Rockwell Automation Application Content

Machine Builder Libraries

Reference Manual

Safe Torque Off – CIP Drive

raM_Opr_SafeTrqOff_CD **v2.x**

March, 2021

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_Opr_SafeTrqOff_CD:

- Provides simplified management of Safe Torque Off (STO) over CIP Safety network connection to CIP Safety-capable drives.

Use when:

- Control of Safe Torque Off for drives supporting CIP Safety over EtherNet/IP is required.

Do NOT use when:

- Drive does not support CIP Safety, or Safe Torque Off management is done by hardwired control.

1.1 Prerequisites

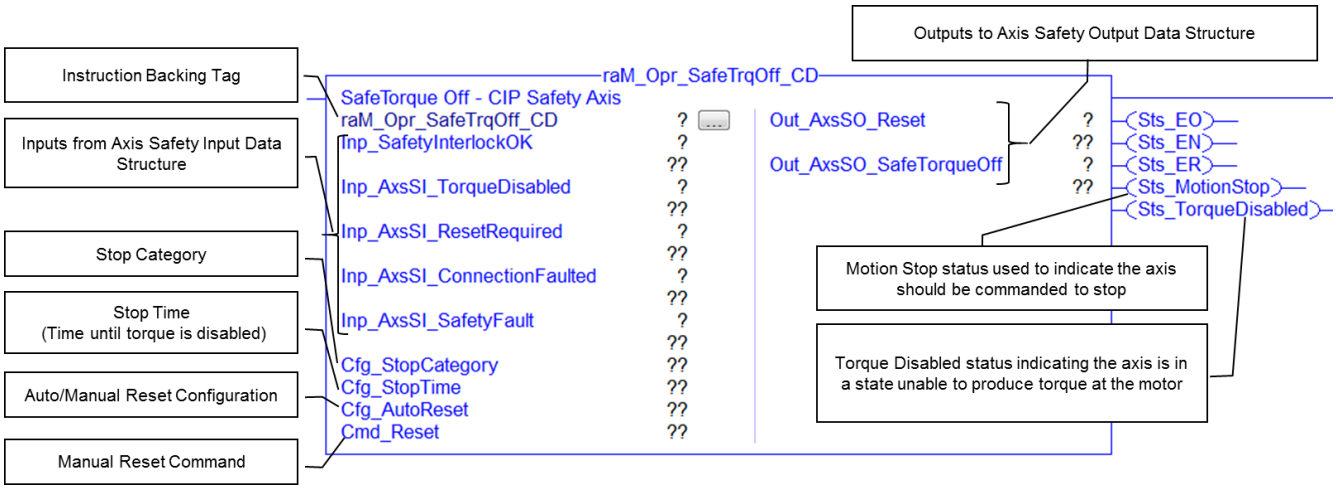
- Drive must support CIP Safety.
- Studio 5000 - Logix Designer
  - v30.0 →
- Studio 5000 - Application Code Manager
  - v2.0 →

1.2 Functional Description

The latest generation of Safe Torque Off-capable drives include the ability to disable the torque producing capability of the drive using the network connection with CIP Safety over EtherNet/IP. The network connection can provide diagnostics on the same connection that provide the loop control all while reducing wiring as drive Safe Torque Off connections no longer require physical hardwiring.

The raM_Opr_SafeTrqOff_CD instruction provides a simple way to manage control of the drive Safe Torque Off operation with the following features:

- Single point evaluation of safety interlock
- Manual Reset
- Automatic Reset
- Configurable Stop Category
  - Stop Category 0
  - Stop Category 1
- Configurable Stop Time (Stop Category 1)
- Clear indication of torque producing potential



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.
*.Sts_IP	• Used to identify the instruction is in the process • 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_PC	• Used when the execution of the instruction requires more than a single scan to complete, and indicates the 'process' carried out by the instruction has successfully completed. • 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

