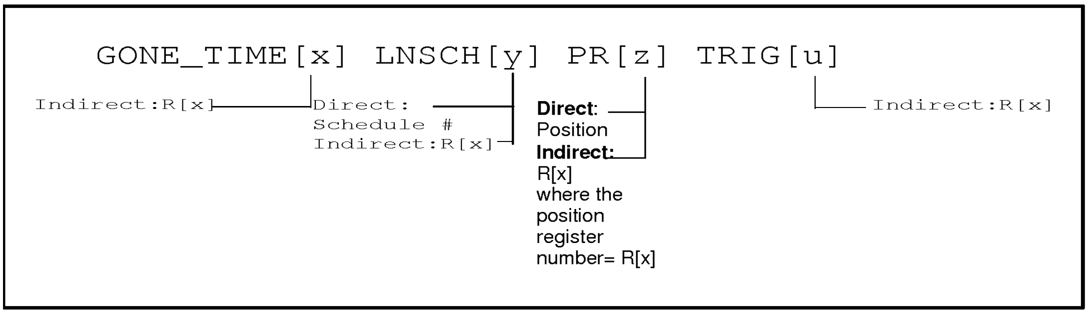
> ⚠**WARNING**
> When you press ENTER on this screen, a count value of the encoder during simulation is set to trigger value in order to execute the program in simulation mode. Program positions should never be taught when using a simulated conveyor. Otherwise, you could injure personnel or damage equipment.

# 4.3 TRACKING INSTRUCTIONS

Tracking commands can be added by pressing F1 [Command] on the program edit screen and selecting "Tracking".

## GONE_TIME

When the GONE_TIME instruction is executed, register x returns the number of seconds before which the position in position register z will be exiting out of the boundary specified in line tracking schedule y with the trigger value stored in register u.




> **NOTE**
> GONE_TIME cannot be used with circular tracking.

> **NOTE**
> When this instruction is called, the conveyor speed at the time will be used for the calculation. If the conveyor speeds up afterward, GONE_TIME might not be accurately estimated. When the conveyor is stopped, GONE_TIME will return a large value instead of an infinite value.

For example,this instruction can be used to monitor whether or not the part will be out of bounds when it is to be picked up. Please refer to Example 4.3(a). Register 12 stores a required time for the “PICK_PART” program.

**Example 4.3 (a) How to use GONE_TIME**

```
GONE_TIME[10] LNSCH[1] PR[1] TRIG[20]
IF R[10]<R[12] JUMP LBL[2]
CALL PICK_PART
LBL[2]:
```

## LINE

The LINE enable instruction enables the encoder for tracking.

```
LINE[x] [value]

Direct    :                          ON  - turns on the encoder
Encoder number (1-8)
Indirect:  R[x]                      OFF - turns off the encoder
Encoder number = contents of R[x]
```

## LINESIM

The LINESIM instruction sets up and enables encoder simulation. The line (encoder) must be enabled to simulate tracking using the LINESIM instruction. The simulation line speed used here is in units of encoder counts per encoder update.

```
LINESIM[x] [value] R[x]

Direct    :               ON
Schedule  number (1-8)    Turns on
Indirect:  R[x]           simulation     Indirect:R[x]
Encoder number = contents OFF            where simulation
of R[x]                   Turns off      line speed =
                          simulation     contents of R[x]
```

## LINECOUNT

The LINECOUNT instruction reports the current tracking encoder count. This instruction must be used immediately after detecting a part trigger, to record the position of the conveyor.

```
LINECOUNT[x] output

Direct :                    Indirect:R[x]
Encoder number (1-8)        where the number of
Indirect: R[x]              encoder counts = contents
Encoder number =            of R[x]
contents of R[x]
```

## LINERATE

The LINERATE instruction reports the tracking encoder rate, in units of encoder counts per encoder update.

```
LINERATE[x] output

Direct:                              Indirect:  R[x]
Encoder number (1-8)                 where the current encoder
Indirect:  R[x]                      rate = contents of R[x]
Encoder number = contents of R[x]
```

## LINESTOP

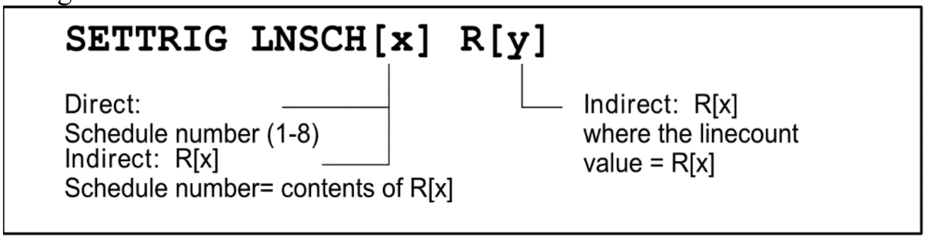
The LINESTOP instruction reports the tracking encoder stopped status, based on the current line rate and encoder stop threshold.

```
LINESTOP[x] output

Direct:                  Indirect:  R[x]
Encoder number (1-8)     indicates whether the encoder is stopped
Indirect:  R[x]          or running; 1 = stopped, 0 = running
Encoder number =
contents of R[x]
```

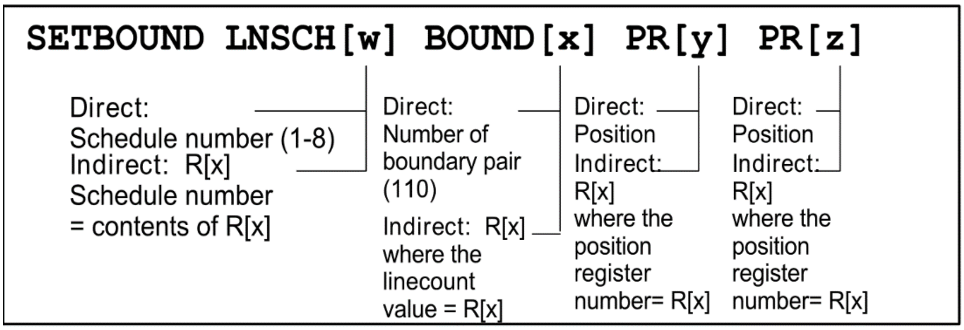
## SETTRIG

The SETTRIG instruction sets the tracking schedule trigger value. The value typically uses the value stored in the register using LINECOUNT instruction.




## SETBOUND

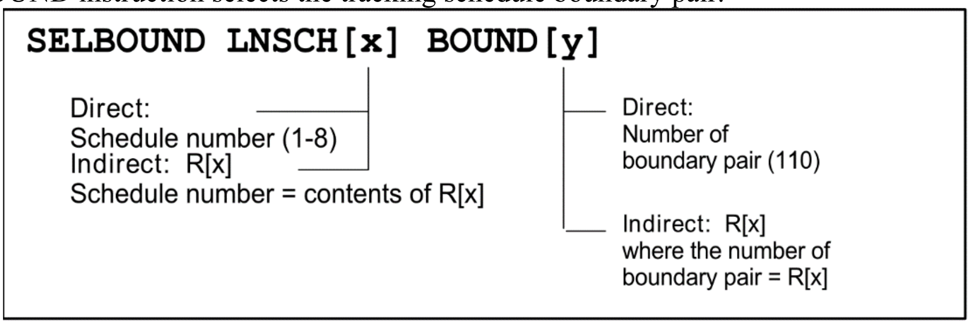
The SETBOUND instruction sets the tracking schedule boundary values, based on the WORLD frame positions stored in the two position registers. Of the two positions, the one on the upstream side is set to Up, and the one on the downstream side is set to Dn.




> **NOTE**
> GONE_TIME cannot be used with circular tracking.

## SELBOUND

The SELBOUND instruction selects the tracking schedule boundary pair.




## ACCUTRIG LNSCH

The ACCUTRIG LNSCH is an instruction to set a more accurate trigger value using LINECOUNT. This instruction activates an interrupt routine to set a system time when I/O is triggered. The first LINECOUNT instruction after that uses this system time to retrieve the encoder count at the system time.

In order to use the ACCUTRIG instruction, the following system variable must be set:

$LNCFG.$SLC_PT_TRIG=TRUE

You must turn the controller off then on again for this variable to take effect. If ACCUTRIG is not being used, then this variable should be set to FALSE.

**ACCUTRIG LNSCH[x]**

Direct:
Schedule number (1-8)
Indirect: R[x]
Schedule number= contents of R[x]

The ACCUTRIG LNSCH instruction should be used in the program before the program waits for the digital input.

**Example 4.3 (b) How to use ACCUTRIG LNSCH**

```
1:ACCUTRIG LNSCH[1]
2:WAIT DI[1]=OFF
3:WAIT DI[1]=ON
4:LINECOUNT[1] R[1]
5:SETTRIG LNSCH[1] R[1]
```

## TRKUFRAME

The TRKUFRAME instruction sets the tracking user frame in the schedule to the value in the position register.

**TRKUFRAME LNSCH[x] PR[y]**

LNSCH[x]:
Direct:
Schedule number (1-8)
Indirect: R[x]
Schedule number= contents of R[x]

PR[y]:
Direct: Tracking User frame in position
Indirect: R[x]
where the position register number = R[x]

## VISUFRAME

The VISUFRAME instruction sets the tracking user frame according to the value in the position register. The position value in the position register is the tracking user frame with respect to the robot world frame.

**VISUFRAME LNSCH[x] PR[y]**

LNSCH[x]:
Direct:
Schedule number (1-8)
Indirect: R[x]
Schedule number= contents of R[x]

PR[y]:
Direct: Tracking User frame in position with respect to world frame
Indirect: R[x]
where the position register number = R[x]

## STOP_TRACKING

The STOP_TRACKING instruction is used inside a tracking program to end tracking motion temporarily. The robot will remain stopped until the program execution reaches the next tracking motion and the destination of that motion enters the boundary.

**Example 4.3 (c) Track while waiting**

```
1:L P[1] 1000mm/sec CNT100
2:WAIT DI[10] = ON
3:L P[2] 1000mm/sec CNT100
```

In the above program, once the robot reaches P[1] it will continue to follow P[1] as it moves with the conveyor until the WAIT condition is satisfied and P[2] enters the boundary. After P[2] enters the boundary, the robot will begin moving to P[2]. If this program is modified as follows:

**Example 4.3 (d) Do not track while waiting**

```
1:L P[1] 1000mm/sec CNT100
2:STOP_TRACKING
3:WAIT DI[10] = ON
4:L P[2] 1000mm/sec CNT100
```

In this version of the program, once the robot reaches P[1] it will stop and remained stopped. The robot will not begin moving again until after the WAIT condition is satisfied and P[2] enters the boundary. Only then will the robot begin moving towards P[2].

> **NOTE**
> The STOP_TRACKING feature is not available for use with Circular tracking. When the STOP_TRACKING instruction is specified in a Circular tracking program, the instruction will do nothing as if the instruction is not there.

## DEFENC

The DEFENC instruction defines the current tracking encoder number. It copies the contents of the specified $LNSNRSCH structure, into the specified $ENC_STAT structure, to configure the encoder parameters.

```
        DEFENC[x]  SNRSCH[x]

Direct:                               Direct:
Encoder number (1-8)                  Schedule number (1-8)
Indirect:R[x]                         Indirect:R[x]
Encoder number = contents of R[x]     Schedule number = contents of R[x]
```

# 5 ADVANCED TECHNIQUES

## 5.1 MULTIPLE BOUNDARY POSITIONS EXAMPLE

This example examines the task of painting a car body. The task will be broken down into three zones or windows within the robot workspace. In this context a boundary set or pair describes the edges of each workspace zone. Refer to Fig. 5.1.

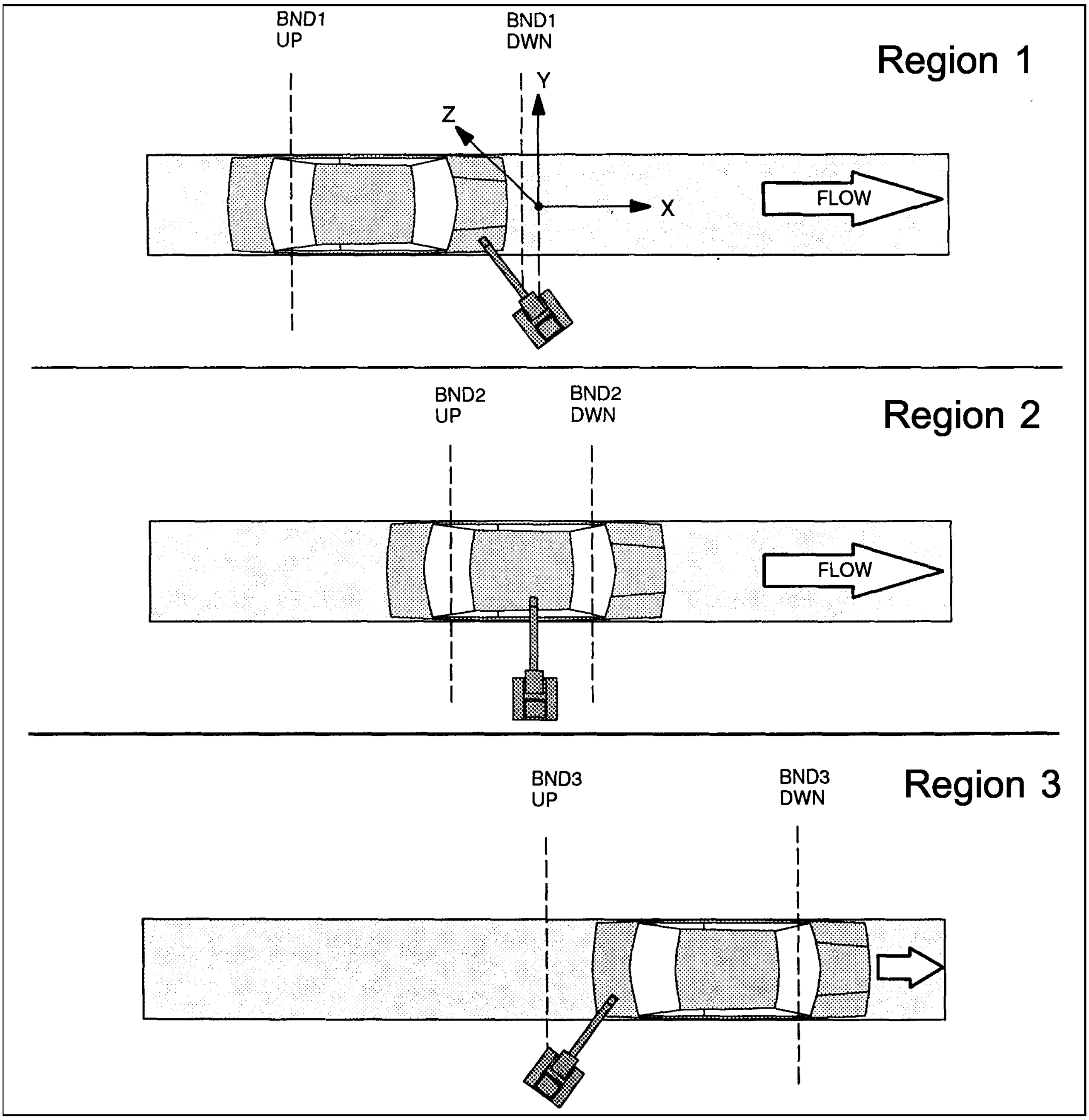



**Fig. 5.1 Tracking boundary positions**

In this example the car will be painted in three regions:

- The front hood
- The top

- The back deck

In Fig. 5.1, the regions labeled 1, 2, and 3, are the boundaries of three work zones. Here UP is the up-stream boundary and DWN is the down-stream boundary where up and down stream refers to the direction of conveyor motion in terms of conveyor flow.

The first region of the car is painted using boundary set 1, the second using boundary set 2, the third using boundary set 3. In each case the line track system will not issue robot motion until the current position is within the selected zone. This ensures that the robot will always be able to reach the program position, even though the position is moving to track the conveyor.

The teach pendant program in Example 5.1 is an example of a line track job that calls the programs to paint the regions of the car shown in Fig. 5.1.

**Example 5.1. Job that calls programs to paint a car**

```
/PROG  PAINT CAR
/MN
  1: LINE[1] ON ;
  2: LBL[1] ;
  3: CALL HOME ;                        -- call rest pos. (non-tracking)
  4: WAIT DI[32]=ON ;                   -- wait for part detect
  5: LINECOUNT[1] R[1] ;
  6: SETTRIG LNSCH[1] R[1] ;
  7: SELBOUND LNSCH[1] BOUND[1] ;       -- select zone 1
  8: CALL CAR_FRNT ;                    -- paint front (tracking)
  9: SELBOUND LNSCH[1] BOUND[2] ;       -- select zone 2
 10: CALL CAR_TOP ;                     -- paint top (tracking)
 11: SELBOUND LNSCH[1] BOUND[3] ;       -- select zone 3
 12: CALL CAR_BACK ;                    -- paint back (tracking)
 13: JMP LBL[1] ;
/POS
/END
```

In this example, the issue of continuous tracking must also be taken into consideration. The tracking motion programs CAR_FRNT and CAR_TOP should have their associated program header data values set for CONTINUOUS TRACKING = TRUE, while CAR_BACK should have CONTINUOUS TRACKING = FALSE.

This allows the robot to continue to track the conveyor between the various tracking motion programs, but will stop the robot from tracking the conveyor upon completion of the last tracking program, before it returns to the rest position.

# 5.2 SKEW BOUNDARY

Sometimes there may be an obstacle that robot has to avoid. As a result, a skew (angular) boundary may be needed. See Fig. 5.2(a)

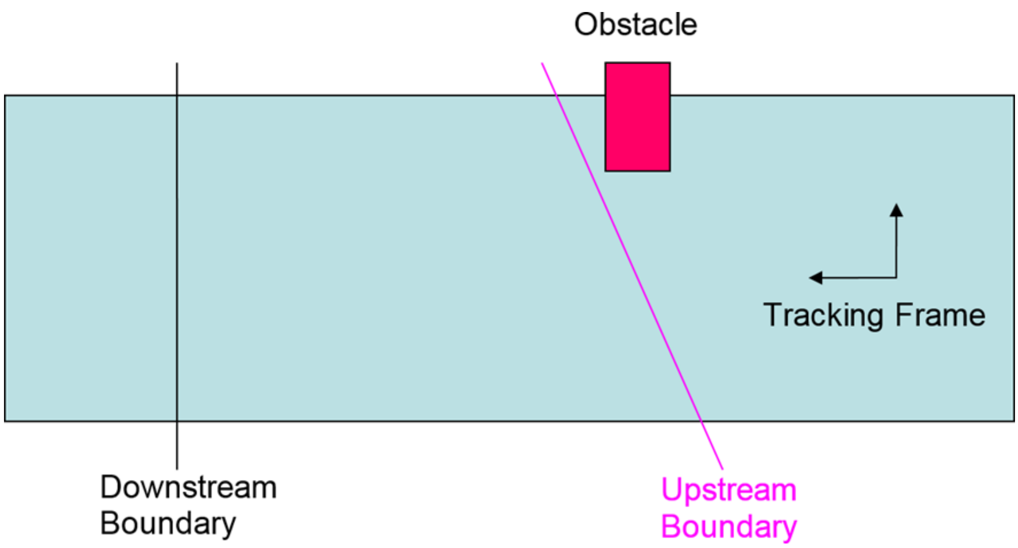



**Fig. 5.2(a) Skew Boundary**

> **NOTE**
> Skew boundary can only use line tracking. It cannot be used for circular tracking and rail tracking, and the set value is ignored.

> **NOTE**
> Skew boundary cannot be used for Paint Line tracking.

The unit is degrees. The default value is 0. Maxmimum: 75. Minimum: -75. A plus-value angle means that the boundary is rotated clockwise, pivoting at the intersection of the tracking frame X axis and the boundary plane. A minus-value angle means that the boundary is rotated counterclockwise.

For example, the upstream boundary and downstream boundary become as shown in Fig. 5.2 (b) in the case that:

| | | | |
|---|---|---|---|
| **Upstream** | **: 500** | **Skew** | **: 30** |
| **Downstream** | **: 1000** | **Skew** | **: -45** |

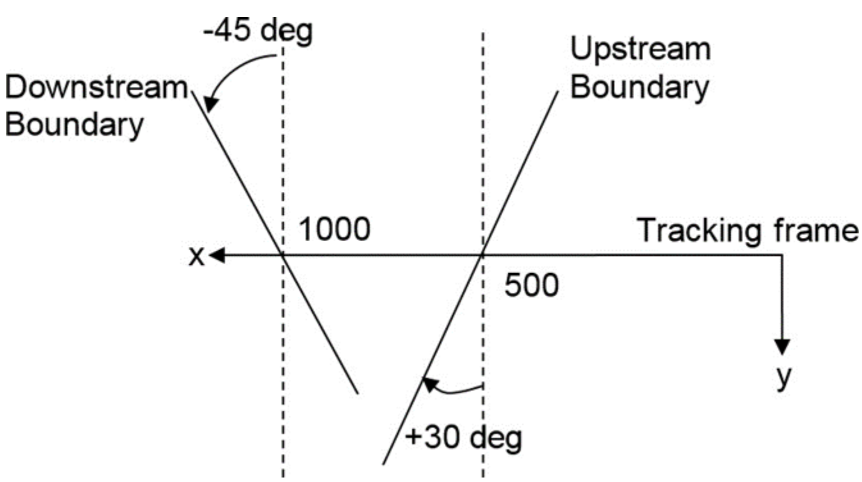


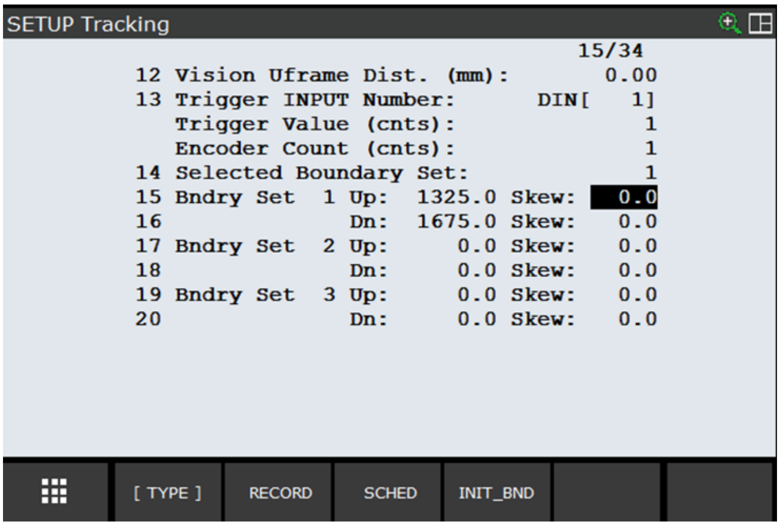

**Fig. 5.2(b) Skew Boundaries Illustrated**

> **NOTE**
> When Skew is set, there is a position where the upstream boundary is on the downstream side of the downstream boundary. An alarm will be displayed if the robot moves to that position.

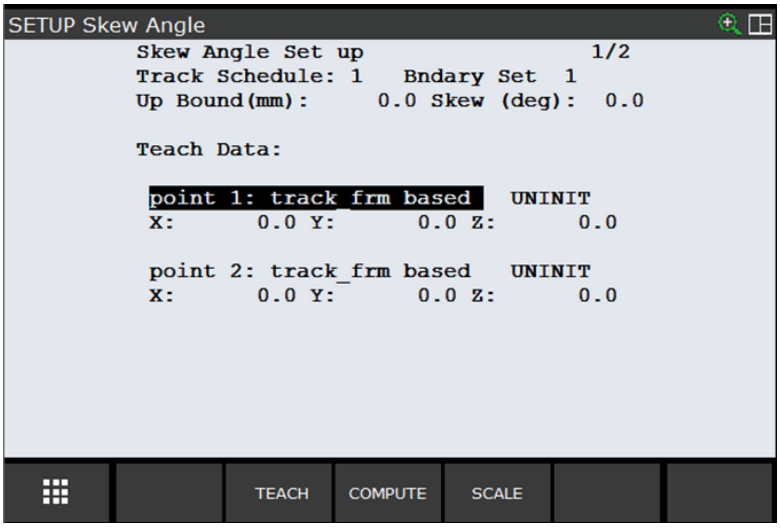
## Procedure 5-1 Setting up Skew Boundary

## Steps

1. On the tracking schedule setting screen, move the cursor to "Skew". At this time, you can also set the value by directly entering it.




2. If you press **SHIFT** and **F2, RECORD** with your cursor in the Skew angle field, the following menu will be displayed.




3. Once in this screen, you use SHIFT and F2, TEACH, to record the 2 points that define the skew angle. See Fig. 5.2 (c).

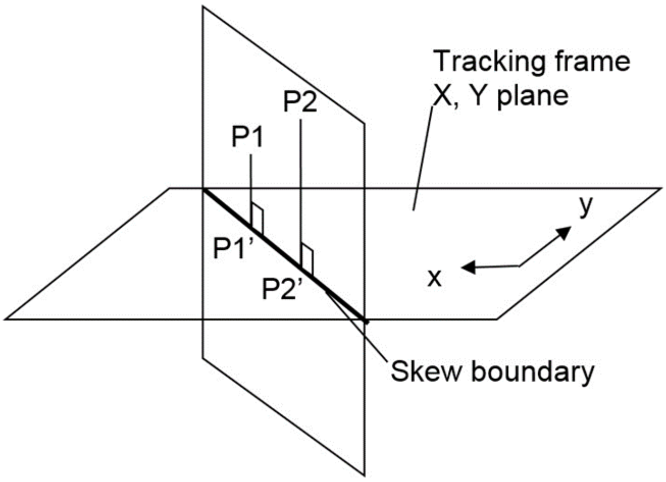



**Fig. 5.2(c) Teaching Skew Angle Points**

4. Press F3, COMPUTE, to update the angle value to pivot the boundary point.
5. Press the PREV key to return to the Line Tracking Setup Menu.

> **NOTE**
> The system will not remember the original Point1 and Point2 value. Every time this sub menu is entered, Point1 and Point 2 will be reinitialized.

> **NOTE**
> SETBOUND instruction is used to determine the upstream and downstream boundaries from PR. When this instruction is executed, the boundary skew is changed to 0 regardless of the previous setting.

## 5.3 CIRCULAR BOUNDARY

For a pick/place application, however, part flow into a boundary may not always be inside the robot's operating space. Therefore, as shown in Figure 5.3 (a), the diamond-shaped part has crossed the upstream boundary but has not yet entered the operating space of the robot. In this case, the robot will try to pick the part, but will be unable to do so because it will fault with a position not reachable error.

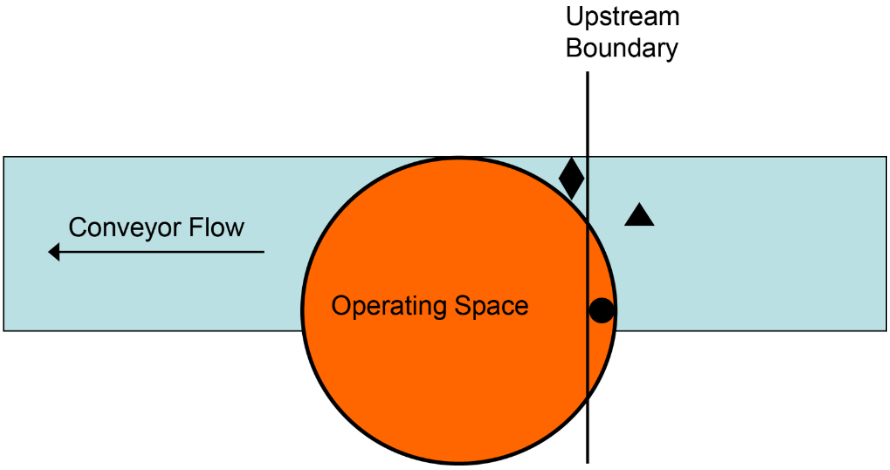



**Fig. 5.3(a) Parts Entering the Operating Space of a Robot**

When circular boundary is used for the robot's range of motion, so that the robot will only pick the part when it is within this circular boundary and has passed the upstream boundary as shown by the hatched tracking area in Fig. 5.3 (b). The robot will wait even when the part passes the user specified upstream boundary but has not yet entered this circular boundary (diamond-shaped part).

