Rockwell Automation Application Content

Rockwell Automation Robotics Libraries

Reference Manual

Comau Calibrate Home - Robot

raM_Comau_Robot_Opr_CalibrateHome v2

January, 2024

Rockwell Automation

Important User Information

Solid-state equipment has operational characteristics differing from those of electromechanical equipment. Safety Guidelines for the Application, Installation and Maintenance of Solid State Controls (publication SGI-1.1 available from your local Rockwell Automation sales office or online at http://literature.rockwellautomation.com) describes some important differences between solid-state equipment and hard-wired electromechanical devices. Because of this difference, and because of the wide variety of uses for solid-state equipment, all persons responsible for applying this equipment must satisfy themselves that each intended application of this equipment is acceptable.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.

WARNING 	Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.
IMPORTANT	Identifies information that is critical for successful application and understanding of the product.
ATTENTION 	Identifies information about practices or circumstances or death, property damage, or economic loss. Attentions avoid a hazard and recognize the consequence.
SHOCK HAZARD 	Labels may be on or inside the equipment, that dangerous voltage may be present.
BURN HAZARD 	Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.

Table of Contents

1 Overview

raM_Comau_Robot_Opr_CalibrateHome:

Performs home calibration of the axes on the Comau robot arms.

Use when:

- Using a Comau robot type listed later in this document
- Using a Device Handler for Robot Management
- Need to recalibrate the home position of any joint
- Not using the Rockwell Automation Robotics Libraries rOS object

Do NOT use when:

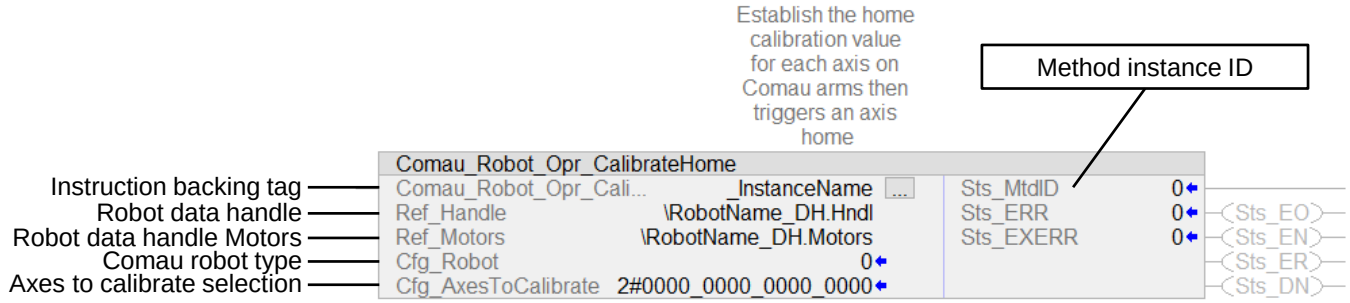
- Not using a Device Handler for Robot Management
- Using the Rockwell Automation Robotics Libraries rOS object
- Not using a Comau robot arm

1.1 Prerequisites

- Device Handler for Robot
  - Rockwell Automation Robotics Libraries v2.0 →
- Studio 5000 – Logix Designer
  - v35.0 →
- Studio 5000 – Application Code Manager
  - v4.03.00 →

