Rockwell Automation Application Content

Machine Builder Libraries

Reference Manual

Axis Handler – CIP Drive

raM_Dvc_AxisHandlerCD	v2.x
raM_Dvc_AHLP	v2.x
raM_Dvc_DH_SysIni	v1.x

Rockwell Automation

September, 2024

Important User Information

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

Table of Contents **3**

1 Overview **5**

1.1 Prerequisites 5

1.2 Functional Description 6

1.3 Execution Scheduling 7

1.4 Footprint 7

2 Axis Handler Architecture **8**

2.1 Understanding Axis Handler Components 8

2.2 Connecting Axis Handler Components 13

2.3 Connecting to the Application 15

3 Handler Operation **16**

3.1 Axis Handler Modes 16

3.2 Axis Handler Transition Rules 16

3.3 Axis Handler States 17

4 User Interface **18**

4.1 Home 18

4.2 Settings 19

4.3 Configuration 21

4.4 Fault 24

4.5 FactoryTalk View 25

4.6 View Designer 26

4.7 FactoryTalk Optix 31

5 Events **35**

5.1 Status 35

5.2 Methods 36

6 Application Code Manager **38**

6.1 Definition Object: raM_Dvc_AxisHandler_CD 38

6.2 Implement Object: raM_LD_AxisHandler_CD 38

6.3 Implement Object: raM_LD_AHLP 39

6.4 Implement Object: raM_LD_DH_SysIni 39

6.5 Attachments 39

7 Application **40**

7.1 Instantiate Axis Handler and Method in Application Code Manager 40
7.2 Interfacing from Application Code 49
7.3 Method Error Configuration 50
7.4 User Defined Language 51

8 Learning Resources 55
8.1 Videos on Rockwell Automation YouTube Channel 55
8.2 Labs available on Rockwell Automation Cloud Based Learning Environment (onCourse) 55

9 For More Information 56

10 Appendix 57
General 57
Common Information for All Instructions 57
Conventions and Related Terms 57

1 Overview

The Axis Handler CIP provides enhanced management of a CIP Motion Axis, including text-based status and diagnostics, manual operator control, portfolio of available Methods, configurable Method Error as warning or fault, and associated faceplate for control and display.

Use when:

- Basic Operator Control of CIP Motion Drive is desired
- Device Faceplate is desired
- Enhanced Axis information views are desired
- Axis Methods (instructions) that require connection to an axis handler are used in the application
- Axis Virtualization is desired
- Utilizing a Device Object for hardware abstraction

Do NOT use when:

- Device is NOT a CIP Motion Drive

1.1 <u>Prerequisites</u>

- PAC
  - ControlLogix / CompactLogix with 2 Mb or greater memory
- Studio 5000 - Logix Designer
  - v30.0 →
- FactoryTalk View Studio ME/SE
  - V10.0 →
- FactoryTalk Optix
  - V1.4 →
- Studio 5000 - Application Code Manager
  - V4.0 →