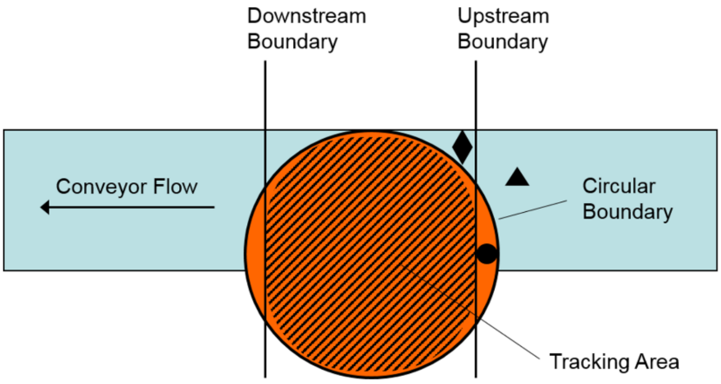



**Fig. 5.3(b) Circular Boundary**

> **NOTE**
> Gone Time will compare the gone time for the user specified boundary, with the gone time for the circular boundary, and provide the shortest time.

Circular boundary can be used for circular tracking as well. The robot can track only in the hatched tracking area shown in Fig. 5.3(c). In this Figure, robot can only move for circular part.

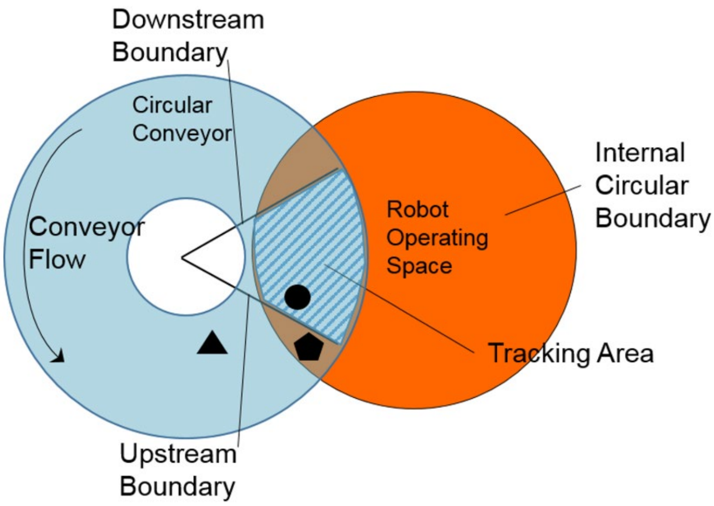



**Fig. 5.3(c) Circular Boundary with Circular Conveyor**

## 5.3.1 Setting a Circular Boundary

The following variables support the internal circular boundary.

- $LNCFG_GRP[group].$WRK_RAD : Radius of circular boundary (mm)
- $LNCFG_GRP[group].$WRK_RAD_ENB : Circular boundary is enabled.

**NOTE**
The center of circular boundary is the world frame origin. At this time, it is necessary to set the length shorter than the maximum reachable distance so that the movement path does not exceed the motion range.

## 5.3.2 Alarms for Using Circular Boundary

This section describes whether tracking motion instructions can be executed when using a circular boundary.

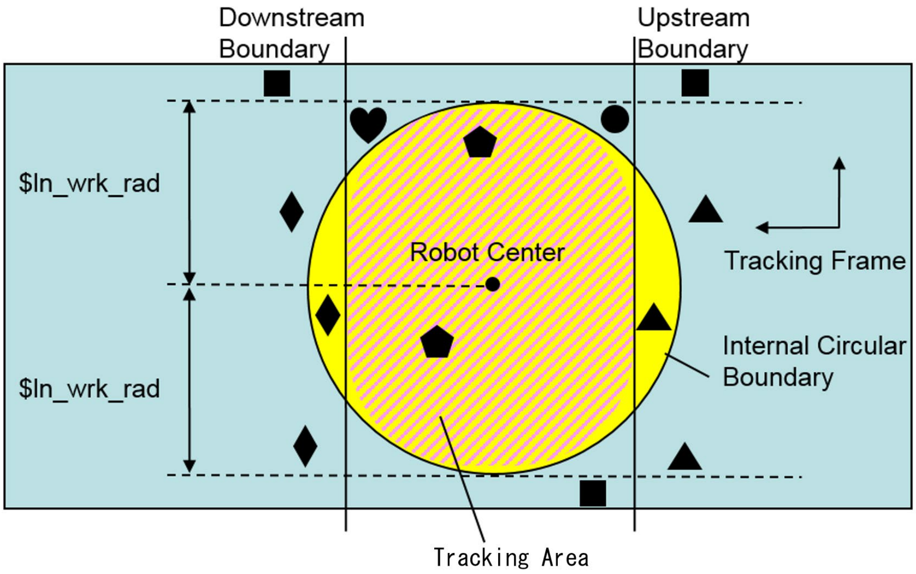



**Fig. 5.3.1(a) Circular Boundary with Circular Conveyor**

When parts are downstream of the user-defined downstream boundary (diamond-shaped parts), the system will post a Track destination gone error if the robot tries to pick/place these parts.
When parts are upstream of the user defined downstream boundary, but downstream of the internal circular boundary (heart-shaped part), the system will post a Track destination gone error if the robot tries to pick/place these parts.
When the part will never enter the internal circular boundary (square parts), the system will post a Track destination gone error if the robot tries to pick/place these parts.

System will also post a Track destination gone error when the part will pass through the internal circular boundary before user-defined upstream boundary (dot X in Fig. 5.3.1 (b)) if the robot tries to pick/place these parts.

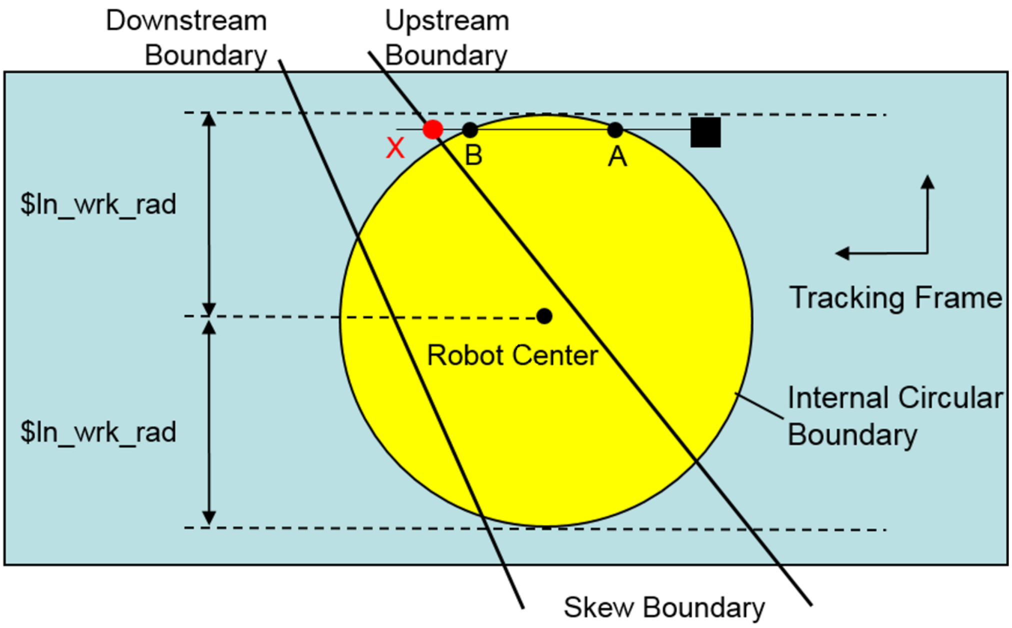



**Fig. 5.3.1(b) Part Exits Circular Boundary before Skewed Upstream Boundary**

## 5.3.3 Restrictions

- Rail tracking does not support circular boundary.
- Paint Line tracking does not support circular boundary.
- When the circular tracking frame is not parallel to the robot world Z (within 12 degrees), a warning "LNTK-75 Circular Boundary not used", will be posted when calling or running a circular tracking program.

# 5.4 TRACKING PART QUEUES

In a system where parts flow one after another on a conveyor, the next part passes in front of the sensor between the time when the sensor detects one part and the time when the robot completes the process and waits for detection again. This may be a problem. You can prevent this problem by using a queue to get the trigger value. It is necessary to create a monitor program that temporarily stores the trigger value by using multiple consecutive registers as a queue. This program runs on other tasks.

> **NOTE**
> Group mask of the monitor program must be set to ' * '. If you turn on "Ignore pause" of the monitor program, the system can continue to queue the trigger value even if the system is paused.

Create a monitor program as shown in Example 5.4.2 (a) and execute it in another task using RUN as shown in Example 5.4.2 (b). The monitor program gets the encoder count, but does not immediately set it in the tracking schedule. The count value is queued until the robot completes the previous process.
When the robot starts processing the next part, the oldest trigger value is set from the queue to the tracking schedule so that the robot can process all parts.

> **NOTE**
> PICK is a tracking program that picks up the parts, and PLACE is the non-tracking program that places the parts. These need to be created appropriately. Refer to Procedure 4-2 for creating a tracking program.

> **NOTE**
> Example 5.4.2 (a) can store the trigger values for 10 parts. If the number of unprocessed parts on the downstream side of the trigger switch is larger than the number that can be stored, you need to increase the number of registers used as queues.

**Example 5.4.2 (a) Monitor Program (MONITOR)**

```
LBL[1]                     -- Main Parts Detect Loop

WAIT DI[1]=ON+             -- Wait for Parts Detect
LINECOUNT[1] R[R[1]]       -- Save Trigger Value to Queue

R[2]=R[1]+1                -- Next Index Register
IF R[2]<=20,JMP LBL[2]     -- R[11]-R[20] are Used for MONITOR
R[2]=11
LBL[2]

WAIT R[2]<>R[3]            -- All Registers are Stored
R[1]=R[2]                  -- Update Index
JMP LBL[1]                 -- Go to Main Loop Start
[End]
```

**Example 5.4.2 (b) Main Program**

```
LINE[1] ON
R[1]=11             -- Initialize Index Register to Save Trigger Value
R[3]=11             -- Initialize Index Register to Use Trigger Value
RUN MONITOR         -- Run Monitor Program in Another Task

LBL[1]                        -- Main Loop

J P[1] 100% CNT100            -- Move to Stationary Position
WAIT R[1]<>R[3]               -- Wait for New Part Trigger
SETTRIG LNSCH[1] R[R[3]]      -- Set Next Trigger Value

CALL PICK                     -- Tracking Program
CALL PLACE                    -- Non-Tracking Program

R[3]=R[3]+1                   -- Increase Index Register
IF R[3]<=20,JMP LBL[1]        -- Use R[11]-R[20]
R[3]=11
JMP LBL[1]                    -- Go to Main Loop Start
[End]
```

# 5.5 MULTIPLE CONVEYORS (DUAL LINE TRACKING)

The typical tracking environment involves manipulating a part moving, through the workspace, on a single conveyor. However, if you have a situation that requires moving a part from one conveyor to another, this can also be handled using line tracking, but requires the following special considerations for all conveyors.

- Settings each encoder
- Setting each tracking schedule
- Creating each tracking program
- Creating each monitor program

The following example shows a typical dual line tracking program. This program envisions a system in which a robot picks up one part that moves on conveyor 1 and puts it in a box that moves on conveyor 2.If the number of unprocessed parts downstream from the trigger switch is greater than can be stored, the system needs to increase the number of registers used as queues.

> **NOTE**
> Group mask of the monitor program must be set to ' * '. If you turn on "Ignore pause" of the monitor program, the system can continue to queue the trigger value even if the system is paused.

> **NOTE**
> PICK is a tracking program that picks up the parts, and PLACE is the non-tracking program that places the parts. These need to be created appropriately. Refer to Procedure 4-2 for creating a tracking program. After creating a new tracking program, make sure that each tracking program can track the conveyor correctly.

**Example 5.5.2 (a) Monitor Program (MONITOR 1)**

```
LBL[1]                      -- Main Parts Detect Loop
WAIT DI[1]=ON+              -- Wait for Part Detect
LINECOUNT[1] R[R[1]]        -- Save Trigger Value to Queue
R[2]=R[1]+1                 -- Next Index Register
IF R[2]<=20,JMP LBL[2]      -- R[11]-R[20] are Used for MONITOR1
R[2]=11
LBL[2]
WAIT R[2]<>R[3]             -- All Registers are Stored
R[1]=R[2]                   -- Update Index
JMP LBL[1]                  -- Go to Main Loop Start
[End]
```

**Example 5.5.2 (b) Monitor Program (MONITOR 2)**

```
LBL[1]                      -- Main boxes Detect Loop
WAIT DI[2]=ON+              -- Wait for boxes Detect
LINECOUNT[2] R[R[4]]        -- Save Trigger Value to Queue
R[5]=R[4]+1                 -- Next Index Register
IF R[5]<=30,JMP LBL[2]      -- R[21]-R[30] are Used for MONITOR2
R[5]=21
LBL[2]
WAIT R[5]<>R[6]             -- All Registers are Stored
R[4]=R[5]                   -- Update Index
JMP LBL[1]                  -- Go to Main Loop Start
[End]
```

**Example 5.5.2 (c) Main Program**

```
  LINE[1] ON
  LINE[2] ON
  R[1]=11      --Index Register for Saving Trigger Values of MONITOR1
  R[3]=11      --Index Register for Using Trigger Values of MONITOR1
  R[4]=21      --Index Register for Saving Trigger Values of MONITOR2
  R[6]=21      --Index Register for Using Trigger Values of MONITOR2

  RUN MONITOR1        -- Run MONITOR1 in Another Task
  RUN MONITOR2        -- Run MONITOR2 in Another Task

  LBL[1]

J P[1] 100% CNT100           -- Move to Stationary Position
  WAIT R[1]<>R[3]            -- Wait for New Part Trigger
  SETTRIG LNSCH[1] R[R[3]]   -- Set Next Trigger Value to Schedule 1
  CALL PICK                  -- PICK is Tracking Program

  R[3]=R[3]+1
  IF R[3]<=20,JMP LBL[2]     -- R[11]-R[20] are Used for MONITOR1
  R[3]=11
  LBL[2]

J P[2] 100% CNT100           -- Move to Stationary Position
  WAIT R[4]<>R[6]            -- Wait for New Box Trigger
  SETTRIG LNSCH[2] R[R[6]]   -- Set Next Trigger Value to Schedule 2
  CALL PLACE                 -- PLASE is Tracking Program

  R[6]=R[6]+1
  IF R[6]<=30,JMP LBL[1]     -- R[21]-R[30] are Used for MONITOR2
  R[6]=21
  JMP LBL[1]
[End]
```

# 5.6 FINE TUNING TRACKING ACCURACY

If you want to fine tune or make the robot move closer to the specified position during line tracking, you can use one or both of the following methods:

- Static Tuning Variable (I/O Delay)
- Dynamic Tuning Variable (Servo Delay)

To fine-tune these parameters, an actual part could be used to set the trigger, or another method for setting the triggerat your chosen point on the conveyor could be used.

> **NOTE**
> You must first complete I/O Delay, afterward you can complete Servo Delay.

> **NOTE**
> For visual tracking systems, adjust the teaching position without changing the I/O Delay.

## 5.6.1 Static Tuning Variable

The first method used to fine-tune line tracking accuracy is to adjust the I/O Delay. I/O Delay is roughly the time it takes for the signal from the part detect to reach the controller. Use the following method to fine-tune $ENC_IO_DLYF[encoder number].

> **NOTE**
> For R-30iB and earlier, use $ENC_IODELAY[encoder number] instead of $ENC_IO_DLYF[encoder number]. At this time, $ENC_IOD_ENB[encoder number] must be set to TRUE.

1. Run a chosen point on the conveyor past the part detect at a very slow speed (~10-30 mm/sec) and set the trigger.
2. After the part detect is complete, run the conveyor at any speed until your chosen point on theconveyor, at which the trigger was set, is in the robot working area.
3. Create a new tracking program and teach a position, P[1]; the desired target point on the conveyor.
4. Using the WAIT instruction, add a long wait to the program, or wait for some user input (Flag or Register). For example your program might look something like this:

```
L P[1] 500 mm/sec FINE
WAIT 30.00 (sec)
```

5. After creating the program, the next step is to move the conveyor at the maximum speed, detect the part, and set the trigger again.
6. After triggering again, move the conveyor at any speed until the part is close to the robot.
7. Run the program with the conveyor stopped.
8. Verify that the robot is lined up at the correct position, P[1]. If the robot is not lined up with P[1], then modify $ENC_IO_DLYF[encoder number] in milliseconds as shown in the following figure:

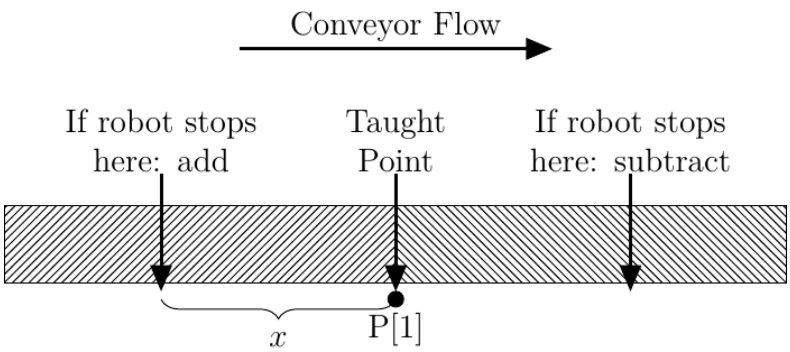



**Fig. 5.6.1 $ENC_IO_DLYF[ ] Variable Adjustment**

For example, if the robot lags by x = 2mm, and the conveyor speed is 750mm/s, then:

$$\frac{2\text{mm}}{750\ \text{mm/s}} \approx 2.7\,ms$$

In this case, add 2.7 to $ENC_IO_DLYF[encoder number].

9. Repeat Step 5 through Step 8 until the robot lines up with P[1].

## 5.6.2 Dynamic Tuning Variable

Servo delay is roughly the communication delay between the conveyor and the robot. This can only be seen when the conveyor is running. Before modifying the Servo Delay, it is necessary to set up the I/O Delay first (See section above).

> **NOTE**
> For servo conveyor tracking, use $SLTK_GRP[group].$SRVO_DELAY instead of $LNCFG_GRP[group].$SRVO_DELAYF.

> **NOTE**
> For R-30iB and earlier, the total value of the following two variables is used as the Servo delay value.
> • $LNCFG_GRP [group].$SRVO_DELAY
> • $LNCFG_GRP [group].$SOFT_DELAY

To fine-tune the Servo Delay, use the following method:

1. Run the same program as used for I/O Delay at maximum production speed and DO NOT STOP the conveyor.
2. Verify Section 6.4, FINE TUNING TRACKING ACCURACY that the robot is lined up at the correct position while the conveyor is moving and the program is running. If the robot is not lined up with P[1], modify $LNCFG_GRP[group].$SRVO_DELAYF in milliseconds, as shown in the following figure:

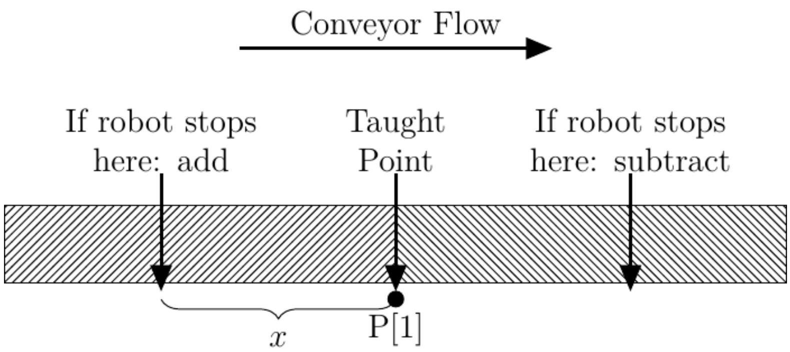



**Fig. 5.4.2 $LNCFG_GRP[ ].$SRVO_DELAYF Variable Adjustment**

For example, if the robot lags $x$ = 1 mm, and the conveyor speed is 750 mm/s, then:

$$\frac{1\text{mm}}{750\ \text{mm/s}} \approx 1.3ms$$

In this case, add 1.3 to $LNCFG_GRP[group].$SRVO_DELAYF.

3. Repeat Step 1 and Step 2, observing the robot accuracy while the WAIT is executed, until the desired accuracy is reached.

# 5.7 TRACKING USER FRAME

## 5.7.1 Overview

A Tracking User frame is used to compensate for part location or orientation changes which is detected by one-dimensional or two-dimensional position sensor.
In general,
- Use one-dimensional position sensor to detect and compensate the translational offset (ΔY) in the direction perpendicular to the conveyor.

- Use two-dimensional position sensor (camera) to detect and compensate the translational offset (ΔX and ΔY) and rotational offset (ΔR).

> **NOTE**
> If you have FANUC vision sensor, don't use Tracking User Frame feature described in this section, but use *i*RVision Visual Tracking feature.
> If you use one-dimensional position sensor to compensate an offset in one direction, you can use Offset instruction or Tool Offset instruction as a substitute for Tracking User Frame.

If Tracking User Frame is used, position data is taught on User/Part Frame which is on the tracked object (part). You only need to shift User/Part Frame according to the offset value of each part so that the robot can approach to the same position of the part.

Use can choose one from the following two types of User/Part Frame definition.

- TRKUFRAME
  User/Part Frame is defined with respect to Nominal Frame.
  Choose this one if you use one or two dimensional position sensor to detect position offset of the part (relative to the "standard part").
  In a TP program, use TRKUFRAME instruction.
- VISUFRAME
  User/Part Frame is defined with respect to World Frame.
  Choose this one if you use two dimensional position sensor to detect the position of the part with respect to World Frame.
  In a TP program, use VISUFRAME instruction.

To use the Tracking User Frame functions, in addition to setting up Line Tracking, you must also set up two other items in Tracking Schedule Setup.

- Set Use Tracking Uframe to YES (default is NO).
- Set Vision Uframe Dist to a proper value (default is zero) in order to use VISUFRAME.

For more information about Tracking Setup, refer to the Tracking Setup section of this manual.

> **NOTE**
> Only one of the two Tracking User frame types, TRKUFRAME or VISUFRAME, can be used in a single tracking schedule.

> **NOTE**
> Circular Tracking does not support tracking user frame

## 5.7.2 Tracking Frame Terminology

In order to use the Tracking User frame functionality, you must understand Line Tracking frames and their relationship. See Fig. 5.7.2 for a typical robot-conveyor setup and frame relationship.

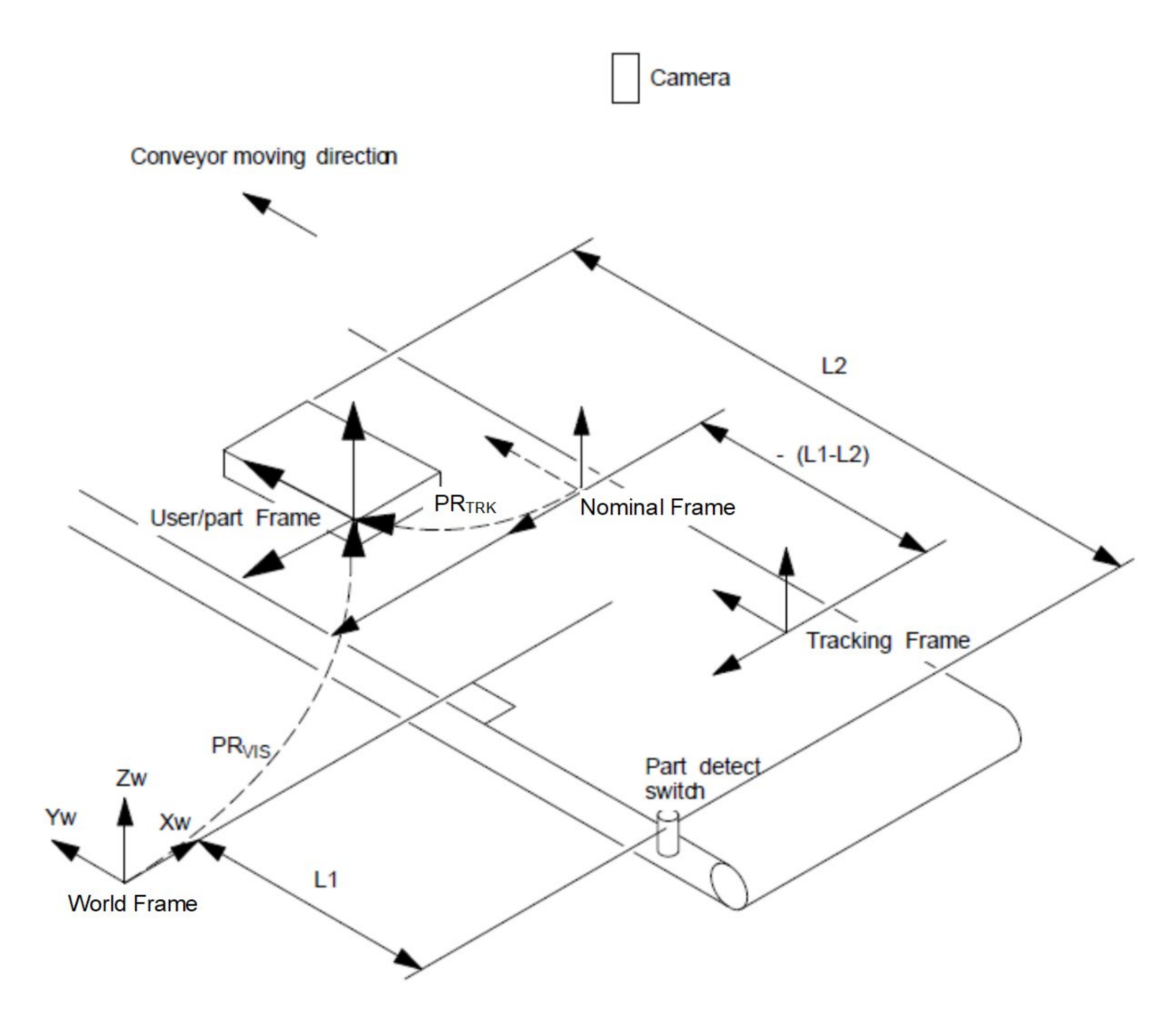



**Fig. 5.7.2 User/Part frame and frame relationship in typical line tracking system**

The following terminology must be understood:

**Table 5.7.2 Terminology of Tracking User Frame**

| Terminology | Description |
|---|---|
| Tracking Frame | A nominal tracking frame defined in Tracking Schedule Setup. It is stationary during the line tracking motion. Refer to the Line Tracking Setup and Operations Manual for more information. |
| Nominal Frame | A runtime nominal tracking frame. It is parallel to the tracking frame and moves with the conveyor while tracking. (If Tracking User Frame is disabled, the position data in tracking program is recorded with respect to this frame.) |
| User/Part Frame | A frame on the tracked object (part) defined by the user. If compensate rotational offset, you must set the origin of this frame to the sensor target position on the part. |
| World Frame | The standard robot world frame. |
| $PR_{TRK}$ | A user/part frame with respect to the Nominal frame in position form. This value varies for each part. A position register is used to store this data. |
| $PR_{VIS}$ | A user/part frame with respect to the robot world frame in position form. This value varies for each part. A position register is used to store this data. This frame is used only for VISUFRAME. |

| Terminology | Description |
|---|---|
| L1 | Part DetectDist.(mm) in the Line Tracking Schedule Setup. If you set this parameter to the distance from the robot to the part detect switch, L1 is as depicted in Fig. 5.7.2. |
| L2 | Travel distance in which the part is past the part detect switch. It can be calculated using the following formula: $L_2 = \dfrac{\text{Encoder Count - Trigger Value}}{\text{Enc Scale Factor}}$ |

## 5.7.3 Setup for TRKUFRAME

If use TRKUFRAME, you need to compute $PR_{TRK}$ (user/part frame with respect to the nominal frame) of each part, and store it to a position register.
Let's think about a typical system as follows;

- The conveyor is placed as in Fig.5.7.2. That is, the orientation of the tracking frame is the same as rotating the world frame by 90 degrees around the z axis. (In other words, the conveyor direction is parallel to World +Y and the conveyor surface is vertical to World +Z.)
- Two-dimensional position sensor detects a position offset of the part relative to the standard part. The offset consists of translational offset ($\Delta X, \Delta Y$) and rotational offset ($\Delta R$). (The offset value is represented with respect to the nominal frame.)

If you compute $PR_{TRK}$ of the standard part in advance, you will only need to add the offset to it to obtain $PR_{TRK}$ of each part. (See Example 5.7.5(a))
Procedure 5-2 can be used to calculate $PR_{TRK}$ of the standard part.