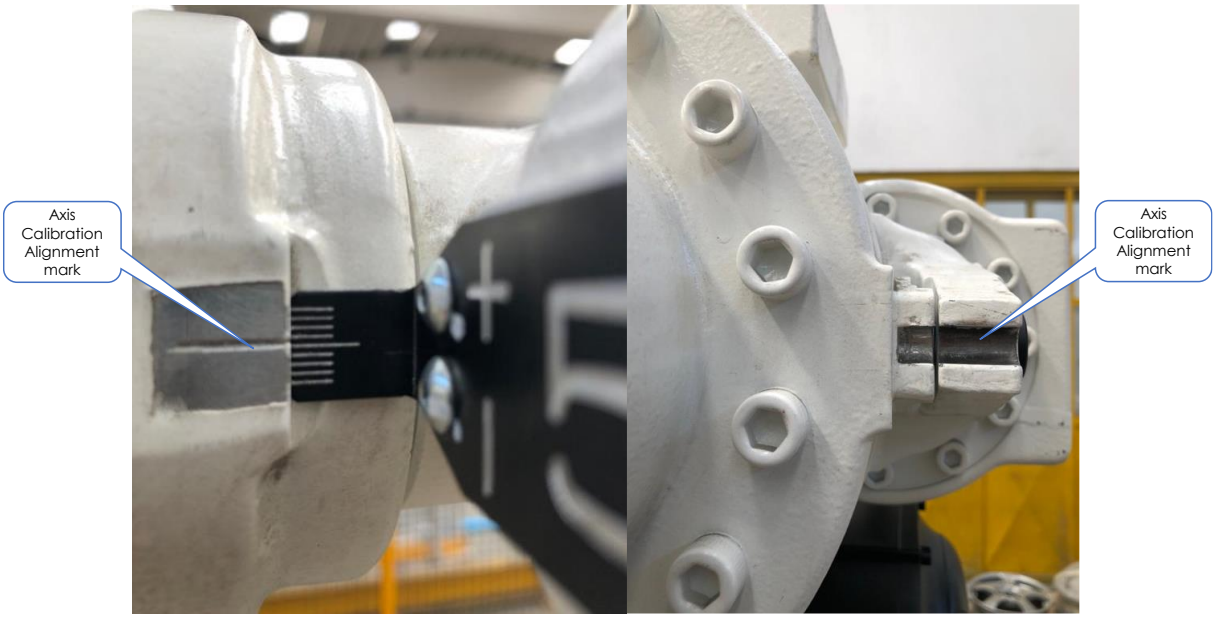
1.2 Functional Description

This instruction is intended for Comau robot arms only. The instruction calculates the current axis position offset from the factory calibrated zero or start position and triggers an axis home command in the device handler. The proper Comau robot type must be selected (Cfg_Robot) prior to executing along with configuring the bit pattern that corresponds to the axes being homed (bit 1=Axis 1, bit 2=Axis 2, etc..). The robot must be disabled before executing this instruction.



Most Comau robot servo axes are equipped with single turn, absolute encoders. The exception are the Comau Rebel SCARA robots. These are equipped with muti-turn, battery backed, absolute encoders. With the Rebels, Joint 1, 2, and 6 come from the factory calibrated. Joint 3 (the quill) would typically be the only axis that would need calibration using this instruction.

During first commissioning, the axis home position must be established on each axis for proper arm function. The procedure is a multi-step process. **This procedure may need to be re-executed on the single turn encoder axes if any axis is moved more than a degree or two while control power is lost to the Kinetix servo drive**. The Rebel SCARA arms, once calibrated, will retain home position as long as the battery is not exhausted. The calibration process begins with jogging the individual axes needing calibration to the factory alignment marks and then disabling the arm. As illustrated below, the alignment marks could be lines or notches depending on the Comau arm being used.



It is not required that the lines or notches be precisely aligned. They only need to be physically within a degree or two. If they are too far apart, the user will receive a "Axis *X* exceeds calibration tolerance" error and the calibration process fails completely. The axis noted in the error message will need to be jogged closer into alignment before reattempting. When first commissioning, it may be found that a joint(s) faults with an "out of position limit" fault since it has not been calibrated. This will prevent the user from completing the calibration process. Joint limit checking can temporarily be inhibited in the robot device handler to eliminate this condition. **Note that there is a possibility that if the marks are grossly misaligned, the calibration process could complete successfully but the result will be a misalignment**. It is up to the technician performing the process to ensure there is proper alignment before executing this instruction. The calibration process can be verified by moving all axes to their respective calibration position using a PTP joint move and verifying that the alignment marks are all precisely aligned.

Note that for the Rebel robots, J3 will need to be positioned with the calibration fixture inserted. This will recalibrate J3 to -120 mm. If the calibration fixture is not available or not desired, the user can move the J3 axis to the toppmost position followed by an icremental move down of 30mm and then execute the Comau_Robot_Opr_Calibration instruction with J3 enabled.

General Status Bit Behavior:

Note: Status bit not shown on the output side of the instruction are not used and will not exist in the instruction backing tag.