1.3.1 Overview

- Level

- Instruction Initialization
  - *.Sts_EO = 0
  - *.Sts_EN = 0
  - *.Sts_ER = 0

- Instruction being executed
  - *.Sts_EO = 1
  - *.Sts_EN = 1
  - *.Sts_ER = 0
    - IF: Error -> Sts_ER = 1 and Sts_IP = 0
    - IF Not Error -> Sts_IP = 1

1.3.2 Execution Table

2 Instruction

2.1 Footprint

Characteristic	Description	Value	Unit
Definition	Estimated memory required to store the object definition, including all dependents.	1.9	kB
Instance	Estimated memory required per object instantiated. This includes the object instance and all datatypes required to verify the project. In the case of user configurable arrays, an application relevant array length will be used for estimation.	0.8	kB
Execution L7x	Estimated execution time / scan footprint evaluated in 1756-L7x PAC	45	us

2.2 Input Data

Input	Function / Description	DataType
Inp_SafetyInterlockOK	Safety interlock (1=Interlocks OK)	BOOL
Inp_AxsSI_TorqueDisabled	Axis safety structure input *.TorqueDisabled	BOOL
Inp_AxsSI_ResetRequired	Axis safety structure input *.ResetRequired or RestartRequired	BOOL
Inp_AxsSI_ConnectionFaulted	Axis safety structure input *.ConnectionFaulted	BOOL
Inp_AxsSI_SafetyFault	Axis safety structure input *.SafetyFault	BOOL
Cfg_StopCategory	Stop category 0 or 1	DINT
Cfg_StopTime	Stop category 1 time delay (ms)	DINT
Cfg_AutoReset	Automatic STO fault reset	BOOL
Cmd_Reset	Safe Torque Off reset command. Edge driven command that is reset back to zero by the instruction.	BOOL

2.3 Output Data

Output	Function / Description	DataType
raM_Opr_SafeTrqOff_CD	Object Identifier	BOOL
Sts_EO	Input energize	BOOL
Sts_EN	Identify object is or has been in scan	BOOL
Sts_ER	Instruction has experienced an internal error	BOOL
Sts_ERR	Instruction Error Code	DINT
Sts_EXERR	Instruction Extended Error Code	DINT
Out_AxsSO_Reset	Axis safety structure output *.Reset or ResetRequest	BOOL
Out_AxsSO_SafeTorqueOff	Axis safety structure output *.SafeTorqueOff or STOOutput	BOOL
Sts_MotionStop	Output to stop a motion (in stop category 1)	BOOL
Sts_TorqueDisabled	The torque from the axis has been removed	BOOL

2.4 Error Codes

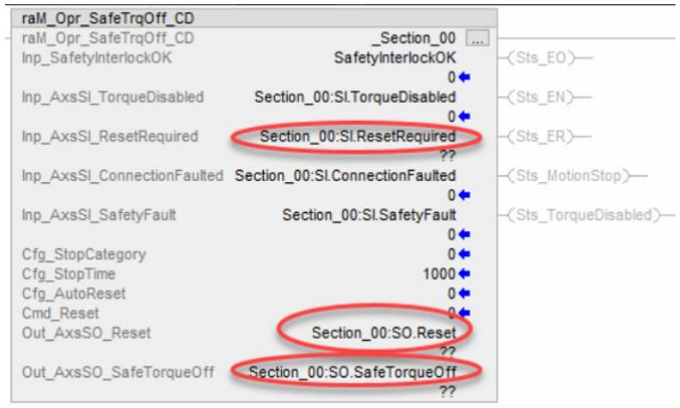
Sts_ERR	Description
1010	Invalid Cfg_StopCategory. Stop Category must be either 0 or 1.
1011	Invalid Cfg_StopTime. When configured for Stop CAT1, stop time must be greater than zero.