### Procedure 5-2 Calculate $PR_{TRK}$ of the standard part for TRKUFRAME

### Steps

1. Pass the standard part through the part detects switch to detect trigger, then stop the conveyor at a position where the robot can reach the part easily.
2. Look at the Tracking Schedule Setup screen to observe the Encoder Count, Trigger Value, Enc Scale Factor, Part Detect Dist., and Tracking frame values.
3. Calculate $L_2$ using the equation in Table 5.7.2.
4. Calculate $L_1 - L_2$
5. Add $-(L_1 - L_2)$ to the Y component of the origin of the Tracking frame to shift the Tracking frame by $-(L_1 - L_2)$ along the Tracking frame x direction. (Note that this calculation is based on the condition that the orientation of the tracking frame is the same as rotating the world frame by 90 degrees around the z axis. If this condition is not met, modify this calculation properly.)
   The resulting frame is the Nominal frame at the current part location (note that conveyor is stopped now).
6. Choose a User frame on the object (part). If you will compensate only the translational offset ($\Delta Y$), you can choose the origin of User frame at anywhere you like. On the other hand, if you will compensate also the rotational offset ($\Delta R$), you must choose the origin of User frame at the same position of the sensor target on the part.
   Move robot tool center point (TCP) to the origin of the User frame you have chosen. Then memorize (write down) the position of TCP in robot world coordinates.
7. Subtract the nominal frame's corresponding component from the User frame's to get $PR_{TRK}$ (User/part frame with respect to the Nominal frame). $PR_{TRK}$ can be calculated by the following expression. User frame= u and nominal frame= n
   $PR_{TRK}(X,Y,Z,W,P,R) = \{Yu - Yn, -(Xu - Xn), Zu - Zn, 0, 0, 0\}$
   NOTE:
   - The calculation of X and Y is based on the condition that the conveyor direction is parallel to World +Y. If this condition is not met, modify this calculation properly.

- That W and P are both zero is based on the condition that the conveyor surface is vertical to World +Z. In this case, R can be chosen arbitrarily so zero is chosen in the expression above.

> **NOTE**
> If the conveyor surface is not vertical to World +Z, we cannot calculate $PR_{TRK}$ by simple subtraction as described above. In this case, calculate PRTRK by matrix computation using KAREL etc.

8. Now $PR_{TRK}$ of the standard part is calculated. Type this value in a position register.

## 5.7.4 Setup for VISUFRAME

When a vision system is used to detect $PR_{VIS}$ (user/part frame with respect to the World frame) of each part, use VISUFRAME instead of TRKUFRAME.
For the VISUFRAME, you need to specify $L_2$ which represents the part travel distance between the part detect switch and the part location where the snapshot is taken by the vision system. The value is measured by the user and put into the Vision Uframe Dist. in the Tracking Schedule Setup.
The Vision system should obtain $PR_{VIS}$ and pass it to the proper position register. Execute VISUFRAME instruction with this position register in the program (See Example 5.7.5(b)) so that the line tracking software automatically converts $PR_{VIS}$ into $PR_{TRK}$ (user/part frame with respect to the nominal frame).

## 5.7.5 Sample Tracking Uframe Program and Execution

In case of TRKUFRAME, use TRKUFRAME instruction to set user/part frame.
A sample TRKUFRAME teach pendant program is shown in Example 5.7.5 (a).

In this example, PR[1], [2] and [3] are used in the following purpose;
PR[1]: $PR_{TRK}$. This is updated per part during program execution.
PR[2]: $PR_{TRK}$ for the standard part. This is NOT updated during program execution.
PR[3]: Offset from the standard part (sensor output). This is updated per part during program execution.

OFFSET1 is a program in which the sensor detects the offset from the standard part then put the value into X, Y and R of PR[3]. This program is created by user based on the sensor specification.

**Example 5.7.5 (a) Sample TRKUFRAME Program**

```
 1: Line[1] ON;                  --turn on encoder2:LBL[1];
 2: LABEL[1];
 3: CALL HOME1;                  --home the robot
 4: WAIT DI[27]=ON;              --wait for part detect switch is triggered
 5: LINECOUNT[1] R[1];           --put the encoder count into the register
 6: SETTRIG LNSCH[1] R[1];       --set the trigger count
 7: SELBOUND LNSCH[1] BOUND[1];  --select a boundary set
 8:
 9: !set the tracking uframe
10: CALL OFFSET1;                --sensor to get offset and put it into PR[3]
11: PR[1]=PR[2]+PR[3];           --calculate PRTRK and set it into PR[1]
12: TRKUFRAME LNSCH[1] PR[1];    --set the tracking uframe
13:
14: CALL LNTK1;                  --call a tracking program
15: JMP LBL[1]                   --restart the process
```

In case of VISUFRAME, use VISUFRAME instruction to set user/part frame.
A sample VISUFRAME teach pendant program is shown in Example 5.7.5 (b).

In this example, PR[1] is used in the following purpose;
PR[1]: $PR_{VIS}$ (sensor output). This is updated per part during program execution.

VOFFSET1 is a program in which the sensor detects $PR_{VIS}$ (position of the part) then pass it into PR[1]. This program is created by user based on the sensor specification.

**Example 5.7.5 (b) Sample VISUFRAME Program**

```
 1: Line[1] ON;                   --turn on encoder2:LBL[1];
 2: LABEL[1];
 3: CALL HOME1;                   --home the robot
 4: WAIT DI[27]=ON;               --wait for part detect switch is triggered
 5: LINECOUNT[1] R[1];            --put the encoder count into the register
 6: SETTRIG LNSCH[1] R[1];        --set the trigger count
 7: SELBOUND LNSCH[1] BOUND[1];   --select a boundary set
 8:
 9: !set the tracking uframe
10: CALL VOFFSET1;                --sensor to get position and pass it into
PR[1]
11: VISUFRAME LNSCH[1] PR[1];     --set the tracking uframe
12:
13: CALL LNTK1;                   --call a tracking program
14: JMP LBL[1]                    --restart the process
```

> **NOTE**
> For one tracking schedule, you can use only one of the two instructions, TRKUFRAME and VISUFRAME.

In Example 5.7.5 (a) and Example 5.7.5 (b), HOME1 is not a tracking program, and LNTK1 is a tracking program like below;

**Example 5.7.5 (c) Sample Sub Program (LNTK1)**

```
1: L P[1] 1000mm/sec FINE;              --move to above of the part
2: L P[2] 1000mm/sec FINE;              --move to P2
3: WAIT 1.00(sec);                      --wait for one second
4: L P[3] 1000mm/sec FINE;              --move to P3
5: L P[4] 1000mm/sec FINE;              --move to P4
6: L P[5] 1000mm/sec FINE;              --move to P5
7: L P[2] 1000mm/sec FINE;              --move to P2
8: L P[1] 1000mm/sec FINE;                     --move to above of the part
```

## 5.7.6 Teaching and Executing the Tracking Uframe Program

After setup is done (Subsection 5.7.3 and Subsection 5.7.4), you can teach and execute the Tracking Uframe program as shown in Procedure 5-2.

### Procedure 5-3 Teach and Execute the Tracking Uframe Program

### Steps

1. Create a main program as shown in Example 5.7.5 (a) or Example 5.7.5 (b). At this time, the content of the sub program can be empty.
2. Run the main program from the start line to the line just before sub program CALL. During this, start the conveyor and let the part pass the part detect switch (detect a trigger), and then stop the conveyor at the proper position so that the robot can reach the part easily.
3. Record the tracking positions in the sub program (line tracking program).
4. Start the conveyor and test run the main program from the start line.
5. There is a limitation of rotational compensation ($\Delta$R) and the default limit value is 5deg. When the compensation value is larger than this limit, an alarm “LNTK-037 Rotation diff exceeds limit” is posted at the beginning of the tracking motion. If needed, modify the limit by the following system variable;

$LNSCH[schedule no.].$UFRM_RT_LIM (default is 5[deg])
Don't modify this system variable if Rail Tracking (Subsection 5.7.7).

Example path compensations are illustrated in the following figures. The translational compensation (ΔY) is in Fig. 5.7.6 (a) and the rotational compensation (ΔR) is in Fig. 5.7.6 (b). The dashed line represents the original path, and the solid line represents the path with compensation.

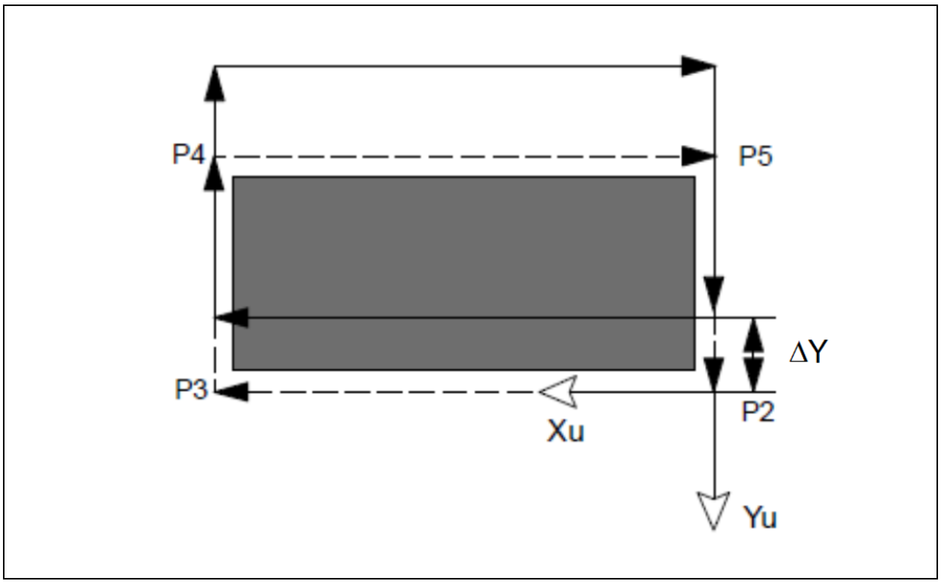



**Fig. 5.7.6(a) Path compensation in the Y direction**

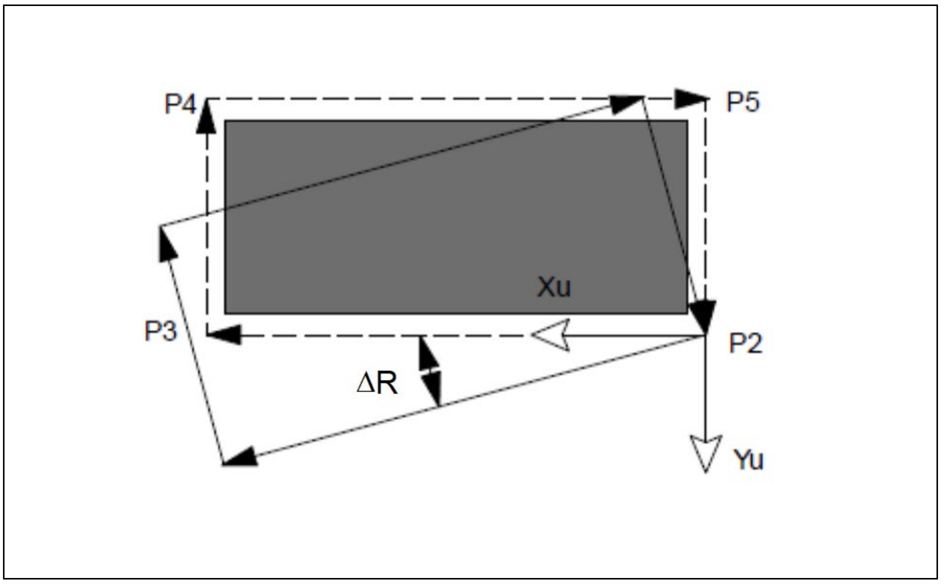



**Fig. 5.7.6(b) Path compensation in the R rotation**

## 5.7.7 Using TRKUFRAME and VISUFRAME in Rail Tracking

Rail tracking is the same as line tracking except that the robot is on an integrated rail (normally in the y direction) and the tracking frame is the same as the robot World frame. See Fig. 5.7.7.
Use the same procedure described in Subsection 5.7.3 and Subsection 5.7.4. Determine PRTRK for TRKUFRAME or PRVIS for VISUFRAME and set it into a position register.

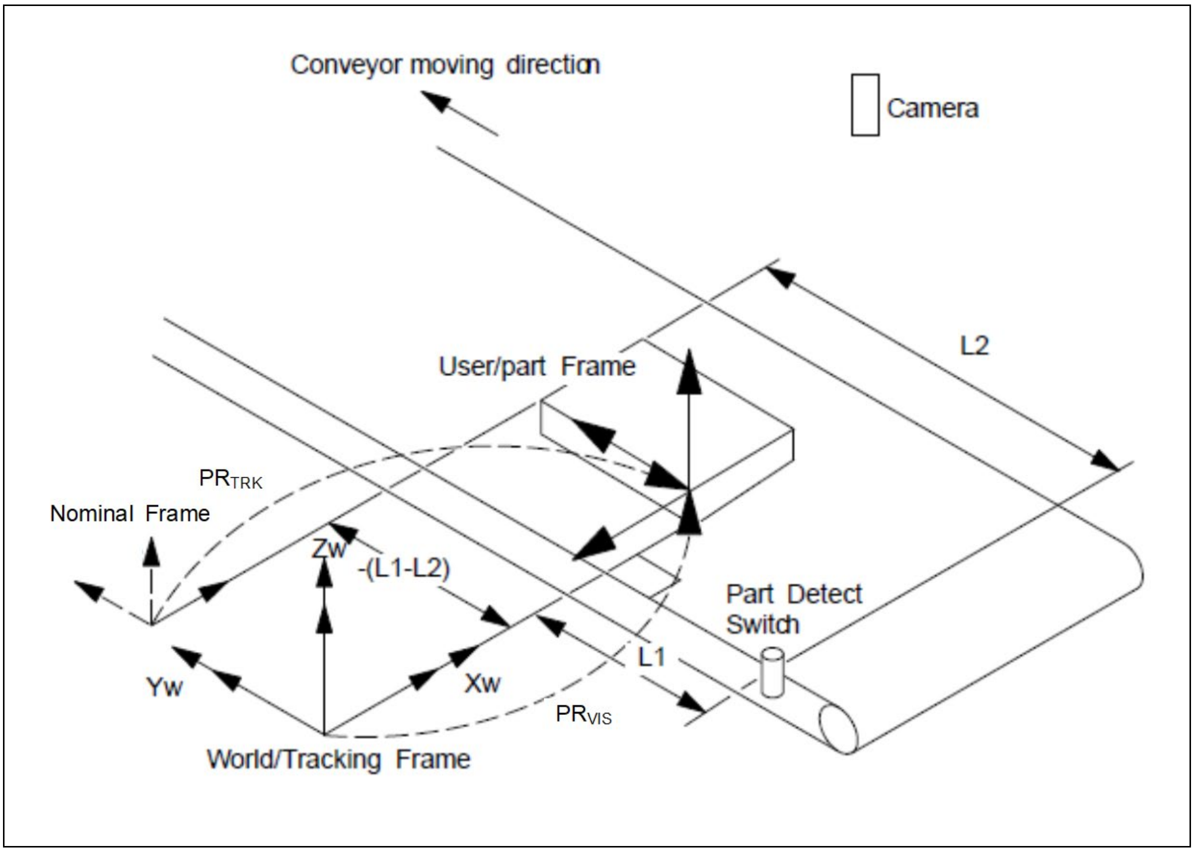



**Fig. 5.7.7 User/Part frame and frame relationship in rail tracking system**

# 5.8 SKIP OUTBOUND MOVE

Skip Outbound Move allows a part to travel out of the boundary window without stopping production. This speeds up production and eliminates the need for you to manage an error condition when this occurs. This feature is enabled using the following system variables:

- $LNCFG_GRP[group].$SKIP_OBNDMV : When this is TRUE, the skip outbound feature is enabled.
- $LNCFG_GRP[group].$SKP_ADJ_MS : Skip adjust time in milliseconds.
- $LNCFG_GRP[group].$SKP_FLG_NO : Flag number to turn on when the skip condition occurs.

When the feature is enabled, the system skips the motion instruction that causes the robot to go out of the down stream boundary. If skipped, the warning "LNTK-042 Skip outbound move" will be displayed.

> **NOTE**
> When enabling Skip Outbound Move, the adjustment time and the robot's motion after skipping should be carefully considered. If they are not appropriate, skips can occur at unexpected times, which is dangerous.

Typically a tracking program that picks up a part on the conveyor would have three tracking motion instructions: above pick (P1), pick (P2), and above pick (P3). Depending on the timing there are four possible conditions that could occur if the Skip Outbound Move feature is enabled:

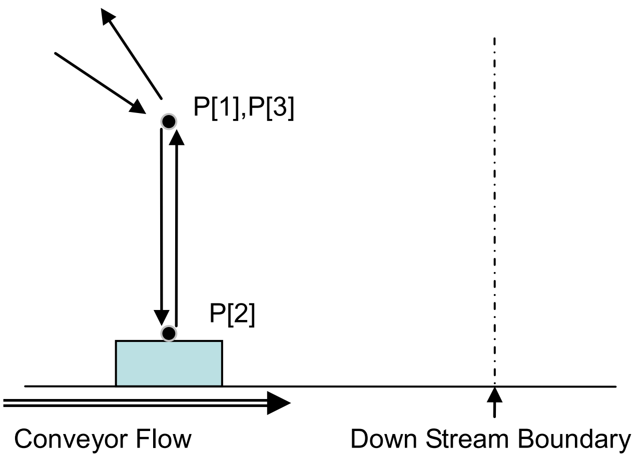



**Fig. 5.8 Example of tracking program that picks up a part**

1. When the program starts, P1, P2 and P3 might already be out of bounds. In this case the system will skip all three positions.
2. When the program starts, P1 is still inbounds, but P2 and P3 will be out of bounds when robot starts to move P2 and P3. In this case the system will reach P1 and skip P2 and P3.
3. When the program starts, P1 and P2 are inbounds. But when the robot reaches P1 and P2 but before the robot starts to move to P3, P3 becomes out of bounds. In this case the system will skip P3.
4. The system can reach all three positions while they are inside the boundary.

For a single pick program, the system will drop the part after picking up the part, so when the system skips the outbound move the robot will directly move to the non-tracking drop position. In this case there is no problem.
For a multiple pick program, the system will wait or execute the pick up for next part when current part is done. With the skip outbound move feature, the robot could be at P1 (condition 2) or P2 (condition 3) location when the skip condition was satisfied.
This feature only skips the outbound move. It does not guarantee the "TRAK-005 destination gone error" would never occur. If the previous motion is a tracking move, the robot might still track out of bounds while waiting for next part to be inbound when user did not specify to stop tracking.
The robot should not stay at the P2 position because it will hold the part at the conveyor position too long and cause the robot to block the part flow on the conveyor.
To overcome this problem, you must set up a system variable $LNCFG_GRP[group].$SKP_ADJ_MS to specify the time margin that would prevent this condition. This should be derived from the user program. The value should be the distance between P2 and P3 divided by the program speed of P3.
When the system determines whether or not P2 is out of bounds, the system uses this value to determine whether or not it has time to reach P3. If it does not have time to reach P3 then the system will skip P2 also. The system will adjust the time internally for a low override condition.
When you specify $LNCFG_GRP[group].$SKP_FLG_NO to a valid flag port the system will turn on the flag you specified when the skip condition occurred. Your application program can set this flag to determine whether or not to use the same tool to pick up the next part. Also, your program can request that the next robot picks up the skipped part. Because the system only sets the flag, you need to reset the flag before using it.
Because the motion is skipped, all the local conditions associated with the motion will be skipped.

# 5.9 LIMIT CHECKING

Before the line tracking motion is executed, the system will check to determine if the last axis will reach the limit or not. If it will reach the limit, then the system will change the direction of the last axis movement. This function works only when the all following conditions are satisfied:

- $LNCFG_GRP[group].$LMT_CHK_ENB = TRUE (this is FALSE by default).
- The motion is:
  - Line Tracking or Circular Tracking
  - Linear motion
  - No Wrist Joint motion instruction

The following system variables are used to support this function:

- $LNCFG_GRP[group].$LMT_CHK_ENB Enable/Disable function (default is FALSE)
- $LNCFG_GRP[group].$LMT_CHK_UL Upper soft limit margin (default is 20deg)
- $LNCFG_GRP[group].$LMT_CHK_LL Lower soft limit margin (default is 20deg)

For example, if the J6 axis stroke range is -360° to 360°, and both $LMT_CHK_UL and $LMT_CHK_LL are 20[deg], when the expected next destination exceeds the range from -340° to 340°, the robot will take another direction.

**NOTE**

Limit checking is worked in case of following tracking motion.

- Motion from reference point of tracking motion program to reference point of tracking motion program
- Motion from reference point of normal motion program to reference point of tracking motion program

Limit checking is not worked in case of following normal motion.

- Motion from reference point of tracking motion program to reference point of normal motion program

However, limit checking is avoided when the additional order ‘Wjnt’ is added on first motion order after tracking motion or when turn on bit18 (262144) of $LNCFG.$COMP_SW.

# 6 ADDITIONAL OPTIONS

## 6.1 HIGH SPEED SCANNING

### 6.1.1 Overview

The HDI for Line Tracking feature (J831) ensures an accurate part detection process when the conveyor operates at very fast speeds. This function uses HDI (High speed DI for application) in place of the standard digital input normally used for part detection.
To use HDI for Line Tracking, you must

- Enable the High Speed Scanning system variable
- Modify your line tracking program

### 6.1.2 Enabling High Speed Scanning

When the high speed scanning option is loaded, a new system variable, $HSLTENBL, is created. To enable this feature, you must set this variable to TRUE. Additional system variable, $HDI_FLAG[port], are used to enable and disable the HDI trigger dynamically. For example, when this flag variable is set to TRUE, HDI triggers will be accepted and processed; when FALSE, HDI triggers will be ignored.
Use Procedure 6-1 to enable the High Speed Scanning feature.

**Procedure 6-1 Enabling High Speed Scanning**

**Conditions**

- The High Speed Scanning option has been loaded.
- The Line Tracking option has been loaded.
- The part detect hardware is wired to HDI#1. (Refer to Appendix B)
- R-30*i*B / R-30*i*B Plus supports up to 5 High Speed Digital Input (HDI) #1 to #5, located on the JRL8 connector of the controller.
- R-30*i*B Mate / R-30*i*B Mate Plus supports up to 2 High Speed Digital Input (HDI) #1 to #2, located on the CRL3 connector of the controller.
- R-30*i*B Compact Plus / R-30*i*B Mini Plus supports up to 2 High Speed Digital Input (HDI) #1 to #2, located on the JRS30 connector of the controller.

**NOTE**

In R-30iB Mate controller, HDI interface is not available with Main CPU board A (A20B-8200-0790), Main CPU board B (A20B-8200-0791), Main CPU board D (A20B-8201-0420) or Main CPU board E (A20B-8201-0421). Main CPU board C (A20B-8200-0792) or Main CPU board F (A20B-8201-0422) is required to enable HDI for Line Tracking feature with R-30iB Mate / R-30*i*B Mate Plus controller. See Table 6.1.2(a) for detail.
In R-30*i*B Mate Plus controller, HDI interface is not available with Main CPU board A (A20B-8201-0750) or Main CPU board B (A20B-8201-0751). Main CPU board C (A20B-8201-0752) is required to enable HDI for Line Tracking feature with R-30*i*B Mate Plus controller. See Table 6.1.2(a) for detail.
In R-30*i*B Compact Plus controller, HDI interface is not available with Main CPU board A (A17B-8100-0800). Main CPU board B (A17B-8100-0801) is required to enable HDI for Line Tracking feature with R-30*i*B Compact Plus controller. See Table 6.1.2 (b) for detail.

**Table 6.1.2 (a) Requirements for R-30*i*B Mate / R-30*i*B Mate Plus Main CPU board (HDI interface)**

| Main CPU board | Board Specification | Available / Unavailable for HDI interface | Comments |
|---|---|---|---|
| Main CPU board A | A20B-8200-0790 (R-30*i*B Mate)<br>A20B-8201-0750 (R-30*i*B Mate Plus) | **Unavailable** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H001, A05B-2661-H001 (R-30*i*B Mate)<br>A05B-2655-H001, A05B-2662-H001 (R-30*i*B Mate (Open Air))<br>A05B-2680-H001 (R-30*i*B Mate Plus) |
| Main CPU board B | A20B-8200-0791 (R-30*i*B Mate)<br>A20B-8201-0751 (R-30*i*B Mate Plus) | **Unavailable** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H002, A05B-2661-H002 (R-30*i*B Mate)<br>A05B-2655-H002, A05B-2662-H002 (R-30*i*B Mate (Open Air))<br>A05B-2680-H002 (R-30*i*B Mate Plus) |
| Main CPU board C | A20B-8200-0792 (R-30*i*B Mate)<br>A20B-8201-0752 (R-30*i*B Mate Plus) | **Available** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H003, A05B-2661-H003 (R-30*i*B Mate)<br>A05B-2655-H003, A05B-2662-H003 (R-30*i*B Mate (Open Air))<br>A05B-2680-H003 (R-30*i*B Mate Plus) |
| Main CPU board D | A20B-8201-0420 (R-30*i*B Mate) | **Unavailable** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H004, A05B-2661-H004 (R-30*i*B Mate)<br>A05B-2655-H004, A05B-2662-H004 (R-30*i*B Mate (Open Air))) |

| Main CPU board | Board Specification | Available / Unavailable for HDI interface | Comments |
|---|---|---|---|
| Main CPU board E | A20B-8201-0421<br>(R-30*i*B Mate) | **Unavailable** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H005, A05B-2661-H005 (R-30*i*B Mate)<br>A05B-2655-H005, A05B-2662-H005 (R-30*i*B Mate (Open Air))) |
| Main CPU board F | A20B-8201-0422<br>(R-30*i*B Mate) | **Available** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2650-H006, A05B-2661-H006 (R-30*i*B Mate)<br>A05B-2655-H006, A05B-2662-H006 (R-30*i*B Mate (Open Air))) |

**Table 6.1.2(b) Requirements for R-30*i*B Compact Plus Main CPU board (HDI interface)**

| Main CPU board | Board Specification | Available / Unavailable for HDI interface | Comments |
|---|---|---|---|
| Main CPU board A | A17B-8100-0800 | **Unavailable** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2690-H001 |
| Main CPU board B | A17B-8100-0801 | **Available** | • The Main CPU board in the left column is included in the following order specification.<br>A05B-2690-H002 |

## Steps

1. Press MENU key.
2. Press NEXT.
3. Select SYSTEM.
4. Press F1, [TYPE].
5. Select Variables.
6. Move the cursor to the following variables and set their values accordingly,
   - $HSLTENBL = TRUE
   - $LNCFG.$HSDI_ENABLE = TRUE
7. Turn off the controller, and then turn it on again to accept the new setting.
8. Use the Encoder Setup Menu to input used HDI port id to HDI port id.
9. Turn off the controller, and then turn it on again to accept the new setting.

```
SETUP Encoders
                                            9/9
          Encoder Number :   1
     1 Encoder Axis :                         1
     2 Encoder Type :              Serial  INC
     3 Encoder Enable :                     OFF
       Current Count (cnts) :                 0
     4 Multiplier (ITP/update) :              1
     5 Average (updates) :                    1
     6 Stop Threshold (cnt/updt) :            0
     7 Simulate :          Enable :         OFF
     8            Rate (cnt/updt) :           0
     9 HDI port Id :                          1

   [ TYPE ]          ENCODER
```

> **NOTE**
> When HDI for Line Tracking (J831) is ordered and tracking sub program is taught by using normal DI input, set $LNCFG.$HSDI_ENABLE to FALSE. Otherwise, it is possible that tracking program can be not edited.

## 6.1.3 Modifying Your Line Tracking Program to Use High Speed Scanning

After you have set the high speed scanning variables to TRUE, you can change your line tracking programs to make use of the high speed scanning option. Example 6.1.3 (a) shows a standard line tracking program that does not use the high speed scanning feature.

**Example 6.1.3 (a) Main Program without High Speed Scanning instructions**

```
1: J P[1] 50% FINE                        -- MOVE TO HOME
2: LINE[1] ON                             -- ENABLE THE ENCODER
3: WAIT DI[1] ON                          -- WAIT FOR PART DETECT
4: LINECOUNT[1] R[1]                      -- GET TRIGGER VALUE
5: SETTRIG LNSCH[1] R[1]                  -- SET TRIGGER VALUE
6: SELBOUND LNSCH[1] BOUND[1]             -- SELECT A BOUNDARY
7: CALL TRACK                             -- CALL TRACKING PROGRAM
8: J P[1] 50% FINE                        -- MOVE TO HOME
```

Example 6.1.3 (b) shows the same program but includes instructions for using the high speed scanning feature. The part of the program that has changed is shown between the dashed lines.