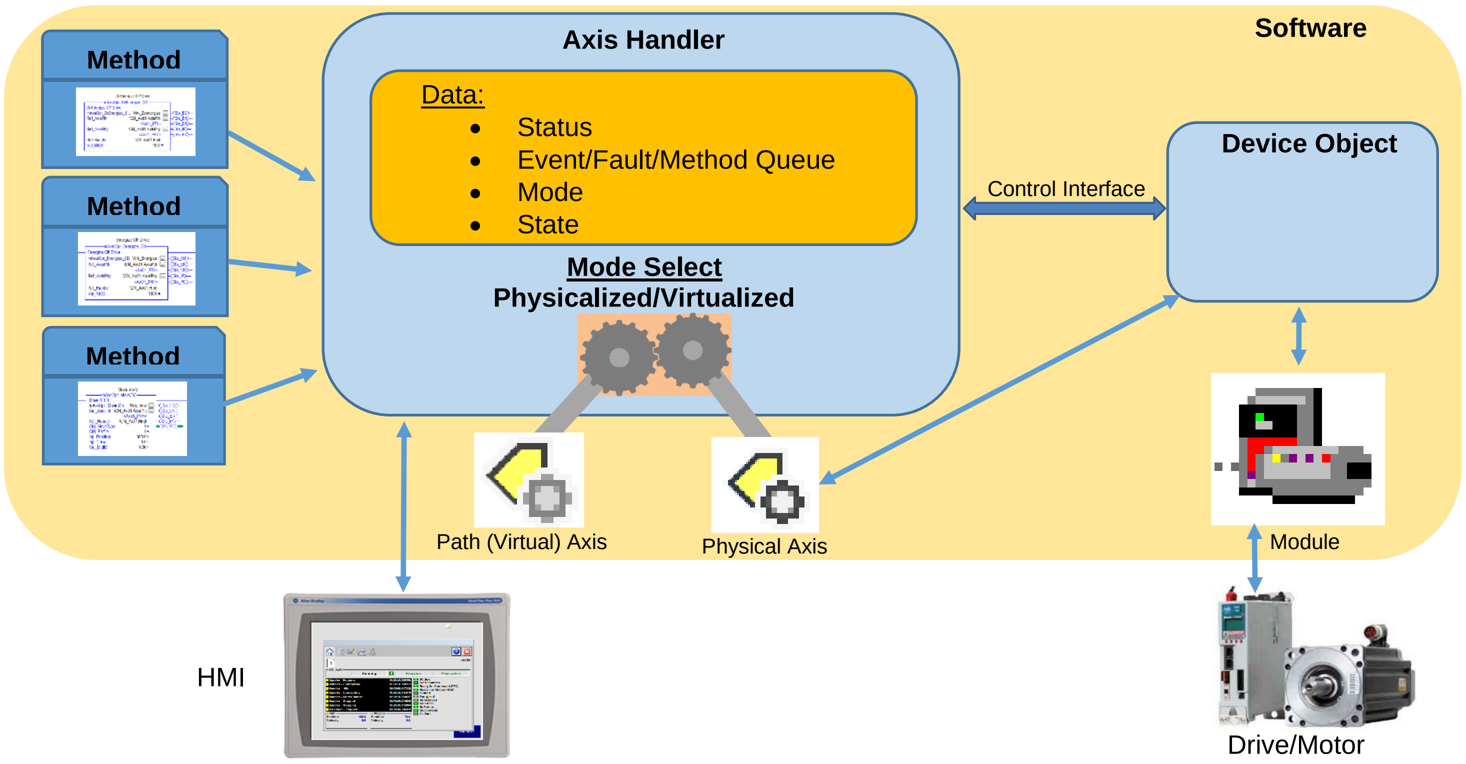
1.2 Functional Description

The CIP Motion Drive Axis Handler is a powerful diagnostic and management tool for use with all supported CIP drives. At its core, it consists of an instruction paired with an associated data structure which directly communicates with a linked path/physical axis pair.

Axis Handler provides:

- Focused and enhanced set of status tags to allow for easy troubleshooting
- Hardware abstraction by sending commands only to the path (virtual) axis
- Physicalized/Virtualized mode selection which communicates to methods whether to engage/disengage gearing of the physical axis to the path axis
- Facilitation of simple axis recovery
- At-a-glance indication of status, methods, faults, states, commands
- Ability for the user to jog an axis, change jog dynamics, and clear faults directly from the HMI.
- Ability to configure handler to fault the device in case of Method Error
- Included faceplate

All of these features deliver the user a tool that provides quick feedback, shortens recovery time, and simplifies implementation.



During the configuring state, the Path Axis automatically takes on the configuration of the Physical Axis.