Sts_EXERR	Description
N/A	N/A

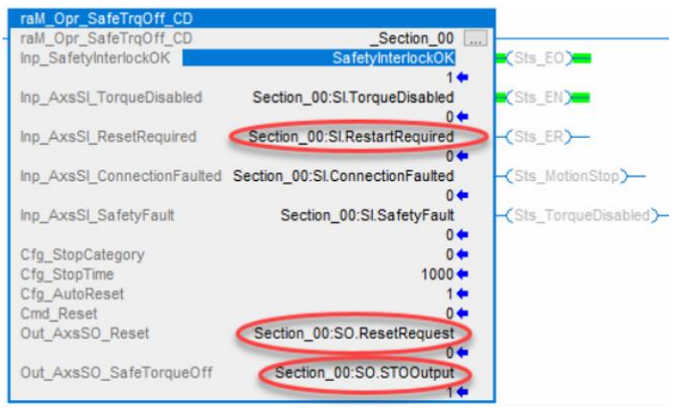
2.5 Hardware Connections based on Revision

Note the connection names changes between different hardware revision of the drive.

Revision 7 and earlier



Above Revision 7



3 Application Code Manager

3.1 Definition Object: raM_Opr_SafeTrqOff_CD

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 Implement Object: raM_LD_SafeTrqOff_CD

Implement Language: Ladder Diagram

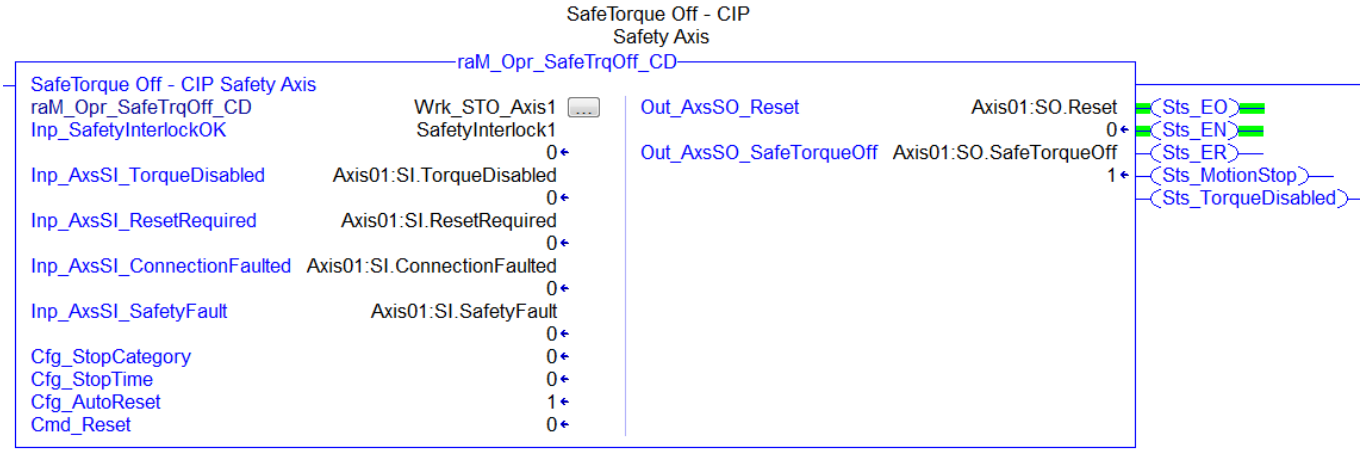
Parameter Name	Default Value	Instance Name	Definition	Description
ObjectName	raM_Opr_SafeTrqOff_CD			Object Name
RoutineName	{ObjectName}	{RoutineName}	Routine	Name of the routine where the object will be placed
TagName	_{ObjectName}		Local Tag	Instruction backing tag
SafetyInterlockOKName	_SafetyInterlockOK	_{SafetyInterlockOKName}{AxNr}	Local Tag	Instruction backing tag for Channel 1
		_{SafetyInterlockOKName}{AxNr}	Local Tag	Instruction backing tag for Channel 2
AxisModuleName	AxisModuleName	{AxisModuleName}	--	Name of the CIP safety Axis Module
NumberOfAxesPerModule	1	--	--	Number of axis channels per HW module

Linked Library

Link Name	Catalog Number	Revision	Solution	Category
raM_Opr_SafeTrqOff_CD	raM_Opr_SafeTrqOff_CD	>=2.0	(RA-LIB) Machine	Safety

4 Application

The instruction sets/clears the Safe Torque Off output bit based on feedback from the axis safety input data structure inputs and status of '*Inp_SafetyInterlockOK'*.