**Example 6.1.3 (b) Main Program with High Speed Scanning Instructions**

```
 1: J P[1] 50% FINE                       -- MOVE TO HOME
 2: LINE[1] ON                            -- ENABLE THE ENCODER
 3: ! ------------------------------
 4: $HDI_FLAG[1] = 1                      -- ENABLE THE HDI port 1
 5: WAIT $ENC_STAT[1].$ENC_HSDI = 1       -- WAIT FOR PART DETECT
 6: ! ------------------------------
 7: LINECOUNT[1] R[1]                     -- GET TRIGGER VALUE
 8: SETTRIG LNSCH[1] R[1]                 -- SET TRIGGER VALUE
 9: SELBOUND LNSCH[1] BOUND[1]            -- SELECT A BOUNDARY
10: CALL TRACK                            -- CALL TRACKING PROGRAM
11: J P[1] 50% FINE                       -- MOVE TO HOME
```

> **NOTE**
> Be sure to execute all the steps from $ HDI_FLAG [1] = 1 to the SETTRIG instruction. If another process is added during this time, the HDI trigger may not be detected correctly. In Example 6.1.3 (b), if another program running in multitasking uses the LINECOUNT instruction after the WAIT on the 5th line of the main program and before the LINECOUNT instruction on the 7th line is executed, HDI may not be able to get the trigger value correctly.

### Differences between the Example Programs

1. WAIT for part detection
   - When using HDI, wait for $ENC_STAT[1].$ENC_HSDI = 1 (TRUE).
2. Setting the trigger input enablement flag $HDI_FLAG [1]
   - When the value is 1, the HDI input is activated.
   - When the value is 0, the HDI input is not activated (HDI is ignored even if it is input).
   - The value after part detection will be 0 because the next HDI will not be accepted until WAIT starts again.

> **NOTE**
> When you are editing tracking programs, conveyor resynchronization automatically uses the HDI #1 hardware input for part detection. Therefore, you do not need to modify $HDI_FLAG[port].

### Limitations

The High Speed Scanning option has the following limitations:

- The ACCUTRIG instruction can not be used simultaneously with High Speed Scanning.
- R-30*i*B / R-30*i*B Plus supports up to 5 High Speed Digital Input (HDI) #1 to #5, located on the JRL8 connector of the controller.
- R-30*i*B Mate / R-30*i*B Mate Plus supports up to 2 High Speed Digital Input (HDI) #1 to #2, located on the CRL3 connector of the controller.
- R-30*i*B Compact Plus / R-30*i*B Mini Plus supports up to 2 High Speed Digital Input (HDI) #1 to #2, located on the JRS30 connector of the controller.

## 6.2 ETHERNET ENCODER

### 6.2.1 Overview

A typical line tracking system uses conveyor/conveyors to transfer a work piece for robot/robots to process. When there are multiple robots working on the same conveyor, each robot needs to know the part's location on the conveyor. The Pulse Multiplexer is used to supply the encoder information to each robot on the same conveyor.

The Ethernet Encoder software option A05B-2600-R762 uses the Ethernet connection between robots instead of Pulse Multiplexer to supply the encoder information to each robot on the same conveyor. The Ethernet Encoder consists of both master and slave controllers where the master is the controller with the encoder(s) connect to it. The master controller transmits the encoder information to other slave controller/controllers over the Ethernet Network connection.

The Ethernet Encoder provides the following benefits:

- No Pulse Multiplexer is needed.
- No Encoder Cable required for slave controller.
- No Line tracking interface board required for slave controller.

**NOTE**
This feature is available with the encoders described in Section 2.1.

## 6.2.2 Explanation of Terms

### Master Controller and Slave Controller

Robot ring internally uses the communication function, “ROS interface packet over Ethernet (RIPE)”. The master controller is the master of RIPE. RIPE consists of one master controller and the other slave controllers. In detail of RIPE, refer to “ROS interface packet over Ethernet” in Ethernet Function OPERATOR’S MANUAL B-82974EN.

### Master Encoder and Slave Encoder

Master encoders are encoders connected with the controller. The information of these Encoders is transmitted to the other controllers over the Ethernet. The controller receiving information can refer to information of master encoders as it is connected with them. The encoder referred to over the Ethernet option is called as the slave encoder.

## 6.2.3 Create a Network

The Ethernet Encoder uses Ethernet network. This Ethernet network should be an isolated network so that it is not affected by building/plant network.

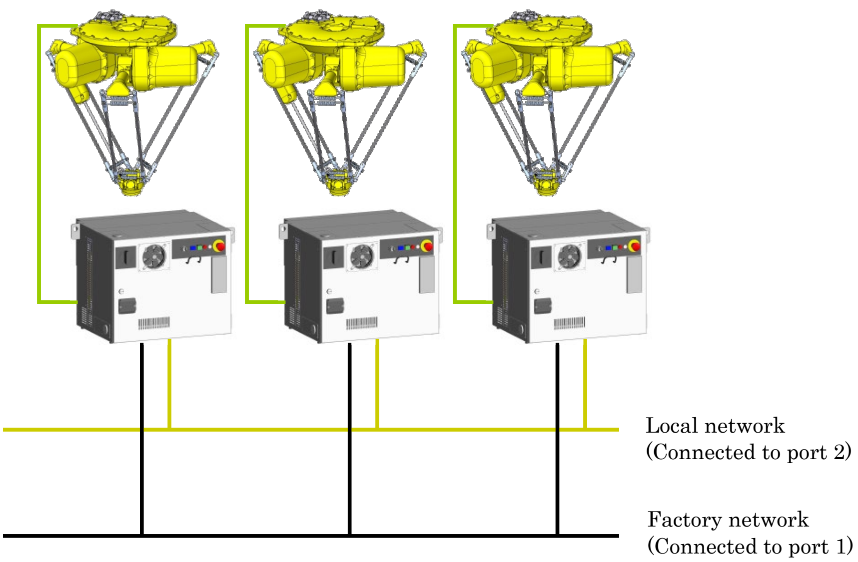



**Fig. 6.2.4.1 Ethernet network**

Robot ring consists of one master controller and the other slave controllers.

**NOTE**
The master encoder needs not to be connected with the master controller. However, it is recommended that the master encoder is connected directly with the master controller for decreasing the communication traffic.

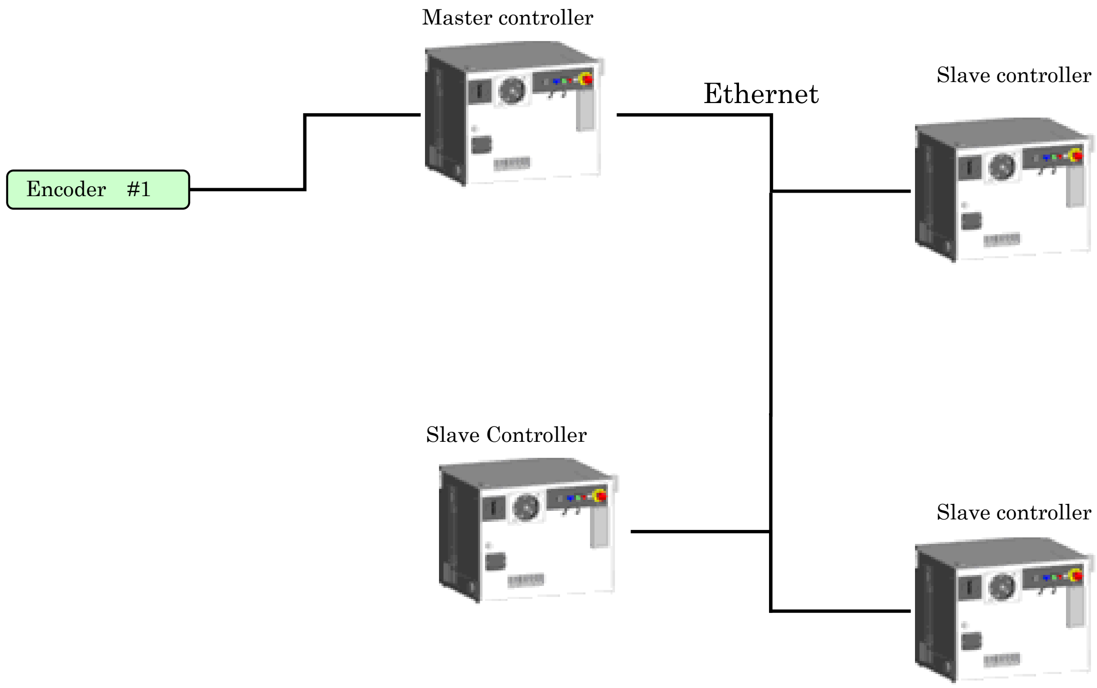



**Fig. 6.2.1 Hardware setup with the Ethernet encoder option**

## 6.2.4 Ethernet Encoder Setup

After completing the RIPE settings, set the Ethernet encoder configuration. Ethernet Encoder is set on Encoder setup menu.

### Procedure 6-2 Set up the Master Encoder

#### Steps

The following setup is performed on the controller connected physically with the encoder used as the Ethernet Encoder.

1. Press MENU key on the teach pendant.
2. Select [6 SETUP].
3. Select [Encoders].
4. Set “Encoder Axis”. Set the servo axis number used for the Master encoder.
5. Set “Ethernet Master RIPE Id” to RIPE index number of this controller.
6. Set “Ethernet Master Encoder” to the encoder number.
7. After completing the above settings for all master encoders, cycle power the controller.

For example, if you set the encoder 1 of RIPE index 2 controller as Master encoder, you set “Ethernet Master RIPE Id” to 1 and “Ethernet Master Encoder” to 1. The master controller screen is displayed. Encoder Set up menu is displayed as below.

```
SETUP Encoders
                                              10/10
           Encoder Number :   1
     1 Encoder Axis :                             1
     2 Encoder Type :              Serial INC
     3 Encoder Enable :                          ON
       Current Count (cnts) :                     0
     4 Multiplier (ITP/update) :                  1
     5 Average (updates) :                        1
     6 Stop Threshold (cnt/updt) :                0
     7 Simulate :          Enable :             OFF
     8           Rate (cnt/updt) :                0
     9 Ethernet Master RIPE Id :                  2
    10 Ethernet Master Encoder :                  1

  [ TYPE ]          ENCODER
```

## Procedure 6-3 Set up the Slave Encoder

## Steps

The following setup is performed on the controller where you want to use Ethernet Encoder.

1. Press MENU key on the teach pendant.
2. Select [6 SETUP].
3. Select [Encoders].
4. Set “Ethernet Master RIPE Id” and “Ethernet Master Encoder” to the same values as when you set the master encoder.
5. After completing the above settings for all slave encoders, cycle power the controller.

**NOTE**

In case that set up Pulsecoder A which are connected to the encoder terminal on own Main CPU board (Encoder Type : Main Serial INC) and Pulsecoder B which are connected to the encoder terminal on another Main CPU board (Encoder Type : Main Serial INC) using Ethernet Encoder function on one controller, Pulsecoder A must be set up to the smaller encoder number than the encoder number that Pulsecoder B is set up. This limitation is also true in the case of 3 or more Pulsecoders. Refer to Fig.6.2.5.

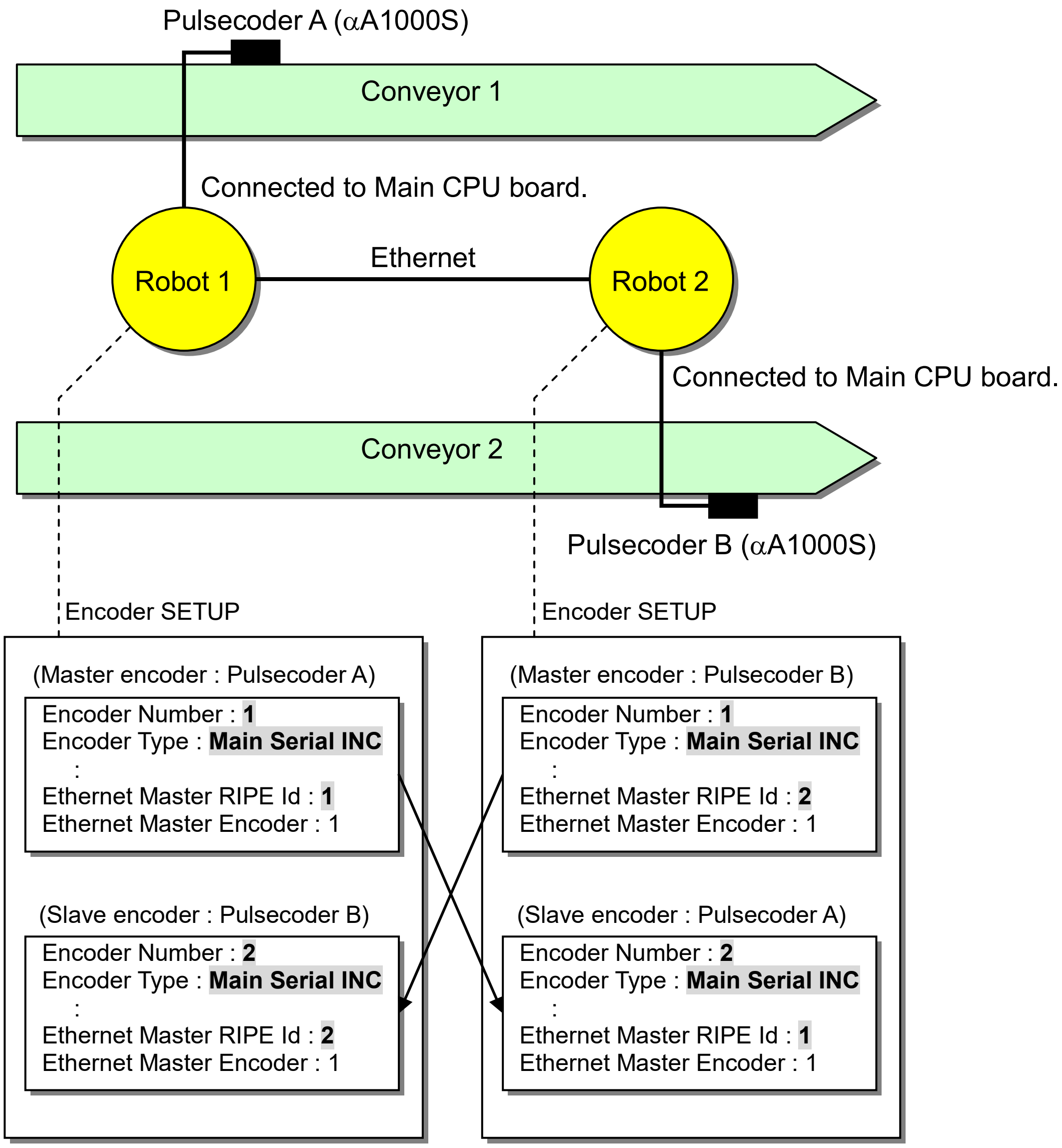



**Fig. 6.2.5 Example of two Pulscoder settings connected to the main CPU board of two robot controllers**

## 6.2.5 Verify Setup

After completing all the settings, move the conveyor and check that the "Current Count (cnts)" of the master encoder and all slave encoders changes. In addition, if the conveyor is stopped, the "Current Count (cnts)" will be the same for the master encoder and all slave encoders.

**NOTE**
If you use the simulation mode on the master encoder setting screen, you can check the same without moving the conveyor.

## 6.2.6 Limitations

- Each controller can support up to four master encoders.
- The Ethernet Encoder in one RIPE network can support up to four controllers.
- A controller supports only one RIPE network.
- Software versions of all robot controllers that participate in the communication must be the same.
- The $SCR.$ITP_TIME for all the controllers should be set to the same value. Please set the maximum value among all controllers on RIPE network. After setting, the controller needs cycle power.
- Encoders using servo conveyor line tracking function is not allowed to be used as the Master Encoder. Encoder numbers that specify an encoder type other than the servo conveyor can be used as the master encoder.
- The communication causes the difference between the master encoder's pulse count and the slave encoder's pulse count. The difference is converted into time is about plus or minus 2ms as compared with using Pulse Multiplexer.

# 6.3 SERVO CONVEYOR LINE TRACKING

## 6.3.1 Overview

Servo Conveyor Line Tracking option (J589) is the function for using an extended axis as a conveyor. Therefore, the robot can track the conveyor that is indexed. This function requires Line Tracking option (J512) and Independent Auxiliary Axis option (H895).
Servo Conveyor Line Tracking option (J589) includes Multi Motion Group option (J601) and Continuous Turn option (J613). These are used for keeping on moving an extended axis as a conveyor.

**NOTE**
The motor selection should consider motor speed and motor torque. Motor speed is dependent on the desired indexer speed and gear box used. Motor torque is dependent on what kind of parts (mass), how many parts on the whole conveyor, the mass and inertia of conveyor (from above information) and gear box used. When motor selected is under size, some problems may occur during the continue operation.

## 6.3.2 Independent extended axis setup for Index Servo Conveyor

Independent Extended Axis is set on the robot maintenance screen of the control start menu. For details on the setting procedure, refer to "INDEPENDENT ADDITIONAL AXIS SETUP" in Basic Function Manual (B-83284EN). When adding the Independent Extended Axis, some setting items enter values as described in order to use the additional axis as Indexer. For other setting items, refer to Basic Function Manual (B-83284EN) and set them appropriately according to the system configuration.

### Independent axes type

Select “2 Rotary Axis” as Independent axis type.

```
-- INDEPENDENT AXES TYPE –
1. Linear Axis
2. Rotary Axis

Select? 2
```

### Gear Ratio

Enter the number of revolution of the motor which corresponds to one pitch of the conveyor. By this setting, a conveyor moves one pitch when Independent Axis moves 360 degrees.

```
-- GEAR RATIO --
Enter Gear Ratio?
```

Pitch: The distance of conveyor when move conveyor to one bucket of conveyor
Bucket: A part of a conveyor is divided by a constant distance.

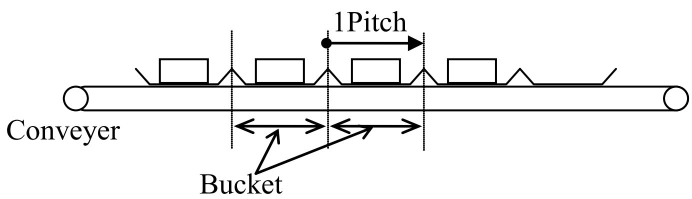



**Fig. 6.3.2 (a) Bucket and Pitch**

If a conveyor rotates “N” times and themotor of it rotates “M” times, the gear ratio can be calculated by the following. Please enter the calculated value.

$$\mathrm{GearRatio} = \frac{\mathrm{M}}{\mathrm{N} \times \mathrm{Number\ of\ Buckets}}$$

- It is also possible to calculate the gear ratio from the set value (Motor Gear teeth and so on) of Servo Conveyor Setup. In this case, the gear ratio can be calculated by the following.

$$\mathrm{GearRatio} = \frac{\mathrm{RotorInput\ Gear\ Teeth} \times \mathrm{Conveyer\ Belt\ Teeth}}{\mathrm{Motor\ Gear\ Teeth} \times \mathrm{Rotor\ Output\ Gear\ Teeth} \times \mathrm{Number\ of\ Flight}}$$

### Max Joint Speed

Select “2:NO Change” for setting a suggested speed as a max joint speed.

```
--MAX JOINT SPEED SETTING --
  Suggested Speed = 800.000(deg/s)
  (Calculated with Max Motor Speed)

Enter (1:Change, 2:No Change)? 2
```

### Motor Direction

Choose “1:TRUE” if the joint coordinate position of the conveyor increases when the motor rotates in the plus direction.
Choose “2:FALSE” if the joint coordinate position of the conveyor decreases when motor rotates in the plus direction.

```
-- MOTOR DIRECTION
INDEPENDENT AXES 1 Motion Sign = TRUE
Enter (1:TRUE, 2:FALSE)?
```

If you look at a motor from the front of the flange, a counter clockwise rotation is plus direction of a motor.

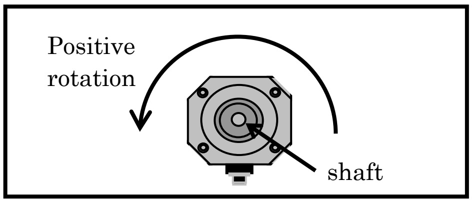



**Fig. 6.3.2 (b) Positive rotation of a motor**

### Upper Limit and Lower Limit

Enter 180[deg] as an upper limit and –180[deg] as a lower limit.

```
-- UPPER LIMIT --
Enter Upper Limit ( deg)? 180
```

```
--LOWER LIMIIT--
Enter Lower Limit ( deg)? -180
```

### Master Position

Normally, the position is “0”.

```
--MASTER POSITION –
Enter Master Position (deg)?
```

### ACC/DEC TIME

Set the values of "the 1st ACC/DEC Time", "the 2nd ACC/DEC Time", and "Minimum Accel Time" to achieve the required speed at the time of maximum production.
For the index conveyor type, It is very important that axis capability is set up so that desire maximum production indexing speed can be achieved. One indexing time = motion time + dwell time. Motion time = acceleration time + deceleration time. Set "the 2nd ACC/DEC Time" to half the value of the "the 1st ACC/DEC Time"."Minimum Accel Time" is the sum of "the 1st ACC/DEC Time" and "the 2nd ACC/DEC Time".

> **NOTE**
> For example, the application required maximum product is 200 ppm (300 ms per index) with dwell of 100 ms. The motion time for motor per indexing will be 200 ms. So the acceleration time should be less than 100 ms. So the system should be set acc time 1 = 64 , acc time 2 = 32 and min acc time = 96 ms.

## 6.3.3 Index Servo Conveyor Setup

Set up Index Servo Conveyor by the following step. The setting of encoder and the setting of Continuous Turn are updated automatically by setting Servo Conveyor.

### Procedure 6-5 Index Servo Conveyor Setup

**Step**

1. Press the MENU key.
2. Select “SETUP”.
3. Press “F1: TYPE”.
4. Select “Indexers”. You will see a screen as following.

```
FANUC Conveyor:
                                             1/4
          FANUC Conveyor:  1
     1  Encoder Number:                          1
     2  Conveyor Type:            Index
     3  Tracking Schedule:                       1
        Encoder Scale (cnt/mm):      524288
     4  program and speed control:  Detail

   [ TYPE ]          CNVY
```

5. Move the cursor to “Encoder Number” and enter the number of the encoder that is used as servo conveyor.
6. Move the cursor to “Conveyor Type” and select “Index”.

```
FANUC Conveyor:
                                          2/4
          FANUC Conveyor:  1
            1  er Number:                        1
 1      Index  yor Type:            Index
 2 Continuous  ng Schedule:                      1
 3             er Scale (cnt/mm):      524288
 4             am and speed control:  Detail
 5

   [ TYPE ]   DETAIL    CNVY    [CHOICE]]
```

7. Move the cursor to “Tracking Schedule” and enter the value of the tracking schedule that is used for servo conveyor.
8. Move the cursor to “Conveyor Type” and press “F2: DETAIL” key or Enter Key. You will see a screen for setting up axis.

9. You will see a screen as following. Because there is no [TYPE] in this menu, you can not select any other setup menu. To return back to previous menu with [TYPE] press PREV key.

```
SETUP Indexer axis:
                                      1/12
         Indexer  1: FANUC motor DONE
         Encoder Number:    1
     1  Robot Group:                       2
             Axis:                         1
     2  Motor Gear teeth                   1
     3  Rotor input Gear teeth             1
     4  Rotor output Gear teeth            1
     5  Conveyor belt teeth                1
     6  Number of Flight                 100
     7  Index Distance (mm)           10.000
     8  Index Advance Trigger DI:          1
     9  Delay move after trig(ms):        16
    10  Indexer Ready DO:                  1
    11  Flag for internal use:           131
    12  Flag 2 for internal use:         132

          EXEC
```

10. Move the cursor to “Robot Group” and enter the group number of the extended axis for the servo conveyor and then move the cursor to “Axis” and enter the axis number.
11. Move the cursor to “Motor Gear teeth”, “Rotor input Gear teeth”, “Rotor output Gear teeth” and “Conveyor belt teeth” and enter each value in the positive integer. Move the cursor to “Number of Flight” and enter the number of bucket on the servo conveyor. The following figure shows the relationship between these values. The gear ratio of the servo conveyor is calculated from these value.

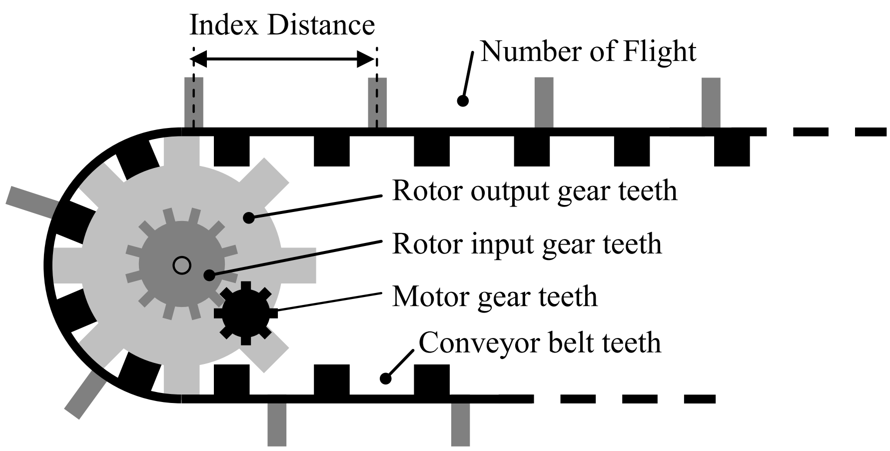



**Fig. 6.3.3 Relationship between motor, rotor and conveyor**

- It is also possible to set up from the relation between the number of revolution of motor and the number of revolution of the conveyor. If a conveyor rotates “N” times and a motor of it rotates “M” times, please setup as following. “M” and “N” must be an integer.
  a. Enter the value of “N” to “Motor Gear teeth”
  b. Enter the value of “M” to “Rotor input Gear teeth”
  c. Enter “1” to “Rotor output Gear teeth” and “Conveyor belt teeth”

12. Move the cursor to “Index Distance” and the distance a one pitch on the servo conveyor.
13. Move the cursor to “Index Advance Trigger DI” and enter the index of DI is used for moving the servo conveyor.

14. Move the cursor to "Delay move after trig (ms)" and enter the value of delay time until the servo conveyor starts after trigger.
15. Move the cursor to "Indexer Ready DO". If you would like to output DO at start of TP program for the servo conveyor, enter the index of DO. Please refer to TP program for Servo Conveyor (INDXG*.TP).
16. Move the cursor to "Flag for internal use" and enter the index of Flag is used for the servo conveyor. Please confirm that the specified Flag is not used for another use.
17. Move the cursor to "Flag 2 for internal use" and enter the index of Flag is used for the servo conveyor. Please confirm that the specified Flag is not used for another use.
18. After the above setup, press "F2: EXEC" and be sure to Power off/on.

> **NOTE**
> By the above setting, Continuous Rotation setup of the conveyor and Encoder setup of the specified encoder are also done.

## 6.3.4 Index Servo Conveyor program

It is necessary to prepare TP program for moving the servo conveyor on tracking because the servo conveyor is setup as extended axis. By following step, the standard TP program for moving the servo conveyor is created.

### Procedure 6-6 Create Index Servo Conveyor program

> **NOTE**
> Before generating indexer program, please set max payload which is used in an indexer tracking motion on MOTION/PAYLOAD SET display.

#### Step

1. Move the cursor to "program and speed control: Detail" in "SETUP Indexers" and press "F2: DETAIL" key or ENTER key.

```
FANUC Conveyor:
                                          4/4
          FANUC Conveyor:  1
     1   Encoder Number:                    1
     2   Conveyor Type:             Index
     3   Tracking Schedule:                 1
         Encoder Scale (cnt/mm):      524288
     4   program and speed control:   Detail

    [ TYPE ]   DETAIL   CNVY
```

You will see a screen as following. Because there is no [TYPE] in this menu, you can not select any other setup menu. To return back to previous menu with [TYPE] press PREV key.

```
SETUP Indexer Program:
                                    1/4
              FANUC Indexer:  1
    1  Index Speed (part/min)        100
    2  Index Dwell (ms):               0
    3  Indexer Register start         60
    4  Generate Index program:
```

2. Move the cursor to “Index Speed (part/min)” and enter the value of Index Speed of the servo conveyor. Please set the number of pitches per minute.
3. Move the cursor to “Index Dwell” and enter the value of the time to stop the servo conveyor. The speed pattern of the servo conveyor is different according to the value.