The handler enhances user experience by:

- Providing code-free commissioning via the faceplate
    - Axis fundamental status
    - Axis jogging without any additional programming
    - Text-based diagnostic messages for faults and events
    - Multilingual support for diagnostic messages

- Combining and simplifying essential axis status
    - Consolidated status bits for use in coding
        - Axis is Connected
        - Axis is Available
        - Axis is Ready
        - Axis is Mode Locked
        - Axis is Energized
    - Text-based first-out fault for fault aggregation
    - Accessible Event Queue

- Virtualization at runtime
    - Manages coupling of physical axis to path axis
    - Motion executed on path axis and status is reflected on physical axis
    - Allows decoupling of physical axis hardware for/during testing, debugging, and simulation

- Simplified Recovery
    - Facilitates bidirectional recovery of physical axis and path axis using raM_Opr_SyncPthPhyAx_CD method
    - Path axis can stay synchronized to master while physical axis is faulted

- Simplified Device interlock and instruction error
    - Handler configuration allows selection between warning or fault when a method error is present
    - Avoid writing logic to deal with Method Error

1.3 Execution Scheduling

- Handler Program containing Handler instruction
    - 64ms suggested (periodic task)
- Language Pack
    - 512ms suggested (periodic task) / low priority
- System Initialization
    - PAC Power-Up Handler

1.4 Footprint

Characteristic	Description	Value	Unit
Definition	Estimated memory required to store the object definition, including all dependents (including 1 Instance).	131	kB
Instance	Estimated memory required per object instantiated.	7	kB
Execution L8x	Estimated execution time / scan footprint evaluated in 1756-L8x PAC	40	Us

NOTE: Footprint estimations above EXCLUDE physical, path axis, language pack and Device Object

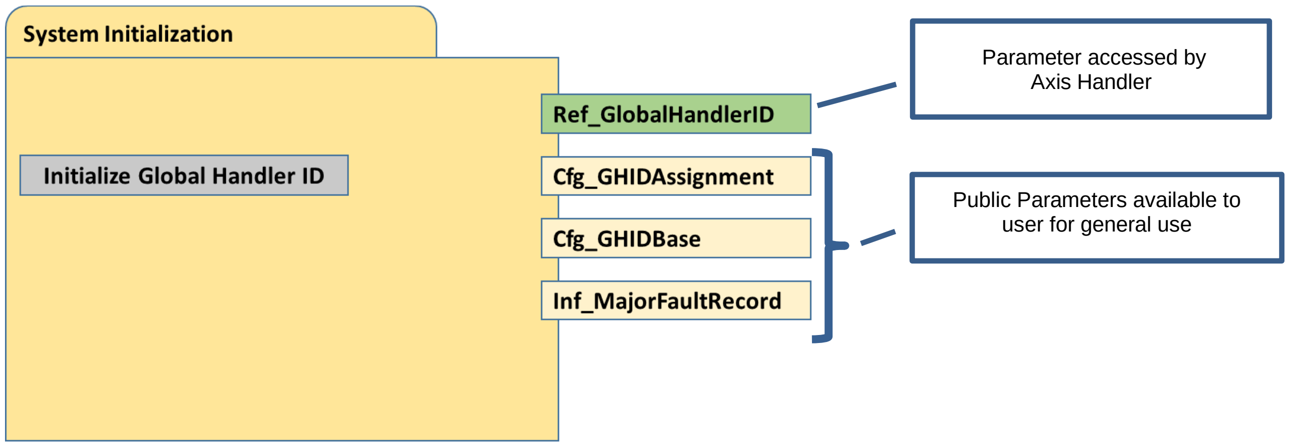
2 Axis Handler Architecture

2.1 Understanding Axis Handler Components

The Axis Handler requires three components to exist in the PAC: System Initialization Program, CIP Drive Handler Language Pack Program, and Axis Handler CIP Drive instruction and data structure

2.1.1 System Initialization (raM_Dvc_DH_SysIni)

- Program Folder
- Must be scheduled under Controller Power-Up Handler
- One instance per PAC is required
- Provides management of the Global Handler ID number at PAC power-up
- Global Handler ID Assignment supports unique ID in multiprocessor applications