Status Bit	Description / Behavior
*.Sts_EO	• Enable Out indicated the status of the output line of the instruction. • If false (logically LO) any instruction on the ladder rung between the instruction and the neutral rail will not be energized. • If the instruction is removed from ladder scan either in a conditional subroutine, MCR zone, JMP/LBL etc., the bit will remain in its last evaluated state.
*.Sts_EN	• The rung-in condition of the ladder rung is true and the instruction is being evaluated. • If the instruction is removed from ladder scan either in a conditional subroutine, MCR zone, JMP/LBL etc., the bit will remain in its last evaluated state.
*.Sts_ER	• If the instruction experiences an internal error, the *. Sts_ER bit will be set. Error codes / Extended codes can be found by monitoring the backing tag *.Sts_ERR / *.Sts_EXERR members respectively. • If the instruction is removed from ladder scan either in a conditional subroutine, MCR zone, JMP/LBL etc., the bit will remain in its last evaluated state.
*.Sts_DN	• Used when the execution of the instruction completes within a single scan. • If the instruction is removed from ladder scan either in a conditional subroutine, MCR zone, JMP/LBL etc., the bit will remain in its last evaluated state.

1.3 Execution

- Edge

1.3.1 Overview

Rung in condition transition response:

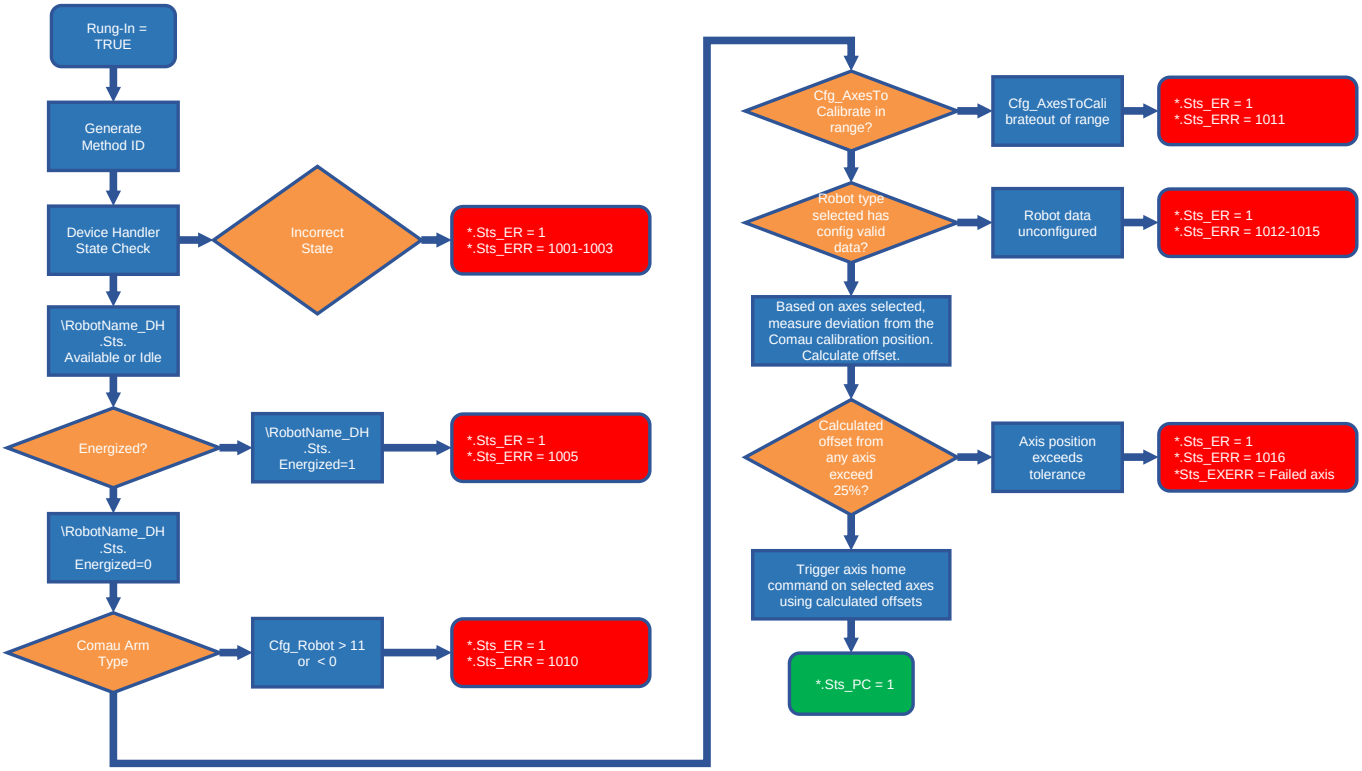
- False → True
  - Initialization
    - *.Sts_EO = 0
    - *.Sts_ER = 0
    - *.Sts_PC = 0
  - Running
    - *.Sts_EO = 1
    - *.Sts_EN = 1
    - IF: Home sequence completes
      - THEN: *.Sts_PC = 1
    - IF: Error
      - THEN: *.Sts_PC = 0 and *.Sts_ER = 1