Movement Time of Conveyor per Pitch [ms] = (60*1000/Index Speed) – Index Dwell

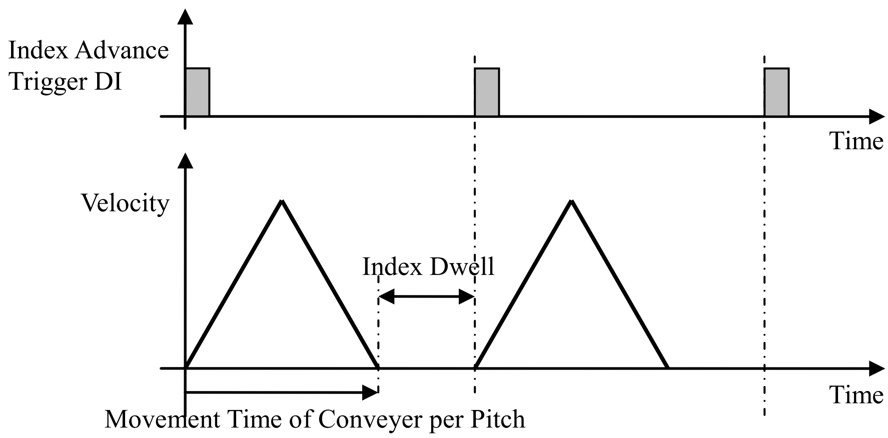



**Fig. 6.3.5 Speed pattern**

4. Move the cursor to “Indexer Resister start” and enter the start index of Register is used in TP program for the servo conveyor. Two registers are used in TP program.
   For example, R[60] and R[61] are used when “Indexer Resister start” is “60”. R[60] is used for setting the number of DO. The specified DO by R[60] is used for checking whether the conveyor finishes moving to initial position. R[61] is used for counting the number of pitches of the conveyor. Please refer to example of INDXG*.TP.
5. After the above setting move the cursor to “Generate Index program” and push “F2: Create” key. If there is no problem in the setting, “Done” is displayed and INDXG*.TP (* is group number of the servo conveyor) is created.

## NOTE

- Do not change INDXG*.TP directly because it is default program in the system. Please rename INDXG*.TP and use it.
- “Part Rate exceeds allowable value” might be displayed by the setting. This message shows that the calculated conveyor acceleration from the setting for TP exceeds allowable acceleration of the servo conveyor or robot. In this case, INDX*.TP is not created. Please adjust the setting (Index Speed, Index Dwell) so that the acceleration decreases.
- If there is INDXG*.TP and you push “F2: Create” key, “Index Program already exist” is displayed. In this case, INDXG*.TP is not created. If you want to create TP program, please push “SHIFT” key and “F3: Replace” key. If there is no problem in the setting, existing INDXG*.TP is overwritten by the created TP program from current setting.

- If you push only “F3 Replace” key, “Hold Shift & Replace to replace program” is displayed. In this case, INDXG*.TP is not overwritten.
- If INDXG*.TP is currently open in editor, the system will not be able to update the program. In this case the error message will displayed and “done” will not be displayed.
- “Part Rate exceed allowable value” might be displayed by the setting when you push “SHIFT” key and “F3: Replace” key. Please adjust the setting so that the acceleration decreases.

---

## Sample Program

The following is an example of INDXG*.TP.

**Example 6.3.5 INDXG*.TP**

```
1:J P[1] 50% CNT0       ;                      Movement to P[1] as an initial position.
2:   IF R[60:G2 Ready DO]=0,JMP LBL[3] ;
3:   DO[R[60]]=ON ;                             Specified DO by R[60] become on
4:   LBL[3] ;
5:   R[61:G2 cur slot ID]=0       ;             Reset R[61]
6:   LBL[1] ;
7:   $INDEXER[1].$INDEX_MV=1 ;                  Setting for waiting for DI trigger
8:J P[2] 180msec CNT100 INC ACC66       ;       One pitch movement
9:   $INDEXER[1].$INDEX_MV=0 ;                  Setting for waiting for DI trigger
10:   R[61:G2 cur slot ID]=R[61:G2 cur slot ID]+1;   Count the number of pitches
11:   IF R[61:G2 cur slot ID]=32,JMP LBL[2] ;   If conveyor is turned once, Jump to LBL[2].
12:   JMP LBL[1] ;
13:   LBL[2] ;
14:J P[1] 100% CNT100       ;                   Movement to P[1] to correct iteration error
15:   R[61:G2 cur slot ID]=0       ;            Reset R[61]
16:   JMP LBL[1] ;

/POS

P[1:""]{
   GP2:
          UF : 0, UT : 1,
          J1=        0.000 deg
};
P[2:""]{
   GP2:
          UF : 0, UT : 1,
          J1=     360.000 deg
};
```

- At line 1, the conveyor moves to the initial position P[1].
- DO is output to check whether the conveyor finishes moving to the initial position by instructions from line 2 to line 4.
- At line 8, the conveyor moves 1 pitch. In other words, the extended axis that is used as the conveyor moves 360 degrees.
- ”$INDEXER[*].$INDEX_MV=1” at line 7 and “$INDEXER[*].$INDEX_MV=0” at line 9 are setting for watching input of Index Advance Trigger DI (* is number of Servo Conveyor). Motion instruction at line 8 is not executed until the DI changes from OFF to ON.
- IF instruction at line11 checks whether the conveyor makes one revolution. In this example case, the conveyor makes one revolution per 32 pitches.
- At line 14, the conveyor moves to P[1] to correct iteration error. Every time the conveyor makes one revolution this line is executed.
- R[60] and R[61] are set by specifying “Indexer Register start”. R[60] is used for setting the number of DO. The specified DO by R[60] is used for checking whether the conveyor finishes moving to initial position. R[61] is used for counting the number of pitches of the conveyor.

## 6.3.5 Tracking schedule setup

Tracking Schedule Setup for servo conveyor is the same as ordinary Line Tracking function. Please refer to the operator's manual for Line Tracking. Refer to Section 3.3 to set the tracking schedule used by Servo Conveyor Tracking.

## 6.3.6 Example of main program

Basically, how to teaching TP program is the same as ordinary Line tracking function. However, it is necessary to run TP program for moving a servo conveyor by multitask. CONVEYOR.TP (Program renamed from INDXG*.TP) for moving a servo conveyor is run at line 3 in the following Sample Program.

**Example 6.3.6 Main Program**

```
1:   J P[1] 100% FINE ;
2:   LINE[1] ON ;
3:   RUN CONVEYOR;                CONVEYOR.TP for moving Servo Conveyor is executed.
4:   LBL[1] ;
5:   WAIT DI[2]=ON       ;
6:   LINECOUNT[1] R[1] ;
7:   SETTRIG LNSCH[1] R[1] ;
8:   CALL TRACK       ;           TRACK.TP for tracking motion is executed.
9:   CALL NORM       ;            NORM.TP for non-tracking motion is executed.
10:   JMP LBL[1] ;
```

> **NOTE**
> For a system in which workpieces are provided one after another even while the robot is running and the indexer repeatedly moves at high speed, please refer to the sample program in Subsection 6.3.9 “KAREL Program for Servo Conveyor Line Tracking”.

## 6.3.7 Error tune variable

On Servo Conveyor Line Tracking, adjust an error of the tracking motion by using the following parameter. Refer to Section 5.6 for how to adjust this parameter.

- Static Tuning Variable (I/O Delay):
  $ENC_IO_DLYF[encoder number]

- Dynamic Tuning Variable (Servo Delay):
  $SLTK_GRP[number of servo conveyor group].$SRVO_DELAY

> **NOTE**
> If you are using the KAREL program for servo conveyor line tracking to set the trigger value, you do not need to adjust the Static Tuning Variable (I/O Delay).

### 6.3.8 Wait indexer stop function

This function is disabled by default. When this function is disabled, a robot starts/ends a tracking regardless of the motion of the servo conveyor. Therefore, for example, if robot starts a tracking motion while a servo conveyor accelerates, the acceleration of the robot is increased in order to catch up the conveyor and the motion of the robot might become aggressive.
When this function is enabled, Robot starts or ends a tracking motion while a servo conveyor stops. By this setting, the acceleration at the start or end of tracking motion is restricted.

> **NOTE**
> If you want to enable this function, it is necessary to set Index Dwell to 0 and over and generate TP program for Servo Conveyor again. If there is no time the conveyor stops, it is not possible to start/end a tracking motion.

The flag for switching the setting of this function is bit 1 and bit 2 of $INDX_TRACK[schedule number]. If you want to enable this function, please set these flag by the following procedure.

- If bit 1 of $INDX_TRACK[schedule number], Robot starts a tracking motion while a servo conveyor stops.
- If bit 2 of $INDX_TRACK[schedule number], Robot ends a tracking motion while a servo conveyor stops.

> **NOTE**
> If the setting of this function is changed, please create TP program for the servo conveyor again.

#### Procedure 6-7 Wait Indexer Stop Function Setup

**Step**

1. Setting bit 1 of $INDX_TRACK[schedule number]

1.1 Please divide the value of $INDX_TRACK[schedule number] by “2” and then check current setting by following step.
- If the integer part of the calculated value is <u>Odd</u> number, bit 1 is TRUE.
- If the integer part of the calculated value is <u>Even</u> number, bit 1 is FALSE.

1.2 Please change the value of $INDX_TRACK[schedule number] according to the current setting.
- If bit 1 is TRUE and you would like to <u>change it to FALSE</u>, please subtract “2” from the value of $INDX_TRACK[schedule number].
- If bit 1 is FALSE and you would like to <u>change it to TRUE</u>, please add “2” to the value of $INDX_TRACK[schedule number].

2. Setting bit 2 of $INDX_TRACK[schedule number]

2.1 Please divide the value of $INDX_TRACK[schedule number] by “4” and then check current setting by following step.
- If the integer part of the calculated value is <u>Odd</u> number, bit 2 is TRUE.
- If the integer part of the calculated value is <u>Even</u> number, bit 2 is FALSE.

2.2 Please change the value of $INDX_TRACK[schedule number] according to the current setting.
- If bit 2 is TRUE and you would like to <u>change it to FALSE</u>, please subtract “4” from the value of $INDX_TRACK[schedule number].
- If bit 2 is FALSE and you would like to <u>change it to TRUE</u>, please add “4” to the value of $INDX_TRACK[schedule number].

3. After the above setting, please Power Off/ON.

## 6.3.9 KAREL program for servo conveyor line tracking

Servo Conveyor Line Tracking provides the following KAREL programs. By using these, it is possible to get a trigger value and save it every time Servo Conveyor moves a fixed distance. It is also possible to use Discard Line function and Stop Conveyor function and Distribution Line function.

- SLTKINIT
- SLTKREST
- SLTKPSHQ
- SLTKPOPQ
- SLTKDELQ
- SLTKRSTQ
- SLTKGTPP

This subsection explains these KAREL programs and Discard Line, Stop Conveyor and Distribution Line.

> **NOTE**
> It is necessary to set $KAREL_ENB=1 to use KAREL program.

---

### SLTKINIT

This program clears information about trigger values. Normally, this program is called once when the system is started.

Argument1: Specify the number of tracking schedule which Servo Conveyor Line Tracking uses.
Argument2: Specify the position register number that is saved an initial position of the conveyor.
Argument3: Specify the position register number.

The specified position register by argument 3 is set to the distance between the current conveyor position and an initial position of the conveyor. The value within 1 pitch (from 0 to 360 degrees) is set. This is available for adjusting the conveyor position at the start.

### SLTKREST

Because the distance of one pitch of Servo Conveyor is constant, an encoder value can be got as a trigger value every a constant distance. This program is used for setting a reference value of the trigger. When SLTKREST is called, the encoder value is saved as a reference value of the trigger. It is necessary to call this program once just after the servo conveyor moves to an initial position.

Argument1: Specify the number of tracking schedule which Servo Conveyor Line Tracking uses.

In the following case, PR[2] is set to the distance between the current conveyor position and an initial position of the conveyor by SLTKINIT at line 1. Next, the conveyor moves to an initial position by the motion instruction and Reference position for trigger is saved by SLTKREST.

```
1:   CALL SLTKINIT(1,1,2)      ;Initialization and Setting of Pos. Register.
2: J PR[2] 50% CNT0 INC       ;               Movement to an initial position.
3:   CALL SLTKREST(1) ;                       Setting of Reference Pos. for Trigger.
```

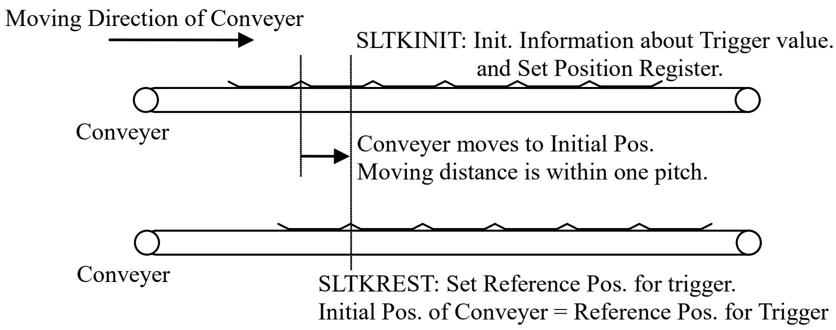



**Fig. 6.3.9(a) SLTKINIT, SLTKREST**

## SLTKPSHQ

This is used for saving trigger value after the conveyor moves the number of pitch.

Argument1: Specify a number of tracking schedule which Servo Conveyor Line Tracking uses.
Argument2: Specify a number of pitch

If you would like to change the save-able number of trigger values, please change the value of the following system variable and power off/on.
$SLTKSCH[n].$QUE_SIZE: Default value is 20 and max value is 100.
"n" is the number of tracking schedule which Servo Conveyor Line Tracking uses.

> **NOTE**
> When the saved number of trigger values become the specified value by $SLTKSCH[n].$QUE_SIZE, the oldest saved trigger value is removed.

It is necessary call this program after the conveyor moves the specified number of pitch. When this program is called, a trigger value is calculated from the reference trigger value and the specified number of pitch and it is saved. At the first, the reference trigger value is saved as a trigger value.
For example, if the conveyor moves 1 pitch and the trigger value is saved once, Argument2 should be set "1". If the conveyor moves 2 pitches and the trigger value is saved once, Argument2 should be set "2". The following shows the case that trigger values are saved during the conveyor moves four pitches.

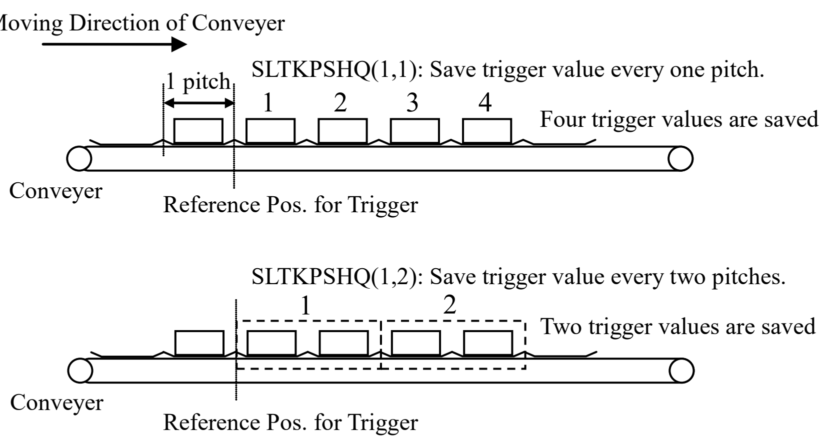



**Fig. 6.3.9(b) SLTKPSHQ**

### SLTKPOPQ

This program is used for getting the saved trigger value. When this program is called, it is possible to get the oldest saved trigger value. If you succeed in getting the trigger value, the trigger value is removed and it is impossible to get it again.

Argument1: Specify a number of tracking schedule which Servo Conveyor Line Tracking uses.
Argument2: Specify a register number for getting a trigger value.
Argument3: Specify a register number for getting a status. If SLTKPOPQ fails to get a trigger value the value of status become nonzero.

The following shows the case that there are four saved trigger values.

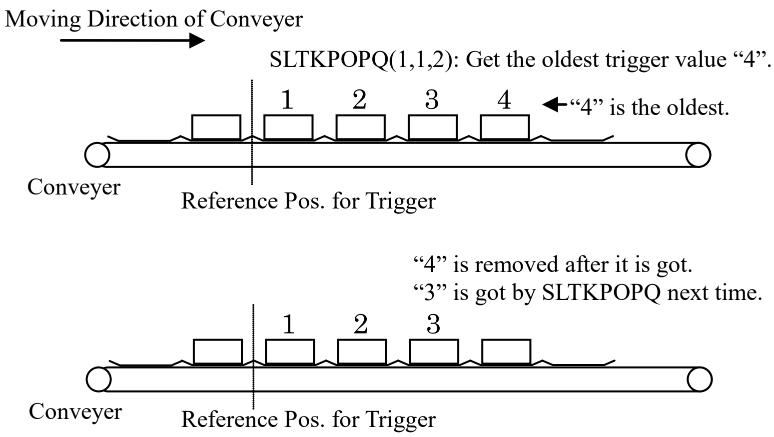



**Fig. 6.3.9(c) SLTKPSHQ**

### SLTKDELQ

This program is used for deleting the specified the saved trigger value. It is not possible to get the deleted trigger value by SLTKPOPQ.

Argument1: Specify a number of tracking schedule which Servo Conveyor Line Tracking uses.
Argument2: Specify a number of trigger to delete.

For example, if argument2 is “2”, SLTKDELQ deletes the second trigger value “2” from the newest trigger value.

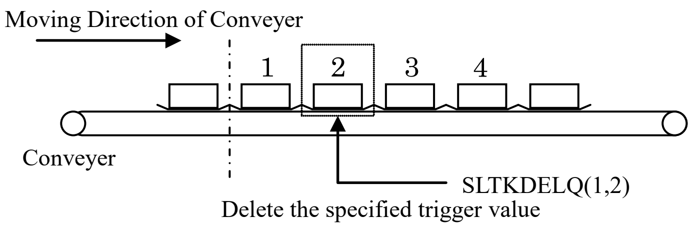



**Fig. 6.3.9(d) SLTKDELQ**

> **NOTE**
> If the selected trigger value does not exist, a message "Trigger count does not exist" is displayed.

### SLTKRSTQ

This program turns off the specified DO for Stop Conveyor. Please refer to STOP CONVEYOR below.

Argument1: Specify a number of tracking schedule which Servo Conveyor Line Tracking uses.

## SLTKGTPP

This program is used for calculating a distance along X axis of tracking frame from the origin of tracking frame to the corresponding position to the oldest available trigger value.

Argument1: Specify a number of tracking schedule which Servo Conveyor Line Tracking uses.
Argument2: Specify a register number for getting a distance from the origin of tracking frame.
Argument3: Specify a register number for getting a status. If SLTKGTPP succeeds, the value of the status becomes "0". If SLTKGTPP fails to calculate a distance, the value of the status becomes nonzero and the value of the register specified by argument 2 is not updated.

The following shows the case that there are four saved trigger values.
For example, if the oldest available trigger value is "4", the corresponding distance to the trigger value "4" is calculated.

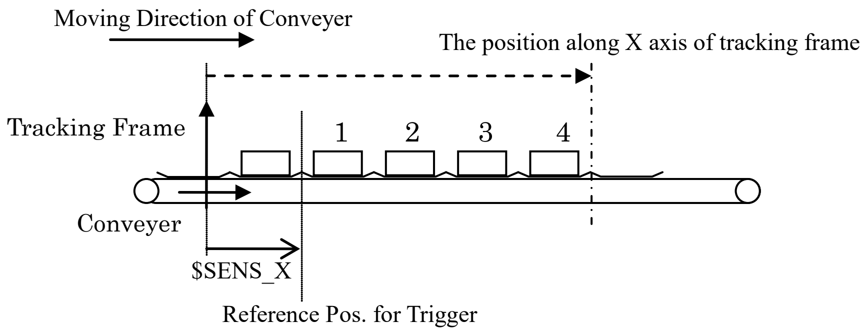



**Fig. 6.3.9(e) SLTKGTPP**

If "4" is deleted by SLTKDELQ, "3" becomes the oldest available trigger value. Therefore, the corresponding distance to the trigger value "3" is calculated by SLTKGTPP.

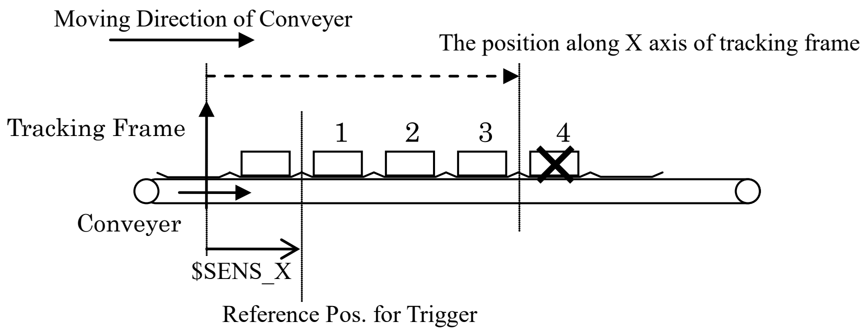



**Fig. 6.3.9(f) SLTKGTPP and SLTKDELQ**

If Discard Line is set and "4" passes the discard line, it is not possible to get the trigger value "4". In this case, the corresponding distance to the trigger value "3" is calculated by SLTKGTPP.

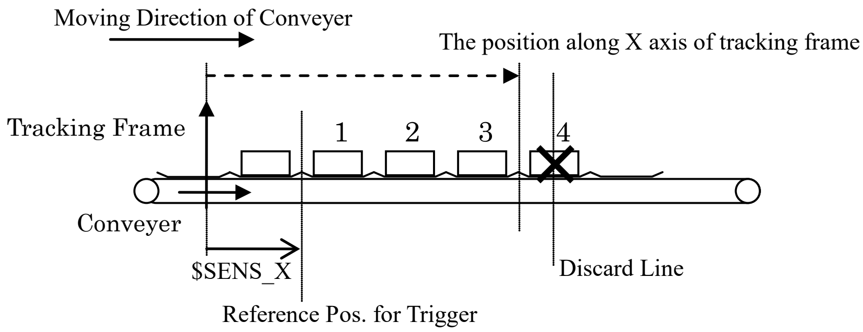


**Fig. 6.3.9(g) SLTKGTPP and Discard Line**

If Stop Conveyor is set and “4” passes the discard line, DO is out put in order to call upon to stop the conveyor. Even if “4” passes the discard line in Stop Conveyor case, it is possible to get the trigger value “4”. The corresponding distance to the trigger value “4” is calculated by SLTKGTPP.

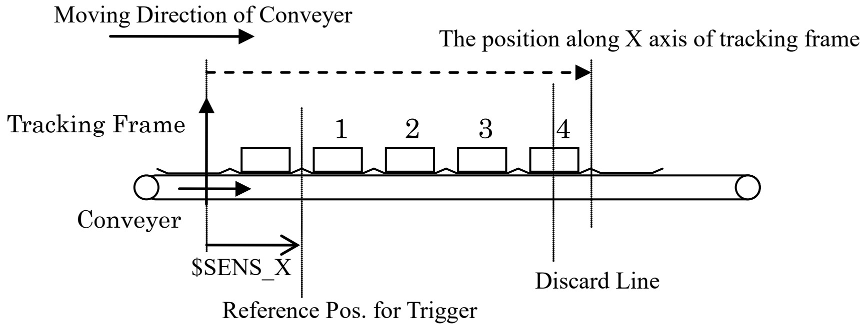



**Fig. 6.3.9(h) SLTKDELQ and Stop Conveyor**

## DISCARD LINE

If a corresponding position to the trigger value passes the discard line, it is possible to discard the trigger value. It is necessary to set Discard Line on the basis of downstream boundary of the tracking area.

It is necessary to set following system variables in order to use this function.

| | |
|---|---|
| $SLTKSCH[n].$SLQ_ENABLE: | TRUE |
| $SLTKSCH[n].$DISCARD_BND: | Specify the distance from the downstream boundary of the tracking area to the Discard Line. The unit is [mm] |
| $SLTKSCH[n].$SENS_X: | Specify the distance from the origin of Tracking frame to Reference Pos. along X axis of Tracking frame. The unit is [mm] |

“n” is the number of tracking schedule which Servo Conveyor Line Tracking uses.

If $DISCARD_BND is negative, Discard Line is set up Down-stream boundary. Normally, you should set $DISCARD_BND to the negative value. If a corresponding position to the trigger value passes the discard line, it is not possible to get the trigger value by SLTKPOPQ. The oldest saved trigger value is used for checking whether this function discard the trigger value.
In the following case, because the corresponding position to the trigger value “4” passes the discard line, “4” is discarded. After that, the corresponding position to the trigger value “3” is checked whether this function discard the trigger value.

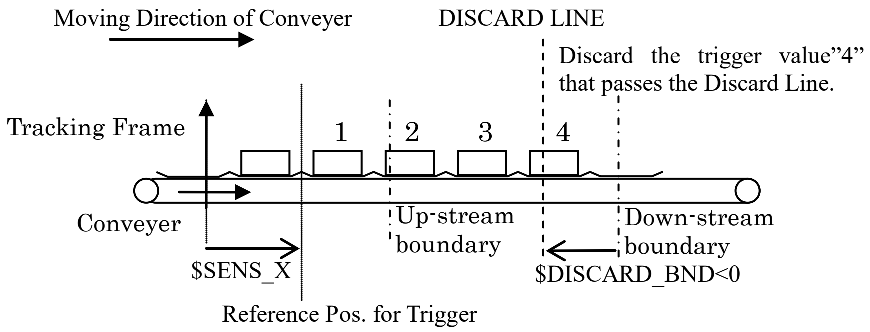



**Fig. 6.3.9(i) Discard line**

### STOP CONVEYOR

If a corresponding position to the trigger value passes Discard Line, the specified DO become ON. While the Specified DO is ON, you should not set "Index Advance Trigger DI" to ON in order to stop the conveyor. It is necessary to set Discard Line so that the desired tracking motion can be done at the conveyor stop position.

It is necessary to set following system variables in order to use this function.

| | |
|---|---|
| $SLTKSCH[n].$SLQ_ENABLE: | TRUE |
| $SLTKSCH[n].$DISCARD_BND: | Specify the distance from the downstream boundary of the tracking area to the Discard Line. The unit is [mm] |
| $SLTKSCH[n].$SENS_X: | Specify the distance from the origin of Tracking frame to Reference Pos. along X axis of Tracking frame. The unit is [mm] |
| $SLTKSCH[n].$STOPBELT: | TRUE |
| $SLTKSCH[n].$STOP_DONUM: | Specify DO number for Stop Conveyor |

"n" is the number of tracking schedule which Servo Conveyor Line Tracking uses.

When Stop Conveyor function is enabled and the corresponding position to the trigger value passes Discard Line, the trigger value is kept. While the Specified DO is ON, it is possible to get the trigger value by SLTKPOPQ. If you would like to change the specified DO from ON to OFF, it is necessary to use SLTKRSTQ. The next trigger value is used for checking the Stop Conveyor after SLTKRSTQ is executed. In the following case, because the corresponding position to the trigger value "4" passes the discard line, the specified DO becomes ON. According the DO, it is necessary to stop the conveyor. The trigger value "4" can be got by SLTKPOPQ. After a tracking motion is done at the corresponding position to the trigger value "4", the DO becomes OFF by executing SLTKRSTQ. According the DO, it is necessary to resume the conveyor. When SLTKRSTQ is executed, the corresponding position to the trigger value "3" will be checked whether this function discard the trigger value.

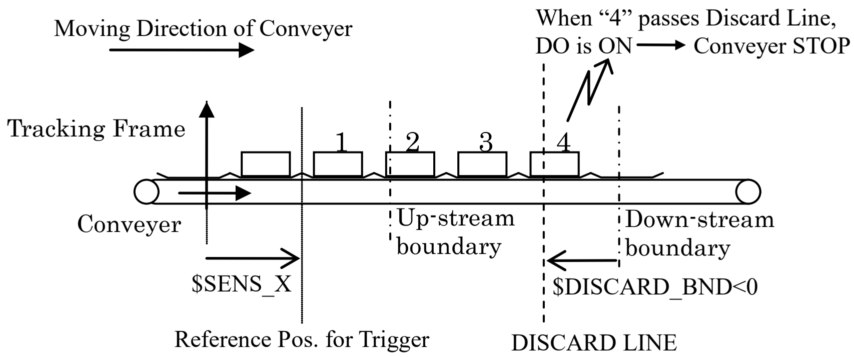



**Fig. 6.3.9(j) Stop conveyor**

> **NOTE**
> It is impossible to use both Discard Line and Stop Conveyor on the same Servo Conveyor.