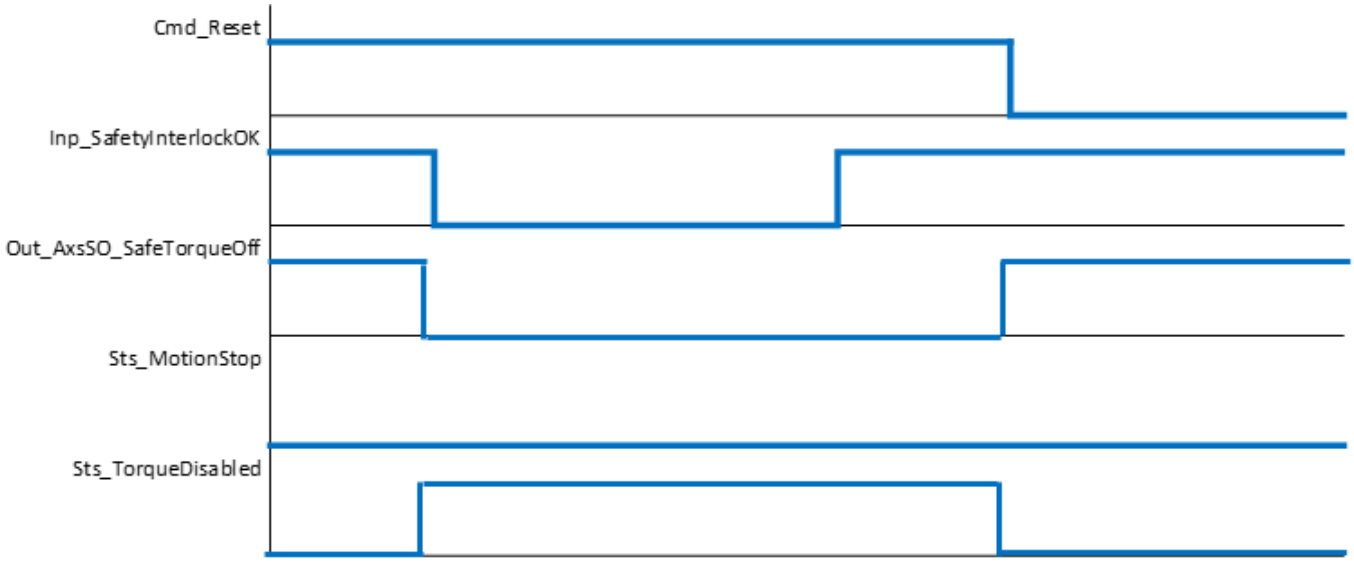


Setup the Instruction:

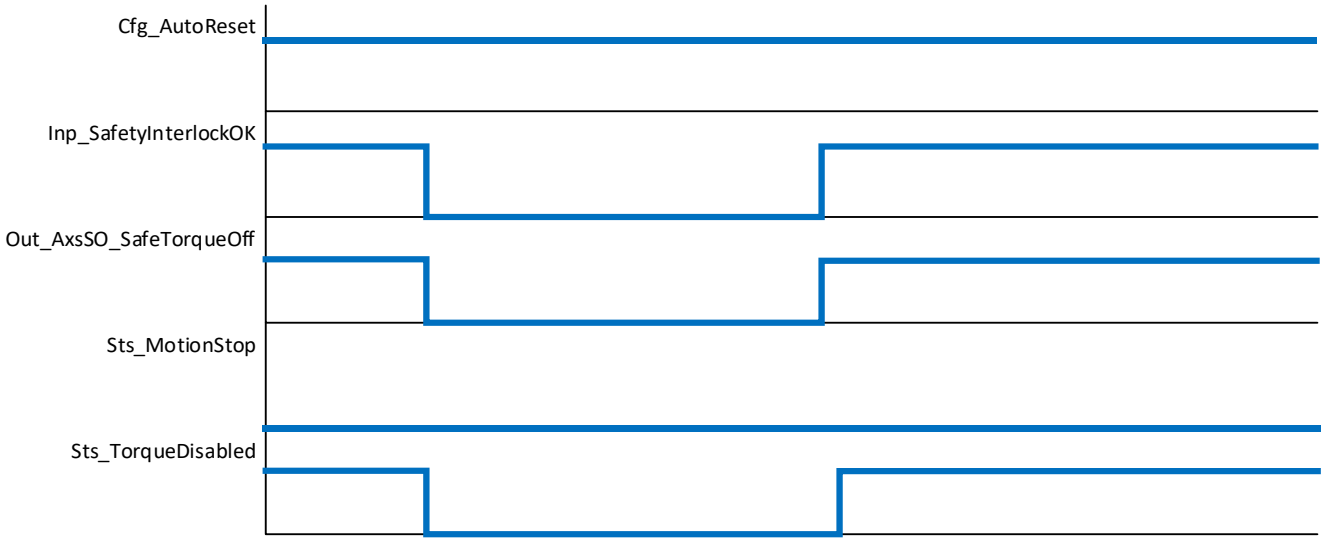
- Select stop category 0 or 1 and set '*Cfg_StopCategory'*
- If Category 1 Stop is selected, set *Cfg_StopTime* to a value greater than 0 milliseconds to allow the application program to stop movement of the axis prior to the output torque of the drive being disabled.
- Populate the safety interlock input. This bit represents the sum of all safety inputs, which will determine when the drive's output power structure should be disabled. (i.e. doors and e-stops in safety zone where the axis is moving)
- Populate axis safety inputs (*Inp_Axs:SI_xxx*)
- Populate axis safety outputs(*Inp_Axs:SO_xxx*)
- If automatic reset of the drive Safe Torque off is desired, set *Cfg_AutoReset* to 1 and AOI will set STO back as soon as axis safety inputs and *Inp_SafetyInterlockOK* allow it.
- If manual reset of the drive Safe Torque off is desired, generate a falling edge on *Cmd_Reset* when axis safety inputs and *Inp_SafetyInterlockOK* are in the appropriate states to allow resetting.