- True → False
  - *.Sts_EO = 0
  - *.Sts_EN = 0
  - IF: Error
    - THEN: *.Sts_ER = 1

1.3.2 Affected Device Handler Status

None

1.3.3 Execution Table



2 Instruction

2.1 Input Data

Input	Function / Description	DataType
Ref_Handle	Device Handler Data Structure	raM_UDT_Robot_Dvc_DataHndl
Ref_Motors	Device Handler Motors Data Structure	raM_UDT_Robot_Dvc_Motion[1..6]
Cfg_Robot	Comau Arm Type: 0 = NJ40_250 1 = NJ60_220 2 = PAL180_310 3 = PAL260_310 4 = Racer3_063 5 = Racer5_063 6 = Racer5_080 7 = Racer7_140 8 = RebelS6_045 9 = RebelS6_060/RebelS6_060c 10 = RebelS6_075/RebelS6_075c 11 = future	DINT
Cfg_AxesToCalibrate	Bit pattern to select which axes to calculated calibrate value and home. (1 = calibrate/home, 0 = do not calibrate/home). Bit 0 corresponds to axis 1, bit 1 corresponds to axis 2, etc…	INT
Cfg_MfgCalibration DataAxisX	Axis X Encoder count offset if provided by manufacturer. (optional) Otherwise, set = 0.0	DINT

2.2 Output Data

Output	Function / Description	DataType
Sts_EO	Instruction has enabled the rung output. Provides a visible indicator of the EnableOut system parameter for use during ladder instantiation	BOOL
Sts_EN	Instruction is Being Scanned - Rung In Condition = TRUE	BOOL
Sts_ER	Instruction is in Error - See Sts_ERR / Sts_EXERR for Additional Error Information	BOOL
Sts_ERR	Instruction Error Code - See Instruction Help for Code Definition	DINT
Sts_EXERR	Instruction Extended Error Code - See Instruction Help for Code Definition	DINT
Sts_MtdID	Method ID	DINT
Sts_DN	Instruction Done	BOOL

2.3 Error Codes

Sts_ERR	Description
0	No errors present
1001	Device Handler is not in a running state. Commands to the device cannot be processed.
1002	Device Handler is faulted. Clear faults to apply commands
1003	Device Handler is not in a supported state.
1005	Robot is energized. Must be deenergized for calibration/homing
1010	Comau Arm type out of range. Valid range is 0 through 11
1011	Axes selected for calibration out of range.
1012	Feedback Counts Unconfigured Axis x. See Sts_EXERR for axis number
1013	Transmission Ratio Input Unconfigured Axis x. See Sts_EXERR for axis number
1014	Transmission Ratio Output Unconfigured Axis x. See Sts_EXERR for axis number
1015	Motion Polarity Unconfigured Axis x. See Sts_EXERR for axis number
1016	Axis position exceeds calibration tolerance. Move axis closer to factory calibration mark. See Sts_EXERR for axis number

Sts_EXERR	Description
< Number >	Axis number that failed calibration in errors from Sts_ERR

3 Application Code Manager

3.1 Definition Object: Comau_Robot_Opr_CalibrateHome

This object contains the AOI definition and used as linked library to implement object. This gives flexibility to choose to instantiate only definition and create custom implement code. User may also create their own implement library and link with this definition library object.

3.2 Implementation Object: Comau_LD _Robot_CalibrateHome

Implementation Language: Ladder
Content Type: Routine

This implement contains only a rung with an instance of the Comau_Robot_Opr_CalibrateHome object.

Parameter Name	Default Value	Instance Name	Definition	Description
RoutineName	_{ObjectName}	{RoutineName}	Routine	Name of the routine where the object will be placed
TagName	{ObjectName}	{TagName}	Tag	Instruction backing tag
StartBitTagName	Cmd_{ObjectName}		Local Tag	Tag name for start command enabling bit

Linked Library

Link Name	Catalog Number	Revision	Solution	Category
RobotHandler	raM_Robot_Dvc_DeviceHandler	2	(RA-LIB) Robotics	Robot Handler
Comau_Robot_Opr_CalibrateHome	Comau_Robot_Opr_CalibrateHome	2	(RA-LIB) Robotics	Asset-Control