### DISTRIBUTION LINE

Until a corresponding position to a trigger value passes the distribution line, it is not possible to get the trigger value with SLTKPOPQ. It is necessary to set the distribution line on the basis of upstream boundary of the tracking area.

It is necessary to set following system variables in order to use this function.

$SLTKSCH[n].$ALLOC_ENB: TRUE

$SLTKSCH[n].$ALLOC_BND: Specify the distance from the upstream boundary of the tracking area to the Distribution Line. The unit is [mm]

$SLTKSCH[n].$SENS_X: Specify the distance from the origin of Tracking frame to Reference Pos. along X axis of Tracking frame. The unit is [mm]

"n" is the number of tracking schedule which Servo Conveyor Line Tracking uses.

If $ALLOC_BND is negative, the distribution line is set up upstream boundary. Usually, you should set $ALLOC_BND to the negative value. Until a corresponding position to a trigger value passes the distribution line, it is not possible to get the trigger value with SLTKPOPQ.
In the following case, because the corresponding position to the trigger value "4" passes the distribution line, you can get the trigger value "4" with SLTKPOPQ.

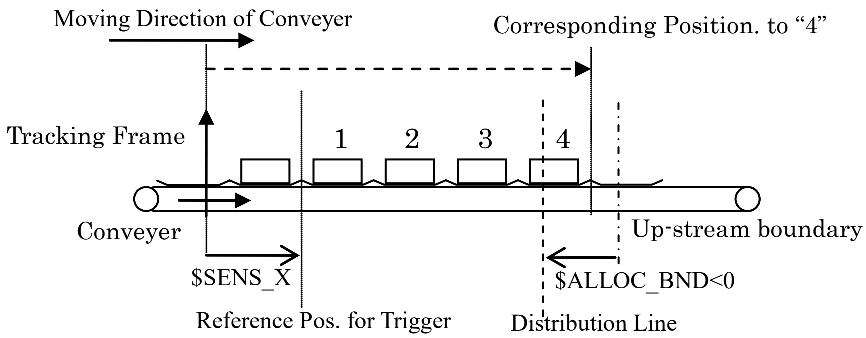



**Fig. 6.3.9(k) Distribution Line**

## Sample Program

The following are sample programs (Main Program and CONVEYOR.TP for moving Servo Conveyor) in case of using Stop Conveyor.

- The used Tracking Schedule number is "1".
- The initial position of the conveyor is saved to PR[1].
- The trigger value is saved every time the conveyor moves 2 pitches

In Main Program, the information about the trigger value is initialized by SLTKINIT at first. After that, TP program (CONVEYOR.TP) for moving Servo Conveyor is executed by the other task.

**Example 6.3.9 (a) Main Program**

```
1:J P[1] 100% FINE      ;
2:   LINE[1] ON ;
3:   CALL SLTKINIT(1,1,2) ;          Information about trigger value is initialized and PR[2] is set.
4:   RUN CONVEYOR;                   CONVEYOR.TP for moving Servo Conveyor is executed.
5:   LBL[1] ;
6:   WAIT      .10(sec) ;
7:   CALL SLTKPOPQ(1,1,2) ;          The trigger value is got.
8:   IF R[2]<>0,JMP LBL[1] ;         If it is impossible to get the trigger value, Jump to LBL[1]
9:   SETTRIG LNSCH[1] R[1] ;         R[1] is set as a trigger value.
10:   CALL TRACK      ;              TRACK.TP for tracking motion is executed.
11:   CALL SLTKRSTQ(1) ;             Turn off DO for Stop Conveyor.
12:   CALL NORM       ;              NORM.TP for non-tracking motion is executed.
13:   JMP LBL[1] ;
```

- At line 3, the information about the trigger value is initialized by SLTKINIT and PR[2] is set to the distance from current position to the position of PR[1].

- At line 4, RUN CONVEYOR.TP in order to execute the program for moving Servo Conveyor by the other task.
- At line from 5 to 8, a trigger value is got by SLTKPOPQ every 100[msec]. At line 8, whether getting a trigger value succeed is checked by R[2]. If it is impossible to get a trigger value, jump to LBL[1] at line 5.
- At line 11, if DO for Stop Conveyor is ON, turn off the DO by SLTKRSTQ.

In TP program for moving Servo Conveyor, a trigger value is got every 2 pitches by SLTKPSHQ. R[3] is used for counting the number of pitches and SLTKPSHQ is executed according to the value of R[3].

**Example 6.3.9 (b) CONVEYOR.TP**

| | |
|---|---|
| 1:J PR[2] 50% CNT0 INC ; | Movement to the initial position. |
| 2: CALL SLTKREST(1) ; | Setting of Reference Pos. for Trigger. |
| 3: IF R[60:G2 Ready DO]=0,JMP LBL[3] ; | |
| 4: DO[R[60]]=ON ; | Specified DO by R[60] become ON. |
| 5: LBL[3] ; | |
| 6: R[61:G2 cur slot ID]=0 ; | Reset R[61]. |
| 7: R[3]=0 ; | R[3] is the counter for SLTKPSHQ. |
| 8: LBL[1] ; | |
| 9: $INDEXER[1].$INDEX_MV=1 ; | Setting for waiting for DI trigger. |
| 10:J P[2] 180msec CNT100 INC ACC66 ; | One pitch movement. |
| 11: $INDEXER[1].$INDEX_MV=0 ; | Setting for waiting for DI trigger. |
| 12: R[61:G2 cur slot ID]=R[61:G2 cur slot ID]+1; | Count the number of pitches. |
| 13: R[3]=R[3]+1 ; | Count the number of pitches. |
| 14: IF R[61:G2 cur slot ID]=32,JMP LBL[2] ; | |
| 15: JMP LBL[4] ; | Jump to LBL[4] at line 19. |
| 16: LBL[2] ; | |
| 17:J PR[1] 100% CNT100 ; | Movement to PR[1] to correct iteration error. |
| 18: R[61:G2 cur slot ID]=0 ; | Reset R[61] |
| 19: LBL[4] ; | |
| 20: IF R[3]<2,JMP LBL[1] ; | Check whether conveyor moves 2 pitches. |
| 21: CALL SLTKPSHQ(1,2) ; | When conveyor moves 2 pitches, the trigger value is saved by SLTKPSHQ. |
| 22: R[3]=0 ; | Reset R[3]. |
| 23: JMP LBL[1] ; | |
| /POS | |

```
P[2:""]{
   GP2:
         UF : 0, UT : 1,
         J1=     360.000 deg
};
```

- Because PR[2] is set to the distance from current position to the position of PR[1], at line 1, the conveyor moves from the current position to the position of PR[1].
- At line 2, Reference position for Trigger is set by SLTKREST.
- R[3] is the counter for SLTKPSHQ. At line 7, 13, 22, R[3] is initialized, counted and reset.
- At line 17, the conveyor moves to PR[1] to correct iteration error. Every time the conveyor makes one revolution this line is executed.
- At line 20, the number of pitches is checked. When the conveyor moves 2 pitches, SLTKPSHQ is executed to save to a trigger value.
- If you would like to change the number of pitches per trigger. Please change “2” at line 20, 21 to the desired number.

### 6.3.10 Limitations

- This feature cannot be used at the same time as Continuous Servo Conveyor Tracking (R884)
- This option requires Line tracking option (J512) and Constant Path option (R663). Standard Setting (R651) includes Constant Path option (R663).
- It is necessary to separate a robot and an extended axis as a conveyor into different groups.
- It is possible to add up to four servo conveyor to a controller. (G1:Robot, G2-G5:Servo Conveyor)
- Servo Conveyor Line Tracking function can be used together with a traditional encoder conveyor.
- Servo Conveyor Line Tracking function supports HDI and ACCUTRIG instruction.
- It is not possible to use Visual Tracking on Servo conveyor. (It is possible to use Visual Tracking on traditional encoder conveyor.)
- The tracking robot can not use Continuous Turn function.
- Original Path Resume feature is disabled.
- It is not possible to execute TP program of which a motion mask has the tracking robot group and Servo conveyor group.
- In a tracking program, Please add the wrist joint motion instruction (Wjnt) to all motion instruction or don’t add Wjnt to motion instruction. If there are a motion instruction with Wjnt and a motion instruction without Wjnt, TCP might be deviated from a destination at tracking motion.
- In a tracking program, robots must not change tools. If the tracking robot changes tools, it may deviate from the target position during tracking.

## 6.4 CONTINUOUS SERVO CONVEYOR TRACKING

### 6.4.1 Overview

This feature allows the robot to track a “Continuous” servo conveyor that moves while switching speeds sequentially. The servo conveyor is used as an additional axis of the robot controller. Even when the conveyor is accelerating, the robot can track more accurately than normal tracking using an encoder. This function can be used with software series of 7DC3 or later. For 7DC3 series, it can be used by including J589 in the order. For 7DF1 and later series, it can be used by including R884 in the order.
Continuous Servo Conveyor Tracking option (R884 or J589) requires Line Tracking option (J512) and Independent Auxiliary Axis option (H895).
Servo Conveyor Line Tracking option (J589) includes Multi Motion Group option (J601) and Continuous Turn option (J613). These are used for keeping on moving an extended axis as a conveyor. In addition, is necessary to control extended axis. J589 and R884 cannot be ordered at the same time.

### 6.4.2 Independent extended axis setup for Continuous Servo Conveyor

Independent Extended Axis is set on the robot maintenance screen of the control start menu. For details on the setting procedure, refer to "INDEPENDENT ADDITIONAL AXIS SETUP" in Basic Function Manual (B-83284EN). When adding the Independent Extended Axis, some setting items enter values as described in order to use the additional axis as Continuous Servo Conveyor. For other setting items, refer to Basic Function Manual (B-83284EN) and set them appropriately according to the system configuration.

#### Independent axes type

Select “2 Rotary Axis” as Independent axis type.

```
-- INDEPENDENT AXES TYPE –
1. Linear Axis
2. Rotary Axis

Select? 2
```

### Gear Ratio

Enter the number of revolution of the motor which corresponds to a belt length of section of the conveyor. By this setting, a conveyor moves the belt length of section when Independent Axis moves 360 degrees.

```
-- GEAR RATIO --
Enter Gear Ratio?
```

If a conveyor rotates "N" times and the motor of it rotates "M" times, the gear ratio can be calculated by the following. Please enter the calculated value.

$$\mathrm{GearRatio} = \frac{\mathrm{M} \times \mathrm{Belt\ \ Section\ \ Teeth}}{\mathrm{N} \times \mathrm{Belt\ \ Teeth\ \ of\ \ the\ \ conveyor}}$$

- It is also possible to calculate the gear ratio from the set value (Motor Gear teeth and so on) of Servo Conveyor Setup. In this case, the gear ratio can be calculated by the following.

$$\mathrm{GearRatio} = \frac{\mathrm{Rotor\ \ Input\ \ Gear\ \ Teeth} \ \times \ \mathrm{Belt\ \ Section\ \ Teeth}}{\mathrm{Motor\ \ Gear\ \ Teeth} \ \times \ \mathrm{Rotor\ \ Output\ \ Gear\ \ Teeth}}$$

### Max Joint Speed

Select "2:NO Change" for setting a suggested speed as a max joint speed.

```
--MAX JOINT SPEED SETTING --
  Suggested Speed = 800.000(deg/s)
  (Calculated with Max Motor Speed)

Enter (1:Change, 2:No Change)? 2
```

### Motor Direction

Choose "1:TRUE" if the joint coordinate position of the conveyor increases when the motor rotates in the plus direction.
Choose "2:FALSE" if the joint coordinate position of the conveyor decreases when motor rotates in the plus direction.

```
-- MOTOR DIRECTION
INDEPENDENT AXES 1 Motion Sign = TRUE
Enter (1:TRUE, 2:FALSE)?
```

If you look at a motor from the front of the flange, a counter clockwise rotation is plus direction of a motor.

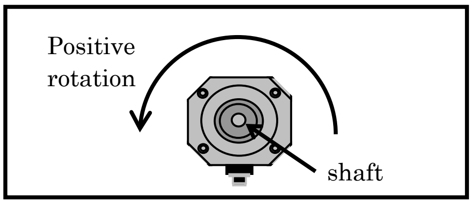



**Fig. 6.4.2 Positive rotation of a motor**

### Upper Limit and Lower Limit

Enter 180[deg] as an upper limit and –180[deg] as a lower limit.

```
-- UPPER LIMIT --
Enter Upper Limit ( deg)? 180
```

```
--LOWER LIMIIT--
Enter Lower Limit ( deg)? -180
```

### Master Position

Normally, the position is “0”.

```
--MASTER POSITION –
Enter Master Position (deg)?
```

### ACC/DEC TIME

Set the values of "the 1st ACC/DEC Time", "the 2nd ACC/DEC Time", and "Minimum Accel Time". User should consider the load on the conveyor and set a reasonable ACC/DEC time. Set "the 2nd ACC/DEC Time" to half the value of the "the 1st ACC/DEC Time". "Minimum Accel Time" is the sum of "the 1st ACC/DEC Time" and "the 2nd ACC/DEC Time".

## 6.4.3 Continuous Servo Conveyor Setup

Please set up Continuous Servo Conveyor by the following step. The setting of encoder and the setting of Continuous Turn are updated automatically by setting Servo Conveyor.

### Procedure 6-8 Continuous Servo Conveyor Setup

### Step

1. Press the MENU key.
2. Select “SETUP”.
3. Press “F1: TYPE”.
4. Select “Indexers”. You will see a screen as following.

```
FANUC Conveyor:
                                                1/4
          FANUC Conveyor:  1
     1  Encoder Number:                          1
     2  Conveyor Type:             Index
     3  Tracking Schedule:                       1
        Encoder Scale (cnt/mm):        524288
     4  program and speed control:  Detail

   [ TYPE ]              CNVY
```

5. Move the cursor to “Encoder Number” and enter the number of the encoder that is used as servo conveyor.
6. Move the cursor to “Conveyor Type” and select “Continuous”.

```
FANUC Conveyor:
                                         2/4
             FANUC Conveyor:  1
        1   r Number:                      1
1     Index  or Type:             Index
2 Continuous ng Schedule:                  1
3           r Scale (cnt/mm):        524288
4           m and speed control:  Detail
5

  [ TYPE ]   DETAIL    CNVY   [CHOICE]]
```

7. Move the cursor to “Tracking Schedule” and enter the value of the tracking schedule that is used for servo conveyor.
8. Move the cursor to “Conveyor Type” and press “F2: DETAIL” key or Enter Key. You will see a screen for setting up axis.
9. You will see a screen as following. Because there is no [TYPE] in this menu, you can not select any other setup menu. To return back to previous menu with [TYPE] press PREV key.

```
SETUP Conveyor axis
                                       1/8
          Conveyor  1: motor           DONE
         Encoder Number:    1
      1  Robot Group:                      2
              Axis:                        1
      2  Motor Gear teeth                  1
      3  Rotor input Gear teeth            1
      4  Rotor output Gear teeth           1
      5  Belt Section teeth                1
      6  Belt Section length (mm)     10.000
      7  Conveyor ON DI:                   1
      8  Flag for internal use:          133

                       EXEC
```

10. Move the cursor to “Robot Group” and enter the group number of the extended axis for the servo conveyor and then move the cursor to “Axis” and enter the axis number.
11. Move the cursor to “Motor Gear teeth”, “Rotor input Gear teeth”, “Rotor output Gear teeth” and “Belt Section teeth” and enter each value in the positive integer. Move the cursor to “Belt Section length (mm)” and enter the belt length corresponding to “Belt section teeth”.
    The following figure shows the relationship between these values. The gear ratio of the servo conveyor is calculated from these values.

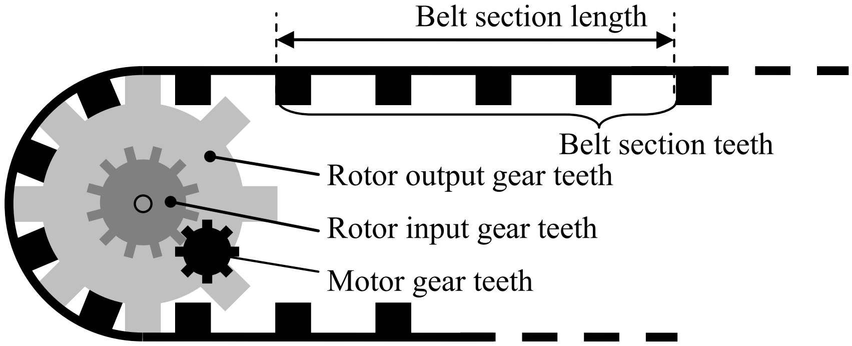



**Fig. 6.4.3 Relationship between motor, rotor and conveyor**

- It is also possible to set up from the relation between the number of revolution of motor and the number of revolution of the conveyor. If a conveyor rotates “N” times and a motor of it rotates “M” times, please setup as following. “M” and “N” must be an integer.
  a. Enter the value of “N” to “Motor Gear teeth”.
  b. Enter the value of “M” to “Rotor input Gear teeth”.
  c. Enter the number of teeth of the conveyor belt to “Rotor output Gear teeth”.
  d. Enter the number of teeth corresponding to “Belt section length” and “Belt section teeth”.

12. Move the cursor to “Conveyor ON DI” and enter the index of DI is used for moving the servo conveyor.
13. Move the cursor to “Flag for internal use” and enter the index of Flag is used for the servo conveyor. Please confirm that the specified Flag is not used for another use.
14. After the above setup, press “F2: EXEC” and be sure to Power off/on.

**NOTE**
By the above setting, Continuous Rotation setup of the conveyor and Encoder setup of the specified encoder are also done.

## 6.4.4 Continuous Servo Conveyor program

It is necessary to prepare TP program for moving the servo conveyor on tracking because the servo conveyor is setup as extended axis. By following step, the standard TP program for moving the servo conveyor is created.

### Procedure 6-9 Create Continuous Servo Conveyor program

**Step**

1. Move the cursor to “Create Index program” in “SETUP Indexers” and press “F2: DETAIL” key or ENTER key.

```
FANUC Conveyor:
                                                4/4
         FANUC Conveyor:  1
   1  Encoder Number:                  1
   2  Conveyor Type:          Continuous
   3  Tracking Schedule:               1
      Encoder Scale (cnt/mm):     524288
   4  program and speed control:  Detail

 [ TYPE ]  DETAIL  CNVY
```

You will see a screen as following. Then CNVYG*.TP (* is group number of the servo conveyor) is created automatically, if CNVYG*.TP does not exist. To recreate CNVYG*.TP, delete CNVYG*.TP and then do this again.

```
SETUP Conveyor Program
                                          1/2
                  FANUC Conveyor:  1
      1  Conveyor Speed (mm/sec)         100
      2  Group Input port for speed        0
```

2. Set "Group Input port for speed" to the GI number to enter the speed of the servo conveyor.
3. Cycle power of the controller. Then the value input to GI [] is displayed in "Conveyor Speed (mm/sec)".

**NOTE**

If "Group Input port for speed" is 0, user can directly enter the conveyor speed value in "Conveyor Speed (mm / sec)" as an integer. The value of "Conveyor Speed (mm / sec)" is set to 0 when the controller is restarted.

**NOTE**

- Do not change CNVYG*.TP directly because it is default program in the system. Please rename CNVYG*.TP and use it.
- If you would like to change the servo conveyor speed, please change the value of "Conveyor Speed (mm/sec)". It is not necessary to edit TP program for changing a conveyor speed. The specified value is used for moving a conveyor. The conveyor speed is updated even if TP program is running. If the specified value is bigger than the conveyor capability, "Speed exceed conveyor capability" is displayed and the value is not updated.

---

### Sample Program

The following is an example of CNVYG*.TP.

**Example 6.4.4 CNVYG*.TP**

```
1:     LBL[1] ;
2:     $INDEXER[1].$INDEX_MV=1 ;                  Setting for waiting for DI trigger
3:  J P[2] 48msec CNT100 INC ACC100     ;         Conveyor movement
4:     $INDEXER[1].$INDEX_MV=0 ;                  Setting for waiting for DI trigger
5:     JMP LBL[1] ;
```

- "$INDEXER[*].$INDEX_MV=1" at line 2 and "$INDEXER[*].$INDEX_MV=0" at line 4 are setting for watching input of Index Advance Trigger DI (* is number of Servo Conveyor). Motion instruction at line 3 is not executed until the DI changes from OFF to ON. When DI changes from ON to OFF, the conveyor will stop.

## 6.4.5 Tracking schedule setup

Tracking Schedule Setup for servo conveyor is the same as ordinary Line Tracking function. Please set items of the specified tracking schedule by the servo conveyor setup display. Please refer to Section 3.3 "TRACKING SETUP".

## 6.4.6 Example of main program

Basically, how to teach TP program is the same as ordinary Line tracking function. However, it is necessary to run TP program for moving a servo conveyor by multitask.
CNVEYOR.TP for moving a servo conveyor is run at line 3 in the following Sample Program.

**Example 6.4.6 Main Program**

```
1:    J P[1] 100% FINE ;
2:    LINE[1] ON ;
3:    RUN CONVEYOR;          CONVEYOR.TP for moving Servo Conveyor is executed.
4:   LBL[1] ;
5:    WAIT DI[2]=ON     ;
6:    LINECOUNT[1] R[1] ;
7:    SETTRIG LNSCH[1] R[1] ;
8:    CALL TRACK     ;                TRACK.TP for tracking motion is executed.
9:    CALL NORM      ;                NORM.TP for non-tracking motion is executed.
10:   JMP LBL[1] ;
```

## 6.4.7 Error tune variable

On Servo Conveyor Line Tracking, adjust an error of the tracking motion by using the following parameter. Refer to Section 5.6 for how to adjust this parameter.

- Static Tuning Variable (I/O Delay):
  $ENC_IO_DLYF[encoder number]

- Dynamic Tuning Variable (Servo Delay):
  $SLTK_GRP[number of servo conveyor group].$SRVO_DELAY

- Acceleration Compensation Variable:
  $ACCCMPSCALE[encoder number]

Acceleration compensation variables affect only when the conveyor is accelerating or decelerating. After completing the static and dynamic error adjustments, adjust $ACCCMPSCALE[ ] (default value: 1.0) as follows:

1. Run a chosen point on the conveyor past the part detect at any speed and set the trigger.
2. After the part detect is complete, run the conveyor until your chosen point on the conveyor, at which the trigger was set, is in the robot working area.
3. Create a new tracking program and teach a position, P[1]; the desired target point on the conveyor.
4. Using the WAIT instruction, add a long wait to the program, or wait for some user input (Flag or Register). For example your program might look something like this:

```
L P[1] 500 mm/sec FINE
WAIT 30.00 (sec)
```

5. After creating the program, the next step is to run the program with the conveyor stopped.
6. Start the conveyor or change the speed while the robot continues tracking in WAIT instruction to confirm the tracking error while the conveyor is accelerating and decelerating.
7. If lagging on the Acceleration side, increase the value; if leading on the Acceleration side, decrease the value.
8. Repeat Step 1 through Step 7 until the robot tracks with sufficient accuracy.

> **NOTE**
> If it is difficult to visually confirm the error, it is recommended to measure the error using an external device such as a camera.

## 6.4.8 Limitations

- This feature cannot be used at the same time as Servo Conveyor Line Tracking option (J589)
- This option requires Line tracking option (J512) and Constant Path option (R663). Standard Setting (R651) includes Constant Path option (R663).
- It is necessary to separate a robot and an extended axis as a conveyor into different groups.
- It is possible to add up to four servo conveyor to a controller. (G1:Robot, G2-G5:Servo Conveyor)
- Servo Conveyor Line Tracking function can be used together with a traditional encoder conveyor.
- Servo Conveyor Line Tracking function supports HDI and ACCUTRIG instruction.
- It is not possible to use Visual Tracking on Servo conveyor. (It is possible to use Visual Tracking on traditional encoder conveyor.)
- The tracking robot can not use Continuous Turn function.
- Original Path Resume feature is disabled.
- It is not possible to execute TP program of which a motion mask has the tracking robot group and Servo conveyor group.
- In a tracking program, Please add the wrist joint motion instruction (Wjnt) to all motion instruction or don’t add Wjnt to motion instruction. If there are a motion instruction with Wjnt and a motion instruction without Wjnt, TCP might be deviated from a destination at tracking motion.
- In a tracking program, robots must not change tools. If the tracking robot changes tools, it may deviate from the target position during tracking.

# APPENDIX

# A TRACKING ACCURACY

Tracking accuracy is the maximum tracking error offset you can expect in an application. The tracking accuracy is a function of synchronizing the part detects switch and the encoder-read/set-trigger operation.
The part might trip the part detect switch (trigger) at any time, and not be synchronized with the controller interpolation cycle in any way. The controller will detect this within one ITP time. If ACCUTRIG is used, the controller will detect this within one system tick (2 ms). Refer to Section 4.3.
However, even under the best of conditions it might take up to one additional ITP time to read and store the value of the encoder (to perform the set trigger operation). The value that is read is the value stored during the last encoder data update.
Therefore, there can be synchronization delays of up to 2 ITP times for this operation; and more if the encoder update time multiplier is set larger than 1. The maximum error can be computed using the following equation.

***ErrorDist.*(mm) = *Conv.Speed*(mm/sec) * 2 * *ITPtime*(sec)**

**For example:**
**3.2(mm) = 200(mm/sec) *2 * 8(msec)**

When ACCUTRIG is used, the system tick would be recorded when the part detect switch is triggered. The system then finds the encoder value at that system tick. Therefore, the formula is

***ErrorDist.*(mm) = *Conv.Speed*(mm/sec) * 2 * *SystemTick*(msec)**

**For example:**
**0.8(mm) = 200(mm/sec) *2 * 2(msec)**

> **NOTE**
> The larger ErrorDist value will always be used.

> **NOTE**
> PaintTool disables the ACCUTRIG tracking accuracy feature.

The tracking accuracy can also be limited by the resolution of the encoder being used, and any gear ratio associated with it. The resolution is a combination of the encoder scale factor and the conveyor speed. The resolution can be computed using the following equation.

***Resolution*(pulses/update) = *ScaleFactor*(pulses/mm) * *Conv.Speed*(mm/sec) * *UpdateTime*(sec/update)**

For example, in a system with an encoder scale factor of 25 pulses/mm and an ITP time (update time) of 8 msec, the resolution for a conveyor speed of 200 mm/sec is:

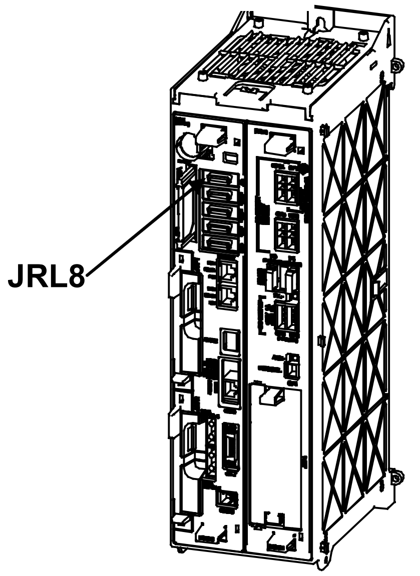
**40(pluses/update) = 25(pulses/mm) * 200(mm/sec) * 0.008(sec/update)**

# B SCHEMATICS

## B.1 OVERVIEW

This section contains the schematic drawings of cables used for the HDI interface and line tracking encoders. The HDI signals are used in combination with special application software. The HDI signals cannot be used as general-purpose DIs.