4.1 Timing Diagrams

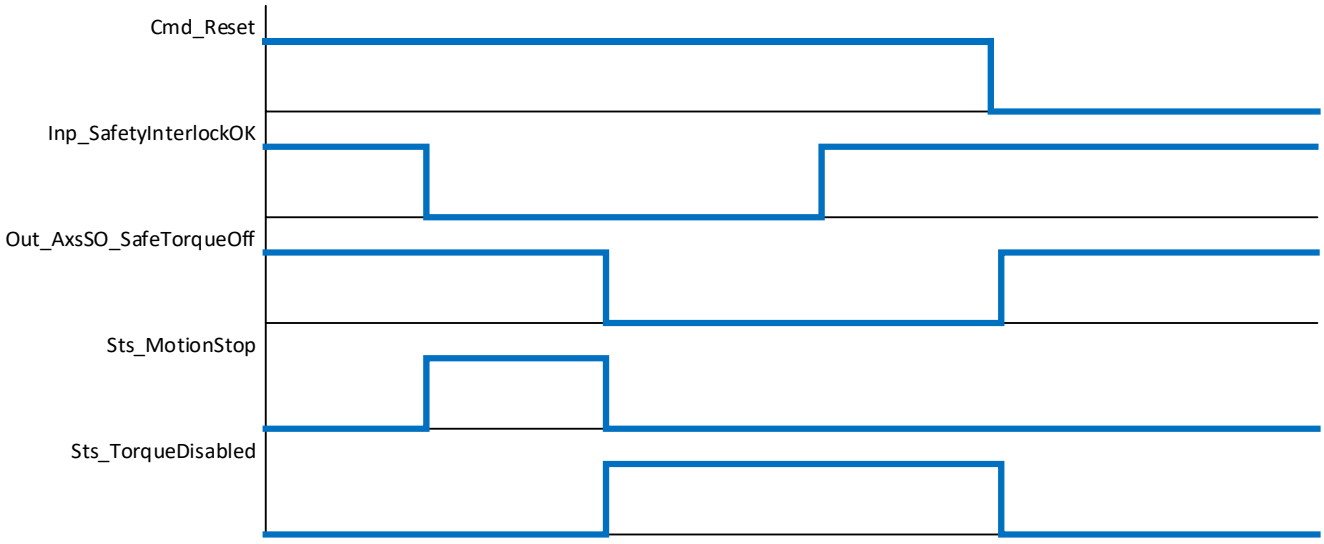
4.1.1 Stop Category 0 / Manual Reset



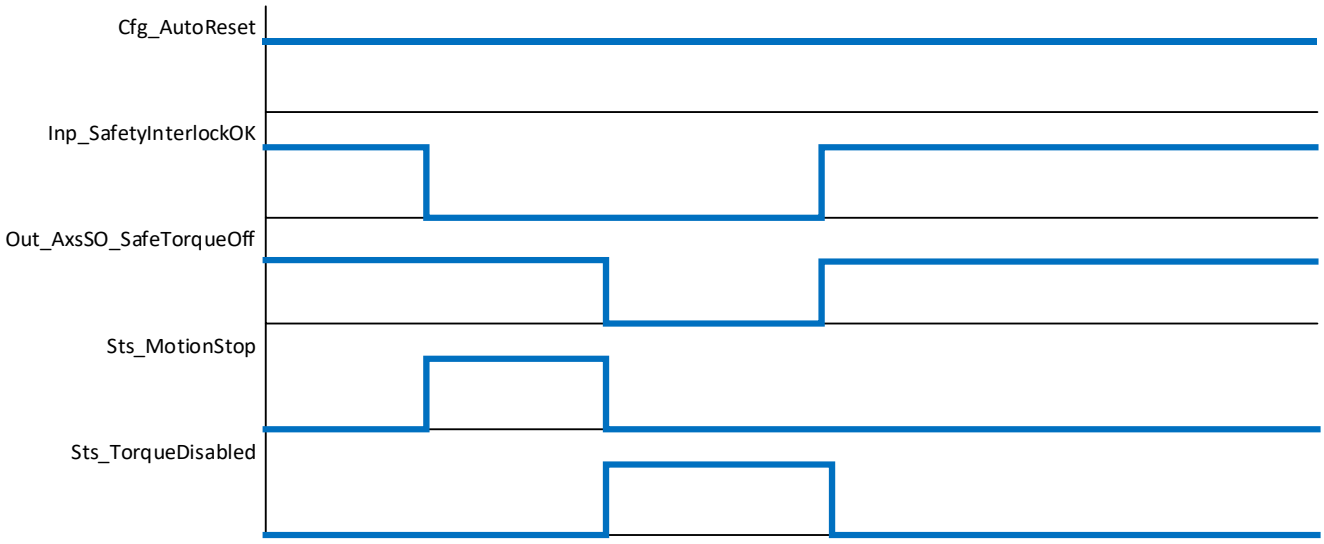
4.1.2 Stop Category 0 / Automatic Reset



4.1.3 Stop Category 1 / Manual Reset



4.1.4 Stop Category 1 / Automatic Reset



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 24 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

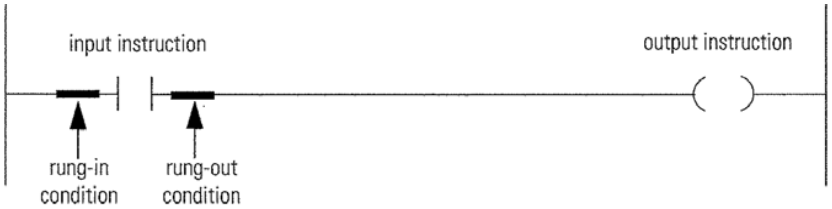
Instruction Execution - Edge and Continuous

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.