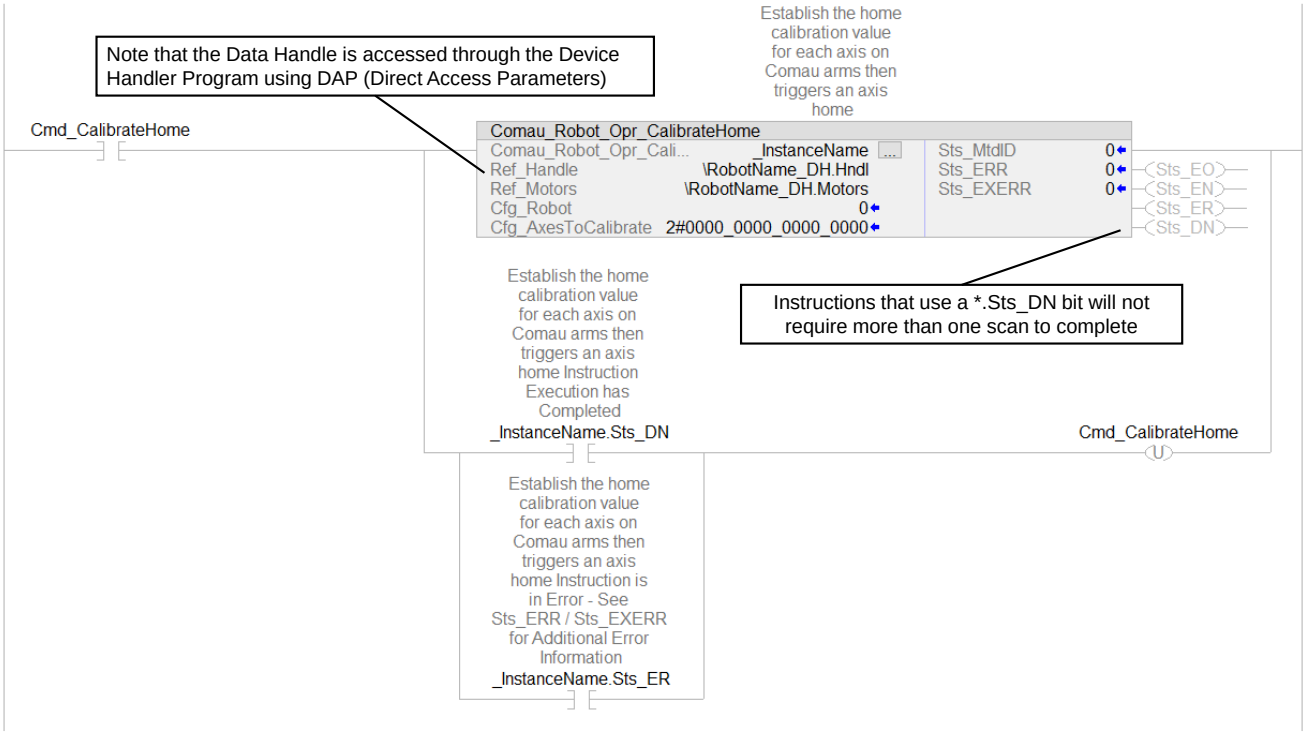
3.3 Attachments

Name	Description	File Name	Extraction path
V2_{LibraryName}	Reference Manual	RM-{LibraryName}.pdf	{ProjectName}\Documentation

4 Application

4.1 Using Comau_Robot_Opr_CalibrateHome

Populate Cfg parameters directly or MOV values into Cfg parameters prior to executing. If using the Rockwell Robotics Libraries rOS program, it is not necessary to instantiate this instruction in user code.



5 Appendix

General

This document provides a programmer with details on this OEM Building Block instruction for a Logix-based controller. You should already be familiar with how the Logix-based controller stores and processes data.

Novice programmers should read all the details about an instruction before using the instruction. Experienced programmers can refer to the instruction information to verify details.

IMPORTANT This OEM Building Block Instruction includes an Add-On Instruction for use with Version 34 or later of Studio 5000 Logix Designer.

Common Information for All Instructions

Rockwell Automation Building Blocks contain many common attributes or objects. Refer to the following reference materials for more information:

- Foundations of Modular Programming, **IA-RM001C-EN-P**

Conventions and Related Terms

Data - Set and Clear

This manual uses set and clear to define the status of bits (Booleans) and values (non-Booleans):

This Term:	Means:
Set	The bit is set to 1 (ON) A value is set to any non-zero number
Clear	The bit is cleared to 0 (OFF) All the bits in a value are cleared to 0

Signal Processing - Edge and Level

This manual uses Edge and Level to describe how bit (BOOL) Commands, Settings, Configurations and Inputs to this instruction are sent by other logic and processed by this instruction.