**R-30*i*B / R-30*i*B Plus**

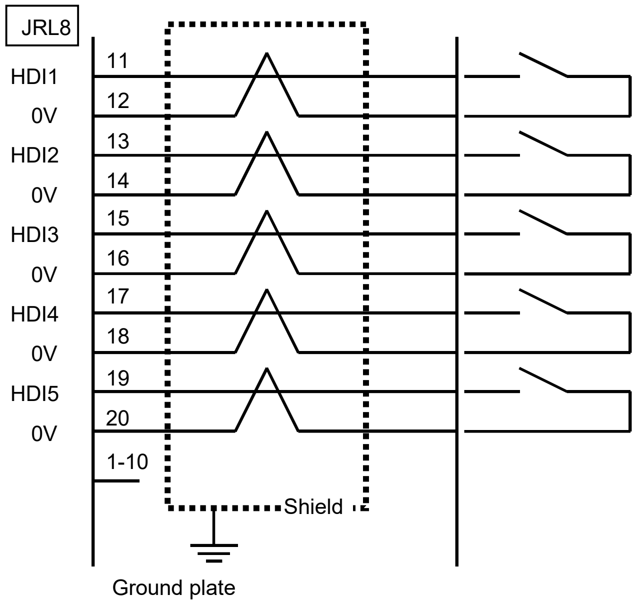



R-30*i*B / R-30*i*B Plus

Main board
JRL8

| | | | |
|---|---|---|---|
| 1 | Reserved | 11 | HDI1 |
| 2 | Reserved | 12 | 0V |
| 3 | Reserved | 13 | HDI2 |
| 4 | Reserved | 14 | 0V |
| 5 | Reserved | 15 | HDI3 |
| 6 | Reserved | 16 | 0V |
| 7 | Reserved | 17 | HDI4 |
| 8 | Reserved | 18 | 0V |
| 9 | Reserved | 19 | HDI5 |
| 10 | Reserved | 20 | 0V |

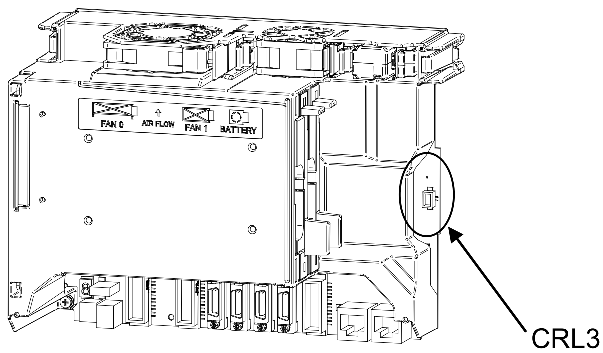
**Fig. B.1 (a) R-30*i*B / R-30*i*B Plus HDI interface**




Recommended cable connector:
PCR-E20FA (Honda Tsushin Kogyo)
FI30-20S (Hirose Electric)
FCN-247J020-G/E (Fujitsu)
52622-2011 (Molex Japan)

**Fig. B.1 (b) R-30*i*B / R-30*i*B Plus HDI cable connections**

## R-30*i*B Mate / R-30*i*B Mate Plus



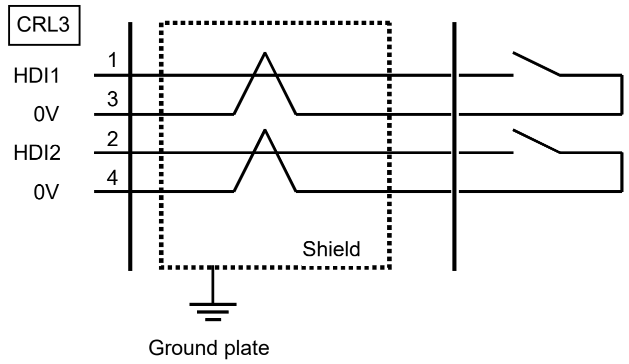

R-30*i*B Mate

Main board

CRL3

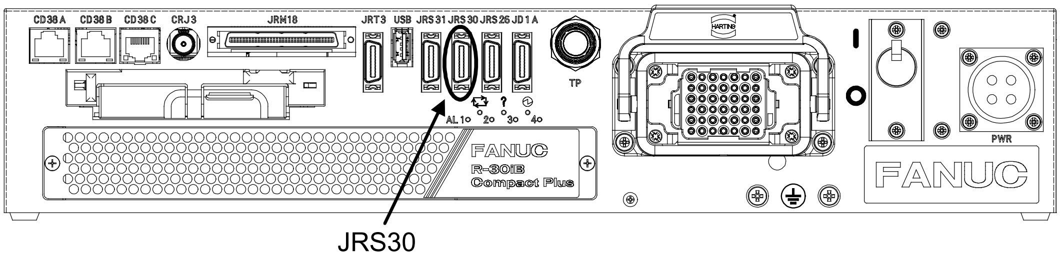
| | | | |
|---|---|---|---|
| 1 | HDI1 | 2 | HDI2 |
| 3 | 0V | 4 | 0V |

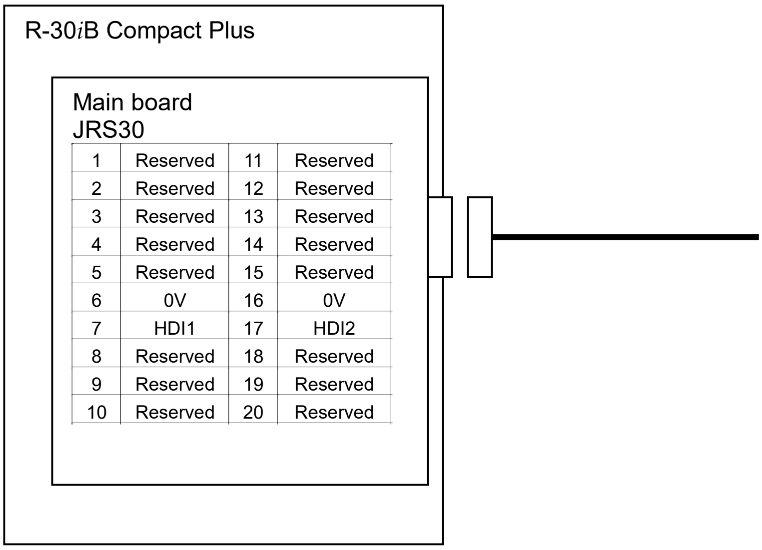
**Fig. B.1 (c) R-30*i*B Mate / R-30*i*B Mate Plus HDI interface**




Recommended cable connector:
DF11-4DS-2C (Hirose Electric)

**Fig. B.1 (d) R-30*i*B Mate / R-30*i*B Mate Plus HDI cable connections**




| | | | |
|---|---|---|---|
| 1 | Reserved | 11 | Reserved |
| 2 | Reserved | 12 | Reserved |
| 3 | Reserved | 13 | Reserved |
| 4 | Reserved | 14 | Reserved |
| 5 | Reserved | 15 | Reserved |
| 6 | 0V | 16 | 0V |
| 7 | HDI1 | 17 | HDI2 |
| 8 | Reserved | 18 | Reserved |
| 9 | Reserved | 19 | Reserved |
| 10 | Reserved | 20 | Reserved |

**Fig. B.1 (e) R-30*i*B Compact Plus HDI interface**

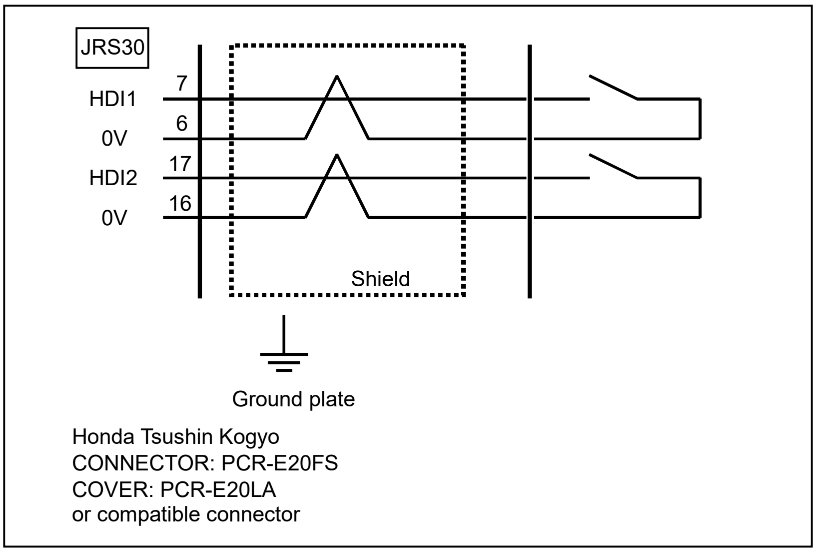
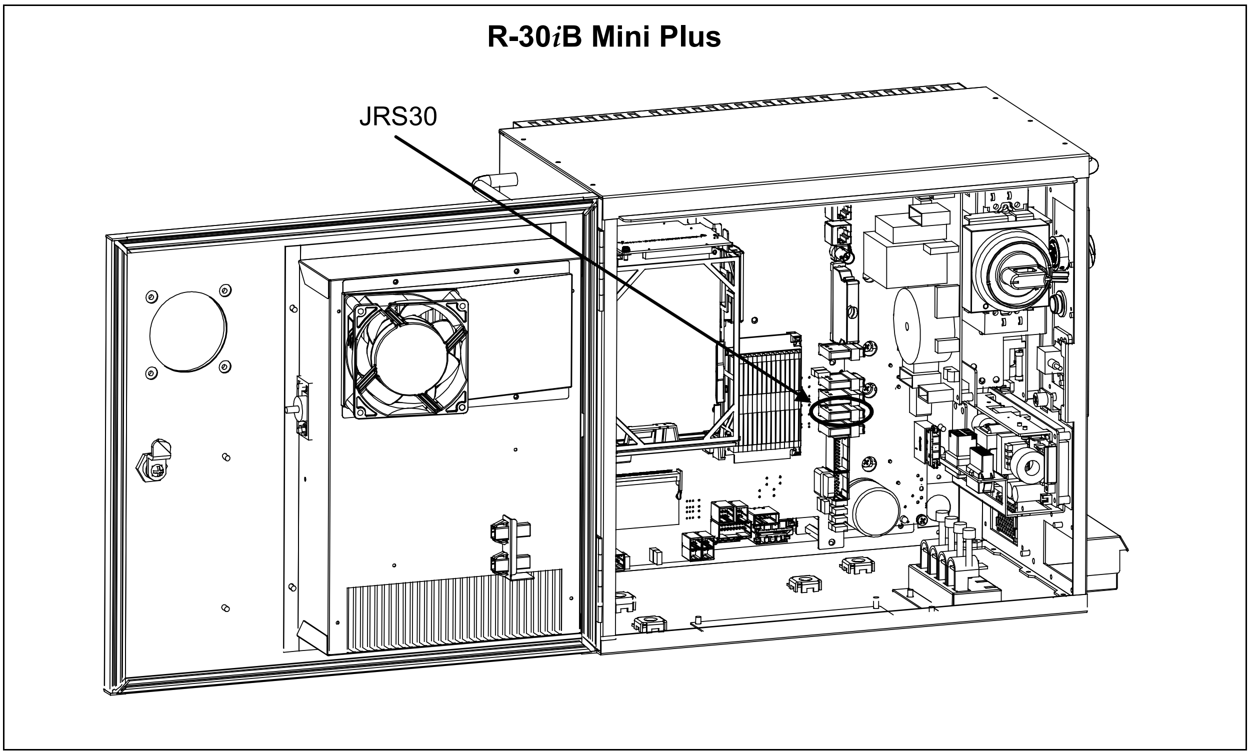
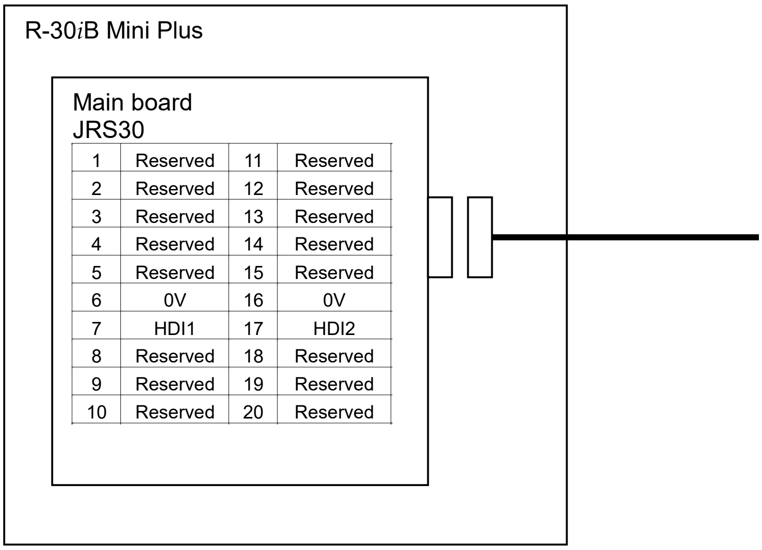



**Fig. B.1 (f) R-30*i*B Compact Plus HDI cable connections**






| | | | |
|---|---|---|---|
| 1 | Reserved | 11 | Reserved |
| 2 | Reserved | 12 | Reserved |
| 3 | Reserved | 13 | Reserved |
| 4 | Reserved | 14 | Reserved |
| 5 | Reserved | 15 | Reserved |
| 6 | 0V | 16 | 0V |
| 7 | HDI1 | 17 | HDI2 |
| 8 | Reserved | 18 | Reserved |
| 9 | Reserved | 19 | Reserved |
| 10 | Reserved | 20 | Reserved |

**Fig. B.1 (g) R-30*i*B Mini Plus HDI interface**

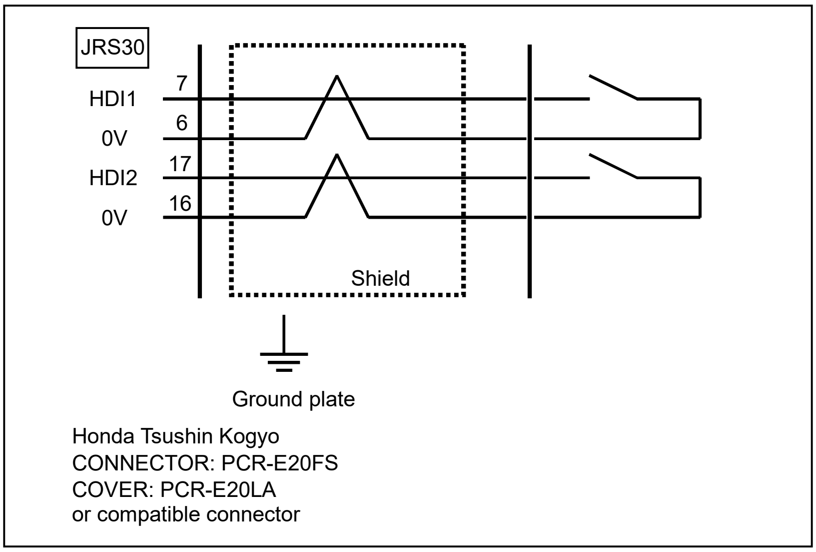
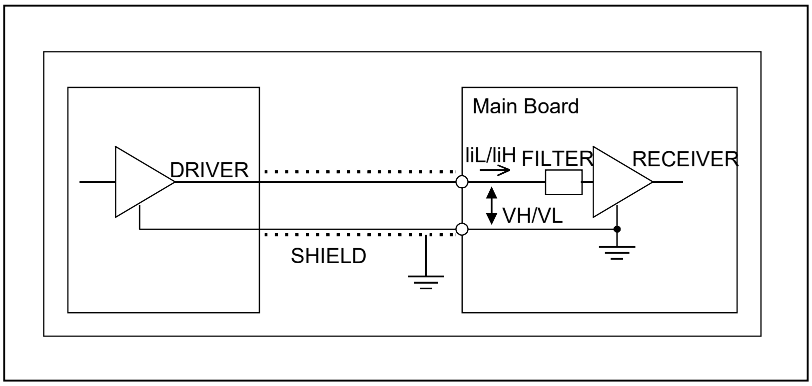



**Fig. B.1 (h) R-30*i*B Mini Plus HDI cable connections**




Absolute maximum rating
Input voltage range Vin: -3.6 to +10 V
Input characteristics

| Unit | Symbol | Specification | Unit | Remark |
|---|---|---|---|---|
| High level input voltage | VH | 3.6 to 11.6 | V | |
| Low level input voltage | VL | 0 to 1.0 | V | |
| High level input current | IiH | 2 max | mA | Vin=5 V |
| | | 11 max | mA | Vin = 10 V |
| Low level input current | IiL | -8.0 max | mA | Vin = 0 V |
| Input signal pulse duration | | 20 min | *m* s | |
| Input signal delay or variations | | 0.02(max) | ms | |

**NOTE**

1 The plus (+) sign of IiH/IiL represents the direction of flow into the receiver. The minus (-) sign of IiH/IiL represents the direction of flow out of the receiver.
2 The high-speed skip signal is assumed to be 1 when the input voltage is at the low level and 0 when it is at the high level.

**Fig. B.1 (i) Input signal rules for the high-speed skip (HDI)**

**NOTE**

In R-30iB Mate controller, HDI interface is not available with Main CPU board A (A20B-8200-0790), Main CPU board B (A20B-8200-0791), Main CPU board D (A20B-8201-0420) or Main CPU board E (A20B-8201-0421). Main CPU board C (A20B-8200-0792) or Main CPU board F (A20B-8201-0422) is required to enable HDI for Line Tracking feature with R-30iB Mate controller. See Table 5.6.2 (a) for detail.
In R-30*i*B Mate Plus controller, HDI interface is not available with Main CPU board A (A20B-8201-0750) or Main CPU board B (A20B-8201-0751). Main CPU board C (A20B-8201-0752) is required to enable HDI for Line Tracking feature with R-30*i*B Mate Plus controller. See Table 5.6.2 (a) for detail.
In R-30*i*B Compact Plus controller, HDI interface is not available with Main CPU board A (A17B-8100-0800). Main CPU board B (A17B-8100-0801) is required to enable HDI for Line Tracking feature with R-30*i*B Compact Plus controller. See Table 5.6.2 (b) for detail.

# C PULSECODER A860-0301-T001 to T004

## C.1 REQUIREMENTS

**Table C.1 (a) Requirements (R-30*i*B / R-30*i*B Plus: Incremental Pulsecoder A860-0301-T001 to T004)**

<table>
<tr><th>Required Components</th><th>R-30iB / R-30iB Plus Controller A-Cabinet</th><th>R-30iB / R-30iB Plus Controller B-Cabinet</th><th rowspan="2">Comments</th></tr>
<tr><td colspan="3">Hardware</td></tr>
<tr><td>Line Tracking Interface Board</td><td>A20B-8101-0421<br>(wide-mini slot)<br>or<br>A20B-8101-0601<br>(mini slot)</td><td>A20B-8101-0421<br>(wide-mini slot)<br>or<br>A20B-8101-0601<br>(mini slot)</td><td>• The Line tracking interface board in the left column is included in the following order specification.<br>A05B-2600-J035, A05B-2660-J035 (A/B-Cabinet, wide-mini slot)<br>A05B-2600-J036, A05B-2660-J036 (A-Cabinet, mini slot)<br>A05B-2600-J037 (B-Cabinet, mini slot)<br><br>• Separate Detector Unit (SDU) - A02B-0323-C205 can be used in place of Line tracking Interface board.<br>NOTE: SDU requires retrofit work to mount in the container. (See Fig. 2.1.1 (f) to Fig. 2.1.1 (m).)</td></tr>
<tr><td>Fiber Optic (FSSB) Cable</td><td>A66L-6001-0023</td><td>A66L-6001-0023</td><td>• The Fiber Optic cable in the left column is included in the following order specification.<br>A05B-2600-J035, A05B-2660-J035 (A/B-Cabinet, wide-mini slot)<br>A05B-2600-J036, A05B-2660-J036 (A-Cabinet, mini slot)<br>A05B-2600-J037 (B-Cabinet, mini slot)</td></tr>
</table>

| **Required Components** | **R-30*i*B / R-30*i*B Plus Controller A-Cabinet** | **R-30*i*B / R-30*i*B Plus Controller B-Cabinet** | **Comments** |
|---|---|---|---|
| Hardware | | | |
| Pulsecoder Cable | for A20B-8101-0421<br>(wide-mini slot) :<br>A05B-2601-J380 (7M)<br>A05B-2601-J381 (14M)<br>A05B-2601-J382 (20M)<br>A05B-2601-J383 (30M)<br>A05B-2660-J380 (7M)<br>A05B-2660-J381 (14M)<br>A05B-2660-J382 (20M)<br>A05B-2660-J383 (30M) | for A20B-8101-0421<br>(wide-mini slot) :<br>A05B-2603-J380 (7M)<br>A05B-2603-J381 (14M)<br>A05B-2603-J382 (20M)<br>A05B-2603-J383 (30M) | • Incremental Pulsecoder A860-0301-T001 to T004 cannot be connected to encoder terminal on Main CPU board. |
| | for A20B-8101-0601<br>(mini slot)<br>(one Pulsecoder) :<br>A05B-2601-J370 (7M)<br>A05B-2601-J371 (14M)<br>A05B-2601-J372 (20M)<br>A05B-2601-J373 (30M)<br>A05B-2660-J370 (7M)<br>A05B-2660-J371 (14M)<br>A05B-2660-J372 (20M)<br>A05B-2660-J373 (30M) | for A20B-8101-0601<br>(mini slot)<br>(one Pulsecoder) :<br>A05B-2603-J370 (7M)<br>A05B-2603-J371 (14M)<br>A05B-2603-J372 (20M)<br>A05B-2603-J373 (30M) | |
| Multiplexer cable | for A20B-8101-0421<br>(wide-mini slot) :<br>A05B-2601-J385 (7M)<br>A05B-2601-J386 (14M)<br>A05B-2601-J387 (20M)<br>A05B-2601-J388 (30M)<br>A05B-2660-J385 (7M)<br>A05B-2660-J386 (14M)<br>A05B-2660-J387 (20M)<br>A05B-2660-J388 (30M) | for A20B-8101-0421<br>(wide-mini slot) :<br>A05B-2603-J385 (7M)<br>A05B-2603-J386 (14M)<br>A05B-2603-J387 (20M)<br>A05B-2603-J388 (30M) | • When using multiple robots to track parts on the conveyor, use a Multiplexer to Controller cable, or use Ethernet encoder function (A05B-2600-R762) (option). |
| | for A20B-8101-0601<br>(mini slot) :<br>A05B-2601-J375 (7M)<br>A05B-2601-J376 (14M)<br>A05B-2601-J377 (20M)<br>A05B-2601-J378 (30M)<br>A05B-2660-J375 (7M)<br>A05B-2660-J376 (14M)<br>A05B-2660-J377 (20M)<br>A05B-2660-J378 (30M) | for A20B-8101-0601<br>(mini slot) :<br>A05B-2603-J375 (7M)<br>A05B-2603-J376 (14M)<br>A05B-2603-J377 (20M)<br>A05B-2603-J378 (30M) | |

**Table C.1 (b) Requirements (R-30*i*B Mate / R-30*i*B Mate Plus: Incremental Pulsecoder A860-0301-T001 to T004)**

| Required Components | R-30*i*B Mate / R-30*i*B Mate Plus Controller | R-30*i*B Mate / R-30*i*B Mate Plus Controller (Open Air) | Comments |
|---|---|---|---|
| Hardware | | | |
| Line Tracking Interface Board | A20B-8101-0601 (mini slot) | A20B-8101-0601 (mini slot) | • The Line tracking interface board in the left column is included in the following order specification.<br>A05B-2650-J035, A05B-2661-J035 (R-30*i*B Mate / R-30*i*B Mate Plus, mini slot)<br>A05B-2655-J035, A05B-2662-J035 (R-30*i*B Mate / R-30*i*B Mate Plus (Open Air), mini slot) |
| Fiber Optic (FSSB) Cable | A66L-6001-0026 | A66L-6001-0023 | • The Fiber Optic cable in the left column is included in the following order specification.<br>A05B-2650-J035, A05B-2661-J035 (R-30*i*B Mate / R-30*i*B Mate Plus, mini slot)<br>A05B-2655-J035, A05B-2662-J035 (R-30*i*B Mate / R-30*i*B Mate Plus (Open Air), mini slot) |
| Pulsecoder Cable | for A20B-8101-0601 (mini slot) (one Pulsecoder) :<br>A05B-2650-J200 (7M)<br>A05B-2650-J201 (14M)<br>A05B-2650-J202 (20M)<br>A05B-2661-J200 (7M)<br>A05B-2661-J201 (14M)<br>A05B-2661-J202 (20M) | for A20B-8101-0601 (mini slot) (one Pulsecoder) :<br>A05B-2655-J200 (7M)<br>A05B-2655-J201 (14M)<br>A05B-2655-J202 (20M)<br>A05B-2662-J200 (7M)<br>A05B-2662-J201 (14M)<br>A05B-2662-J202 (20M) | • Incremental Pulsecoder A860-0301-T001 to T004 cannot be connected to encoder terminal on Main CPU board. |
| | for A20B-8101-0601 (mini slot) (two Pulsecoders) :<br>A05B-2650-J210 (7M)<br>A05B-2650-J211 (14M)<br>A05B-2650-J212 (20M)<br>A05B-2661-J210 (7M)<br>A05B-2661-J211 (14M)<br>A05B-2661-J212 (20M) | A20B-8101-0601 (mini slot) (two Pulsecoders) :<br>A05B-2655-J210 (7M)<br>A05B-2655-J211 (14M)<br>A05B-2655-J212 (20M)<br>A05B-2662-J210 (7M)<br>A05B-2662-J211 (14M)<br>A05B-2662-J212 (20M) | |

| Required Components | R-30*i*B Mate / R-30*i*B Mate Plus Controller | R-30*i*B Mate / R-30*i*B Mate Plus Controller (Open Air) | Comments |
|---|---|---|---|
| Hardware | | | |
| Multiplexer Cable | for A20B-8101-0601 (mini slot) (one Pulsecoder) : A05B-2650-J225 (7M) A05B-2650-J226 (14M) A05B-2650-J227 (20M) A05B-2650-J228 (30M) | - | • When using multiple robots to track parts on the conveyor, use a Multiplexer to Controller cable, or use Ethernet encoder function (A05B-2600-R762) (option). |
| | for A20B-8101-0601 (mini slot) (two Pulsecoders) : A05B-2650-J230 (7M) A05B-2650-J231 (14M) A05B-2650-J232 (20M) A05B-2650-J233 (30M) | - | |

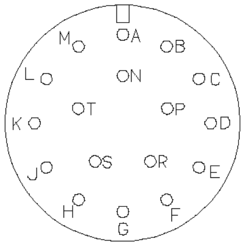
**Table C.1 (c) Requirements (R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus: Multiplexer A16B-1212-0290)**

| Required Components | R-30*i*B / R-30*i*B Mate / R-30*i*B Plus / R-30*i*B Mate Plus Controller | Comments |
|---|---|---|
| Hardware | | |
| Multiplexer | A16B-1212-0290 | • When using multiple robots to track parts on the conveyor, use a Multiplexer to Controller cable, or use Ethernet encoder function (A05B-2600-R762) (option). |
| Pulsecoder to Multiplexer Cable | A05B-2451-K102(7M) A05B-2451-K103(14M) | • |
| Multiplexer to Multiplexer Cable | A05B-2600-K155 (7M) A05B-2600-K156(14M) A05B-2660-K155 (7M) A05B-2660-K156(14M) | • An investigation is necessary if three or more Multiplexers are connected in series. |
| 5V Multiplexer Connector | A02B-0061-K203 | • One housing and six contacts(soldering type) are included. |
| 5V DC power supply (rated power 10W or more) is necessary when a Multiplexer is used. | | |

Refer to Fig. C.2 (a) to Fig. C.2 (c) for Pulsecoder signal information, and images containing the dimensions of the Pulsecoders.
Refer to Table C.2 (a) and Fig. C.2 (d) for Multiplexer specifications, and images containing the dimensions of the multiplexers.
Refer to Fig. C.3 (a) to Fig. C.3 (g) for information on dimensions, connections, and installation of the Detector Interface Units.