Send/Receive Method:	Description:
Edge	• Action is triggered by "rising edge" transition of input (0-1) • Separate inputs are provided for complementary functions (such as "enable" and "disable") • Sending logic SETS the bit (writes a 1) to initiate the action; this instruction CLEARS the bit (to 0) immediately, then acts on the request if possible • LD: use conditioned OTL (Latch) to send • ST: use conditional assignment [if (condition) then bit:=1;] to send • FBD: OREF writes a 1 or 0 every scan, should use Level, not Edge Edge triggering allows multiple senders per Command, Setting, Configuration or Input (many-to-one relationship)
Level	• Action ("enable") is triggered by input being at a level (in a state, usually 1) • Opposite action ("disable") is triggered by input being in opposite state (0) • Sending logic SETS the bit (writes a 1) or CLEARS the bit (writes a 0); this instruction does not change the bit • LD: use OTE (Energize) to send • ST: use unconditional assignment [bit:= expression_resulting_in_1_or_0;] or "if-then-else" logic [if (condition) then bit:= 1; else bit:= 0;] • FBD: use OREF to the input bit Level triggering allows only one sender can drive each Level

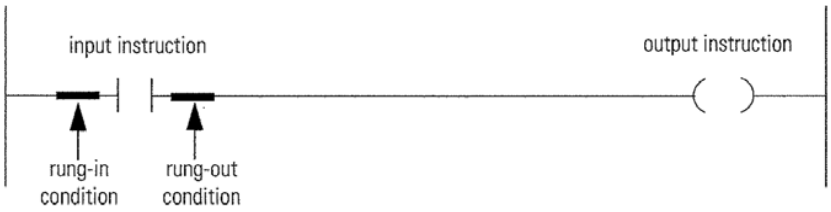
Instruction Execution - Edge and Continuous

This manual uses Edge and Continuous to describe how an instruction is designed to be executed.

Method:	Description:
Edge	• Instruction Action is triggered by "rising edge" transition of the rung-in-condition
Continuous	• Instruction Action is triggered by input being at a level (in a state, usually 1) • Opposite action is triggered by input being in opposite state (0) • Instructions designed for continuous execution should typically be used on rungs without input conditions present allowing the instruction to be continuously scanned

Relay Ladder Rung Condition

The controller evaluates ladder instructions based on the rung condition preceding the instruction (rung-in condition). Based on the rung-in condition and the instruction, the controller sets the rung condition following the instruction (rung-out condition), which in turn, affects any subsequent instruction.



If the rung-in condition to an input instruction is true, the controller evaluates the instruction and sets the rung-out condition based on the results of the instruction. If the instruction evaluates to true, the rung-out condition is true; if the instruction evaluates to false, the rung-out condition is false.

IMPORTANT

The rung-in condition is reflected in the EnableIn parameter and determines how the system performs each Add-On Instruction. If the EnableIn signal is TRUE, the system performs the instruction's main logic routine. Conversely, if the EnableIn signal is FALSE, the system performs the instruction's EnableInFalse routine.

The instruction's main logic routine sets/clears the EnableOut parameter, which then determines the rung-out condition. The EnableInFalse routine cannot set the EnableOut parameter. If the rung-in condition is FALSE, then the EnableOut parameter and the rung-out condition will also be FALSE.

Pre-scan

On transition into RUN, the controller performs a pre-scan before the first scan. Pre-scan is a special scan of all routines in the controller. The controller scans all main routines and subroutines during pre-scan, but ignores jumps that could skip the execution of instructions. The controller performs all FOR loops and subroutine calls. If a subroutine is called more than once, it is performed each time it is called. The controller uses pre-scan of relay ladder instructions to reset non-retentive I/O and internal values.

During pre-scan, input values are not current and outputs are not written. The following conditions generate pre-scan:

- Transition from Program to Run mode.
- Automatically enter Run mode from a power-up condition.

Pre-scan does not occur for a program when:

- Program becomes scheduled while the controller is running.
- Program is unscheduled when the controller enters Run mode.

IMPORTANT The Pre-scan process performs the Process Add-On Instruction's logic routine as FALSE and then performs its Pre-scan routine as TRUE.