# C.2 FIGURES

A860-0301-T001




3102A20-29P

| Signal Name | Pin No. 310A20-29P |
|---|---|
| A | A |
| *A | D |
| B | B |
| *B | E |
| Z | F |
| *Z | G |
| C1 | - |
| C2 | - |
| C4 | - |
| C8 | - |
| +5V | C,J,K |
| 0V | N,P,T |
| Shield | H |
| OH1 | |
| OH2 | |
| REQ | |
| +6VA | |
| 0VA | |

**Fig. C.2 (a) Incremental Pulsecoder (A860-0301-T001) connection signal information**

| Item | | Specification |
|---|---|---|
| Power voltage | | 5 (V) 5% |
| Current consumption | | Up to 0.35 (A) |
| Working temperature range | | 0 to +60 (deg C) |
| Maximum response frequency | | 100x $10^{3}$ (Hz) |
| Input shaft inertia | | Up to 5x $10^{-3}$ (kg·$m^{2}$) |
| Input shaft startup torque | | Up to 0.8 (Nm) |
| Rated loads | Radial | 20 (N) |
| | Axial | 10 (N) |
| Shaft diameter runout | | 0.02x $10^{-3}$ (m) |
| Weight | | Approx. 2.0 (kg) |

**Fig. C.2 (b) Incremental Pulsecoder (A860-0301-T001) specifications**

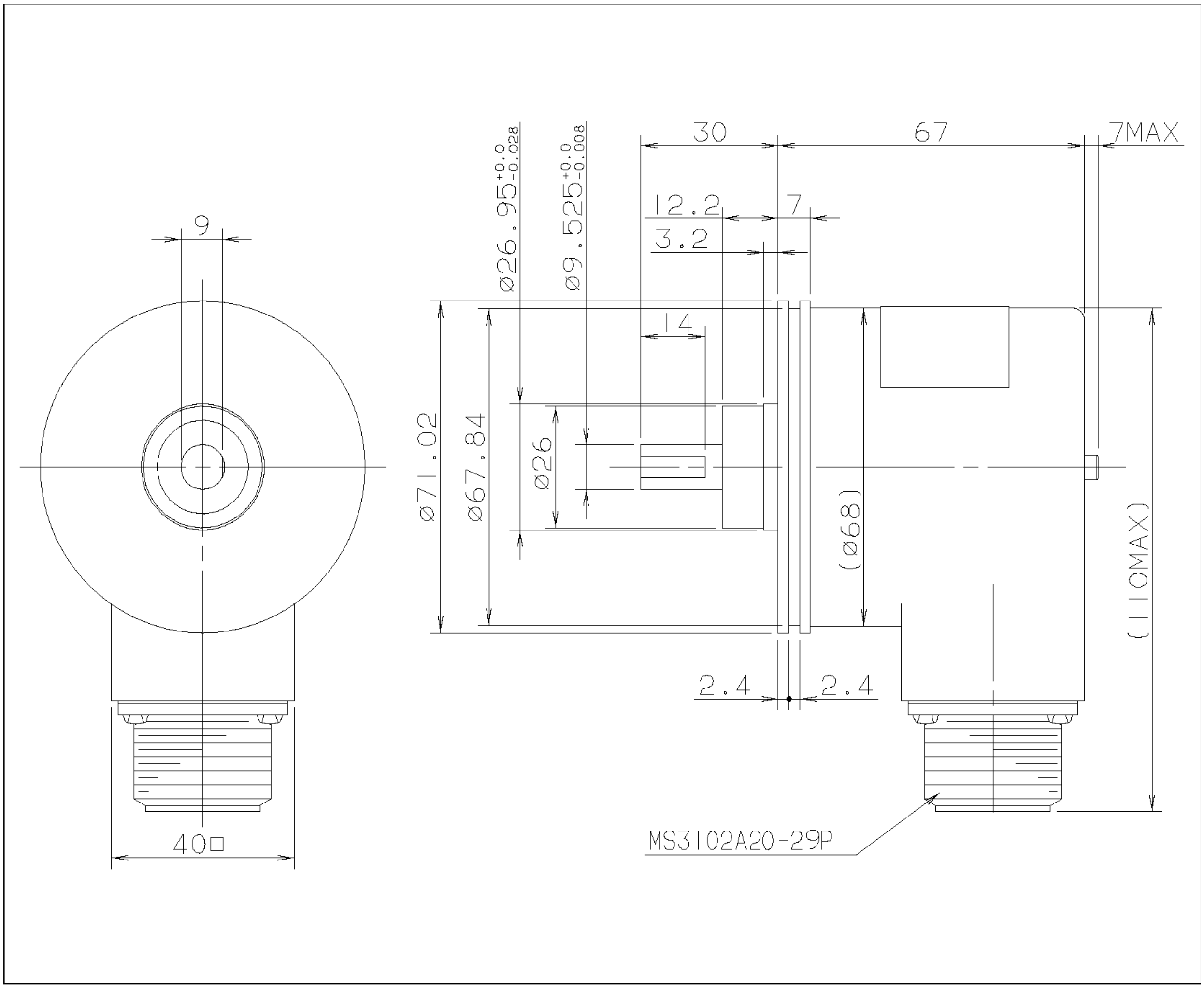



**Fig. C.2 (c) Incremental Pulsecoder dimensions (A860-0301-T001)**

**Table C.2 (a) Installation conditions of multiplexer (A16B-1212-0290)**

| Item | Condition |
|---|---|
| Permissible ambient temperature | Operating 0°C to 55°C<br>Storage, Transport -20°C to +60°C |
| Permissible ambient humidity | 95%RH or less, no condensation |
| Vibration acceleration | 4.9m/s$^2$ (0.5G) or less. |
| Atmosphere | Multiplexers must be installed in a cabinet to keep from conductive dust, cutting fluid and organic solvent. |

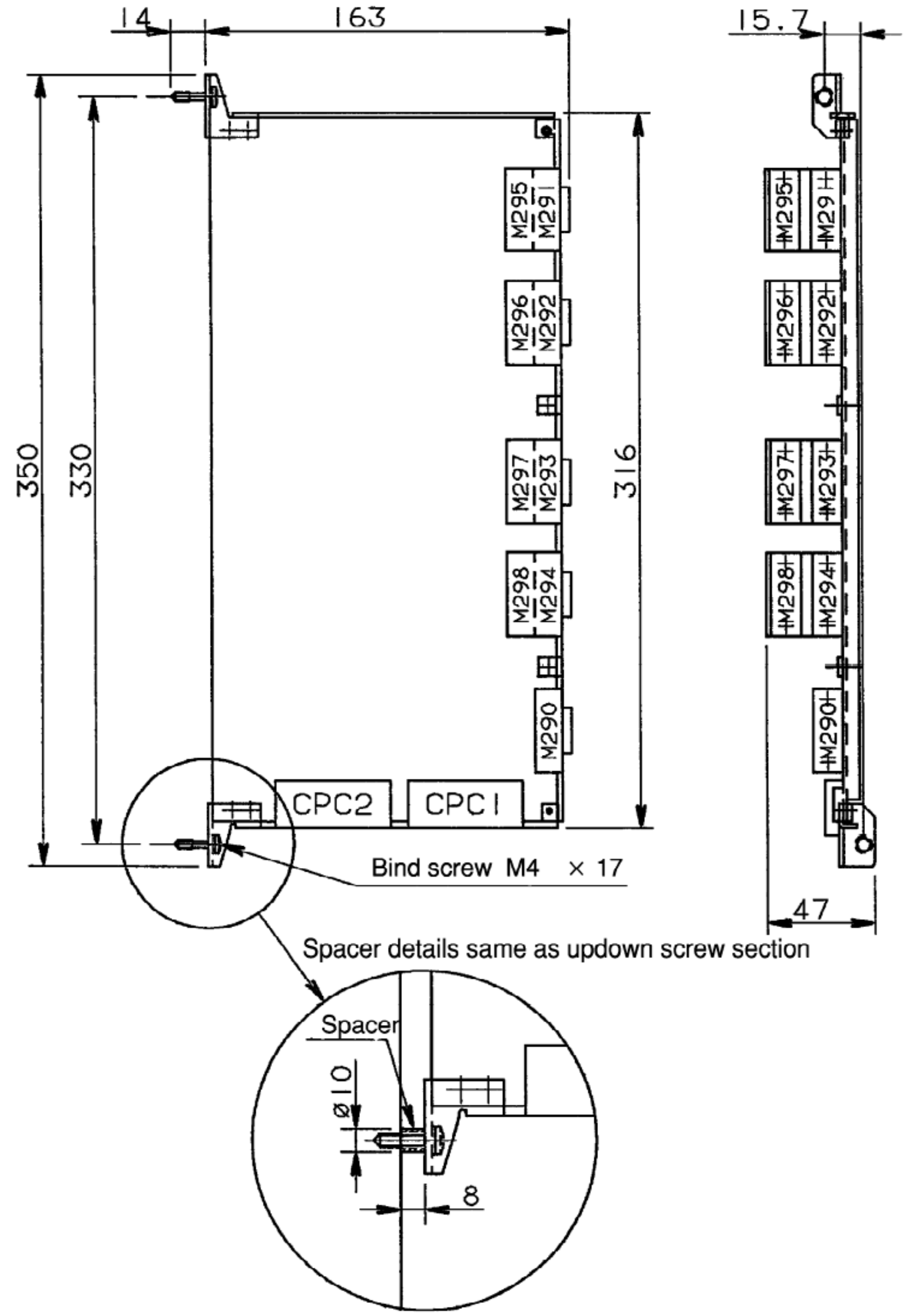



**Fig. C.2 (d) Multiplexer dimensions (A16B-1212-0290)**

# C.3 HOW TO CONNECT

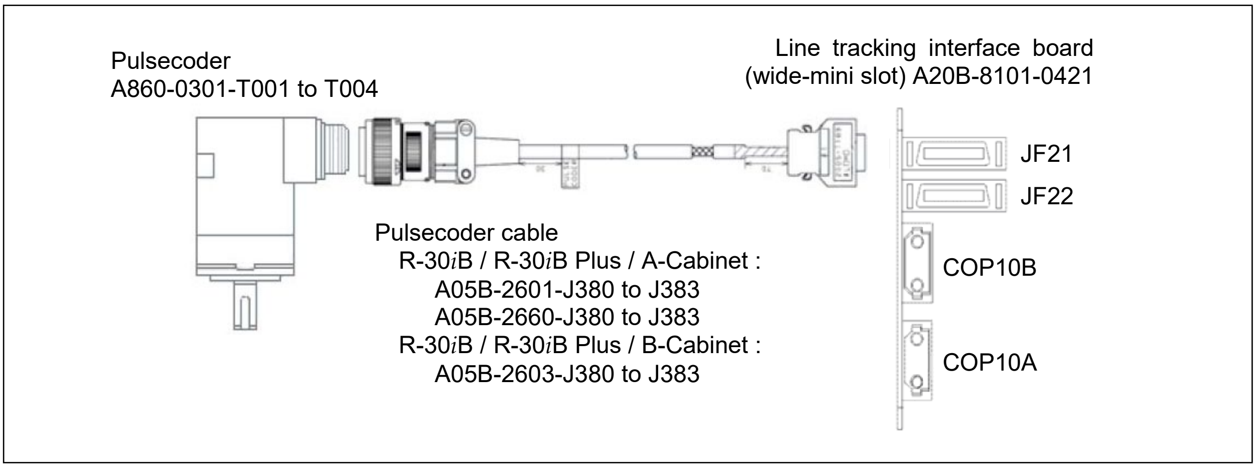



**Fig. C.3 (a) Connecting cables with Line tracking interface board A05B-2600-J035, A05B-2660-J035 (one Pulsecoder)**

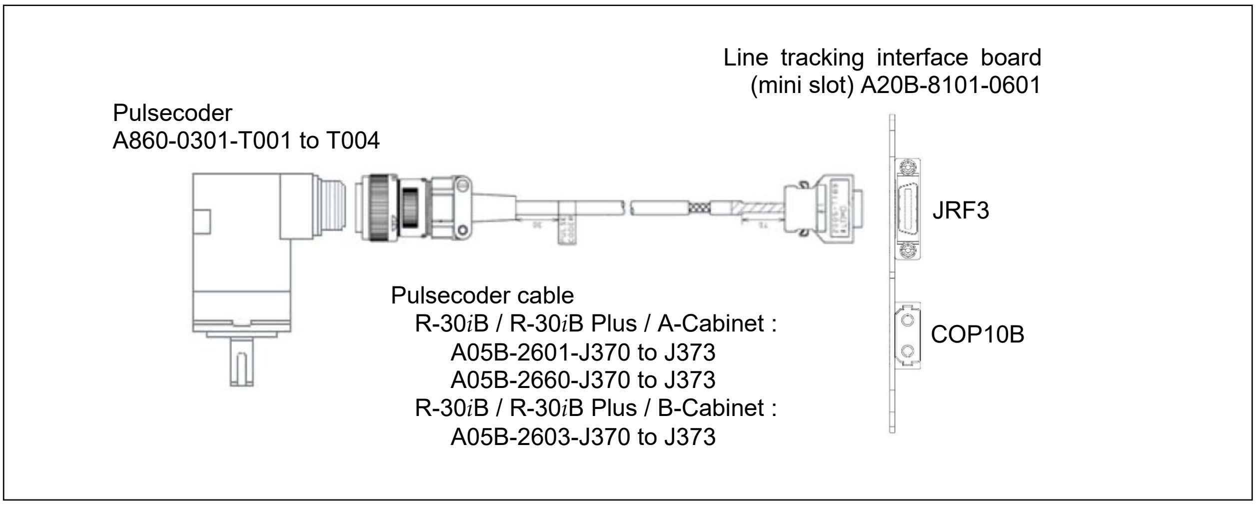



**Fig. C.3 (b) Connecting cables with Line tracking interface board A05B-2600-J036, A05B-2660-J036 or A05B-2600-J037 (one Pulsecoder)**

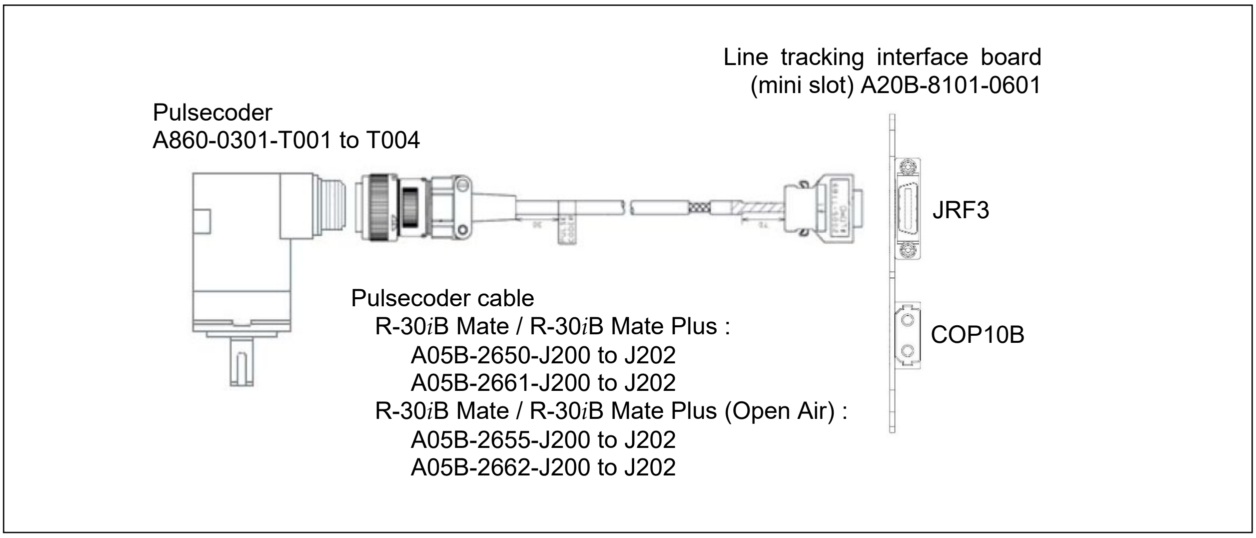



**Fig. C.3 (c) Connecting cables with Line tracking interface board A05B-2650-J035, A05B-2655-J035, A05B-2661-J035 or A05B-2662-J035 (one Pulsecoder)**

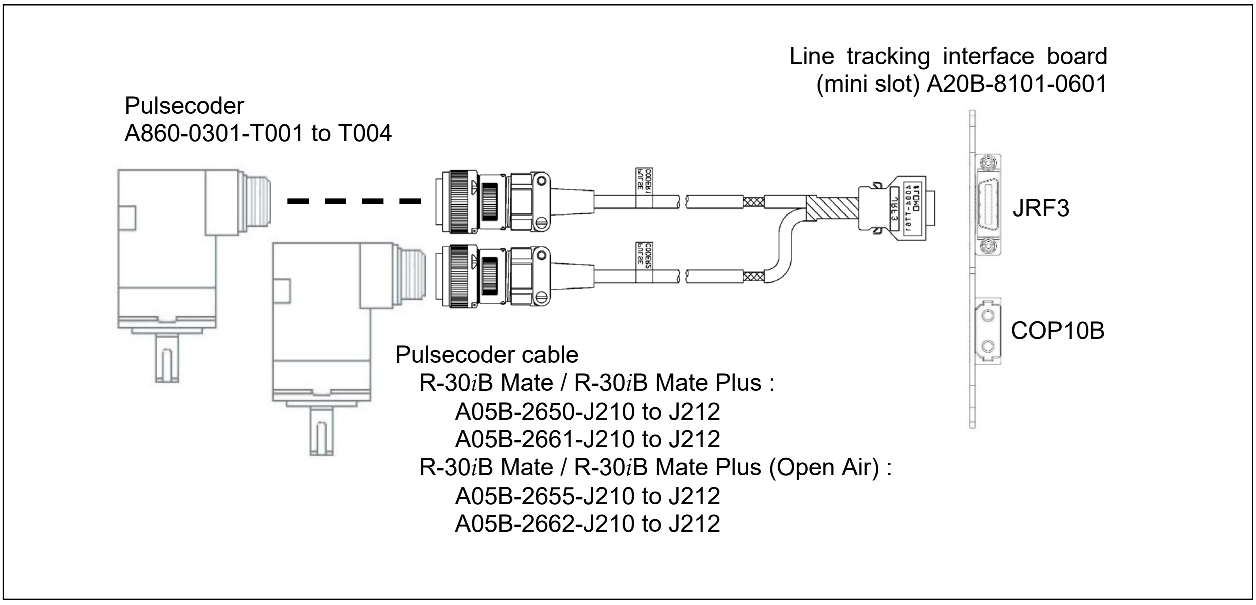



**Fig. C.3 (d) Connecting cables with Line tracking interface board A05B-2650-J035, A05B-2655-J035, A05B-2661-J035 or A05B-2662-J035 (two Pulsecoders)**

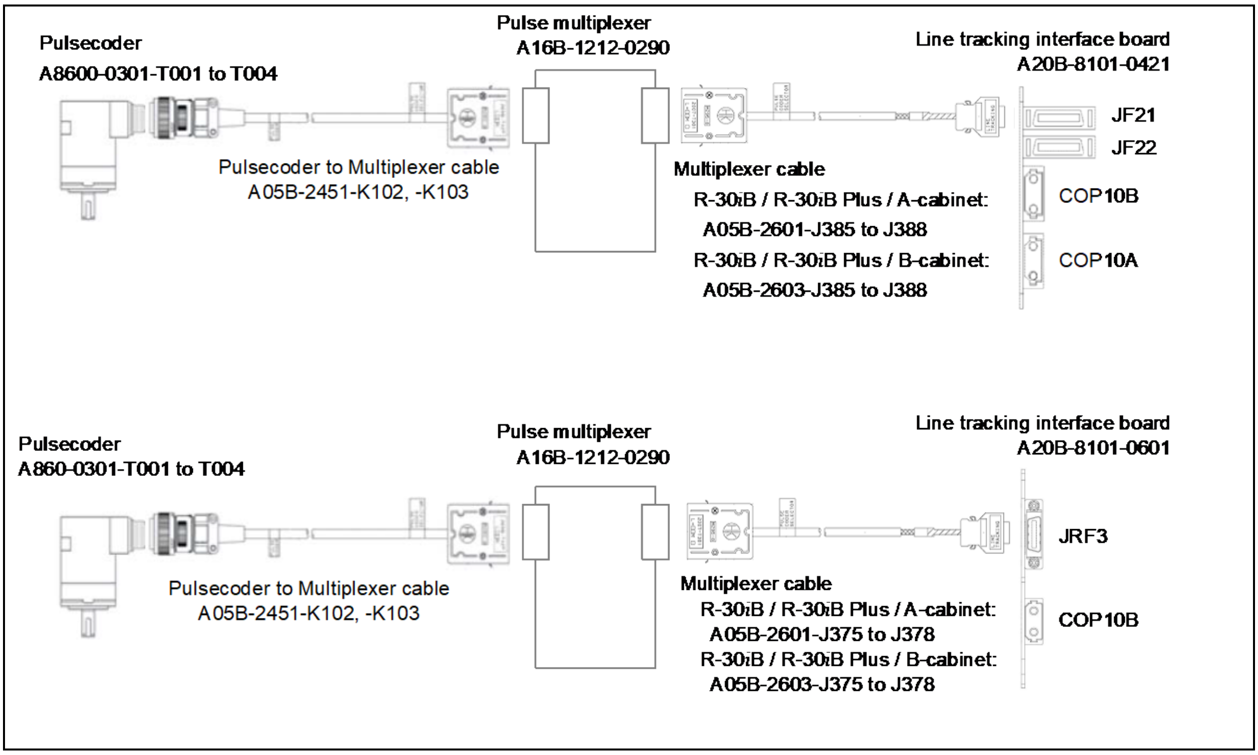



**Fig. C.3 (e) Pulse multiplexer and connecting cables 1 (Pulsecoder, A860-0301-T001 to T004)**

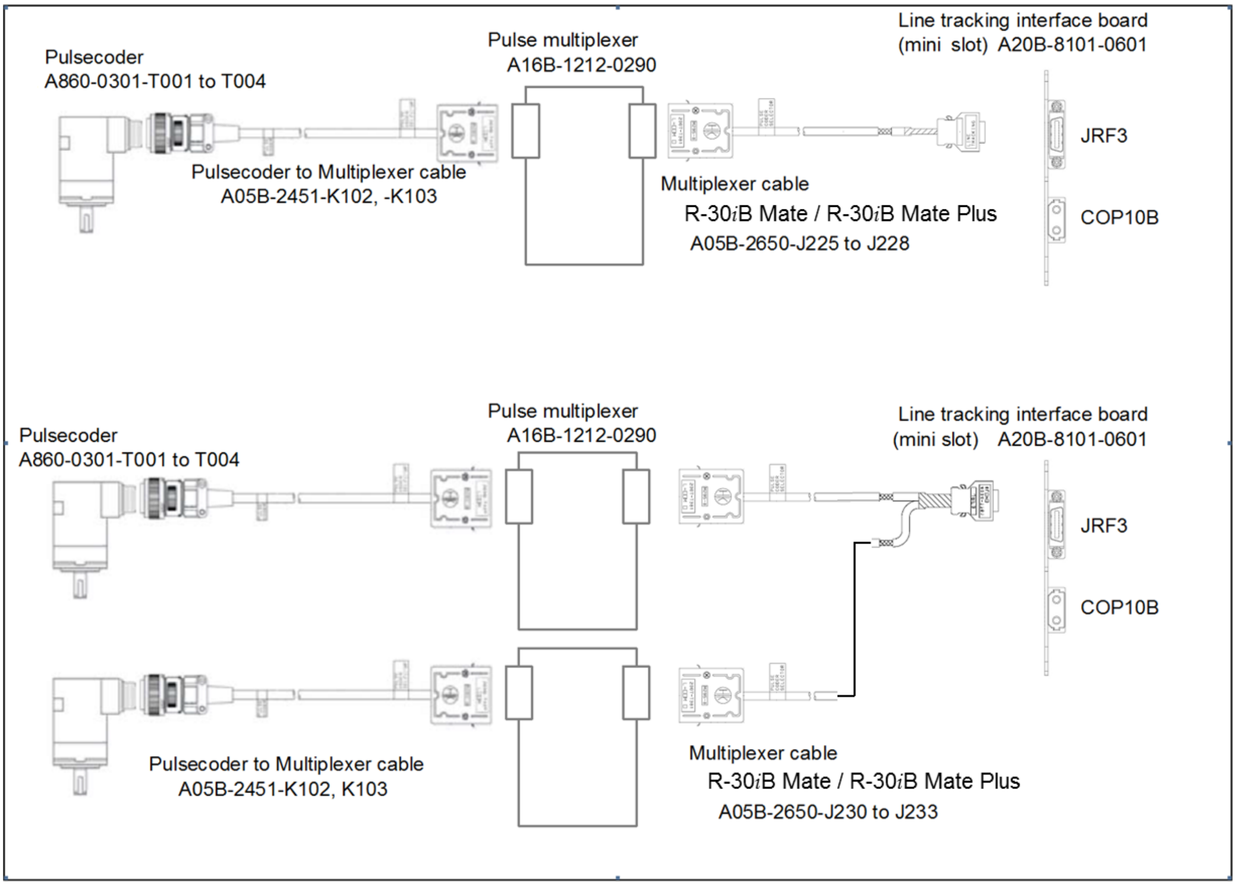



**Fig. C.3 (f) Pulse multiplexer and connecting cables 2 (Pulsecoder, A860-0301-T001 to T004)**

> **NOTE**
> If the line tracking interface board cannot be used or is not available, you can use the SDU shown in Fig. 2.1.1 (f) to Fig. 2.1.1 (n).

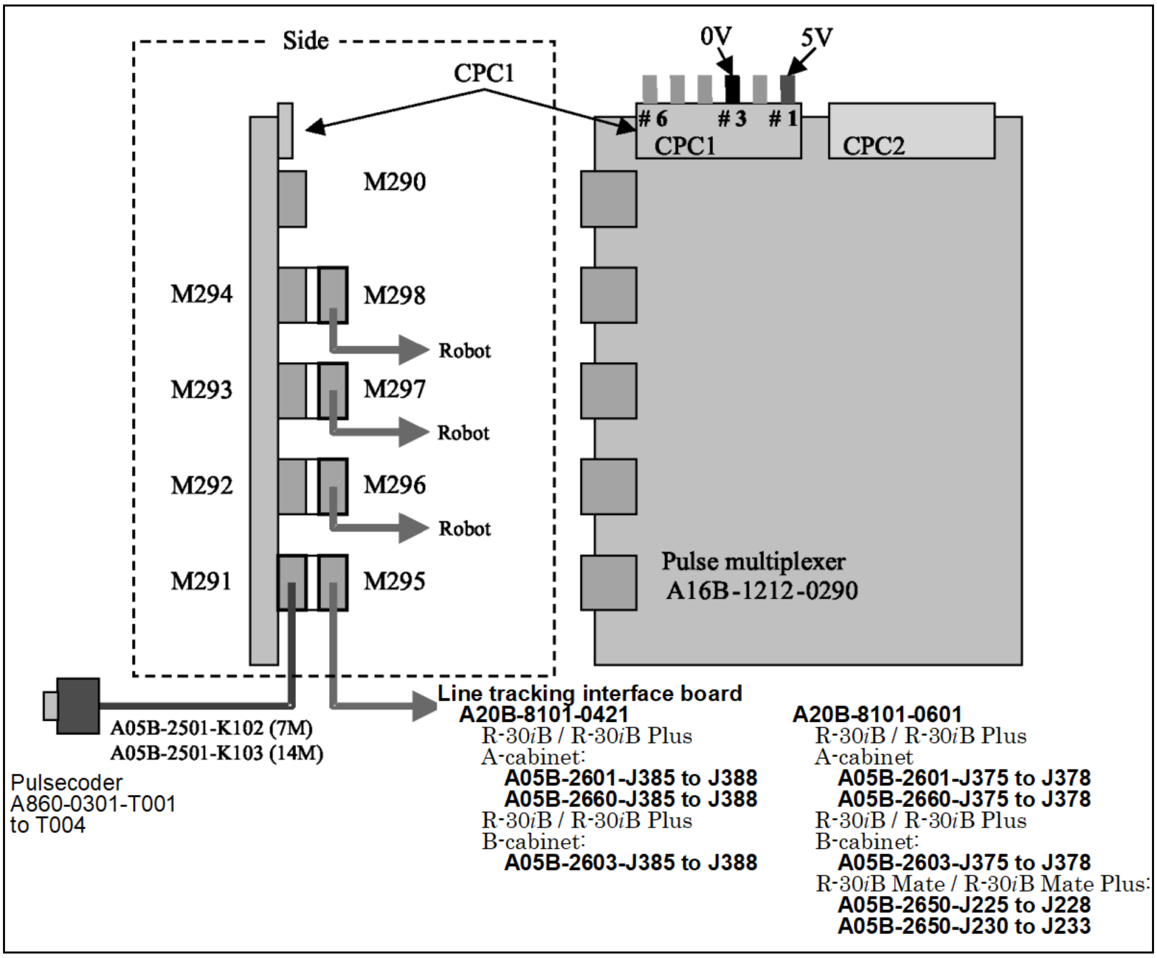



**Fig. C.3 (g) Connecting cables to the multiplexer 3 (Pulsecoder, A860-0301-T001 to T004)**

**NOTE**

You can connect up to four cables (pulse multiplexer to/from controller) to one pulse multiplexer.

# INDEX

# REVISION RECORD

| Edition | Date | Contents |
|---|---|---|
| 03 | Jun., 2022 | • Addition of R-30*i*B Plus / R-30*i*B Mate Plus/ R-30*i*B Compact Plus/ R-30*i*B Mini Plus.<br>• Specification has been modified in some functions.<br>• Some errors in writing have been corrected. |
| 02 | Oct., 2013 | • Addition of R-30*i*B Mate.<br>• Addition of $\alpha$A1000S Pulsecoder.<br>• 3.1.2 Encoder Number Setup has been added.<br>• Specification has been modified in some functions.<br>• Some errors in writing have been corrected. |
| 01 | Oct., 2012 | |

**B-83474EN/03**

* B - 8 3 4 7 4 E N / 0 3 *