System Initialization
Initialize Global Handler ID
Ref_GlobalHandlerID
Cfg_GHIDAssignment
Cfg_GHIDBase
Inf_MajorFaultRecord
Parameter accessed by Axis Handler
Public Parameters available to user for general use

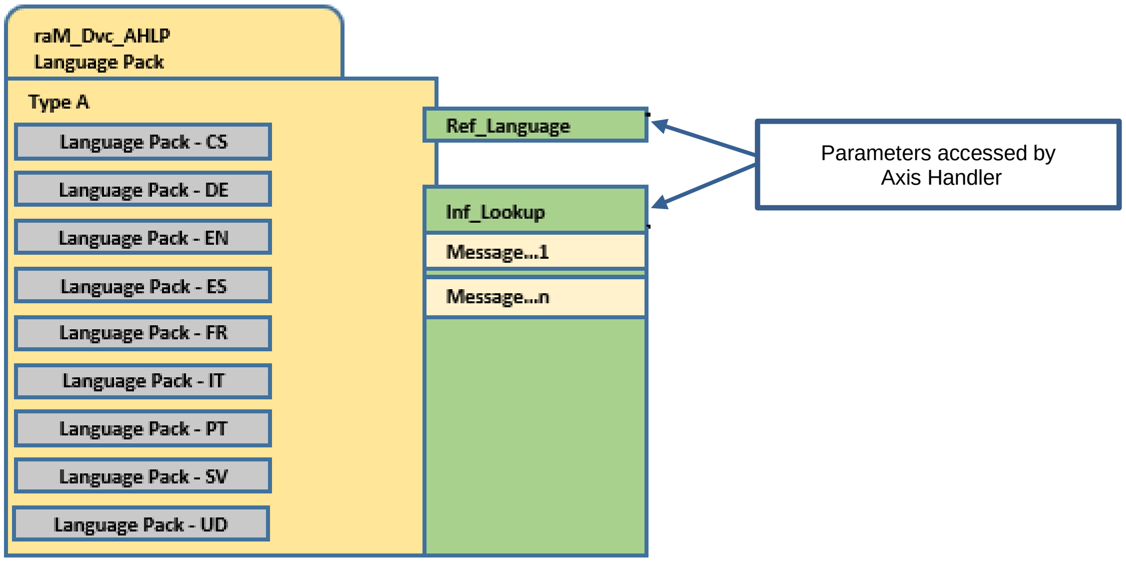
2.1.1.1 Public Program Parameters Tags

Tag Name	Usage	Data Type	Connection	Description
Ref_GlobalHandlerID	Public	DINT	Alias connection is made at the Axis handler instance	Global Axis Handler ID
Cfg_GHIDAssignment	Public	DINT	No connections assigned	Global Axis Handler ID Assignment Configuration 0 = System 1 = User
Cfg_GHIDBase	Public	DINT	No connections assigned	Global Axis Handler ID Numbering Start GHID Assignment Type 1
Inf_MajorFaultRecord	Public	raC_UDT_ControllerFaultRecord	No connections assigned	MajorFaultRecord attribute or MinorFault Record of the PROGRAM object

2.1.2 Axis Handler Language Pack (raM_Dvc_AHLP)

- Program Folder
- Scheduled for background operations; Language Pack only operates during handler language changes

- Logical Parent is user defined
- One instance per controller
- Provides language management for all handler event messages
- English is provided as the default language, but the following language add-ins are available:
    - PT - Portuguese
    - ES - Spanish
    - FR - French
    - IT - Italian
    - DE - German
    - SV – Swedish
    - CS – Czech
    - UD – User Defined
- User Defined Language allows user to add their desired language
    - Language must be compatible with ASCII characters
    - For more information please see Language Package Add-Ins Reference Manual

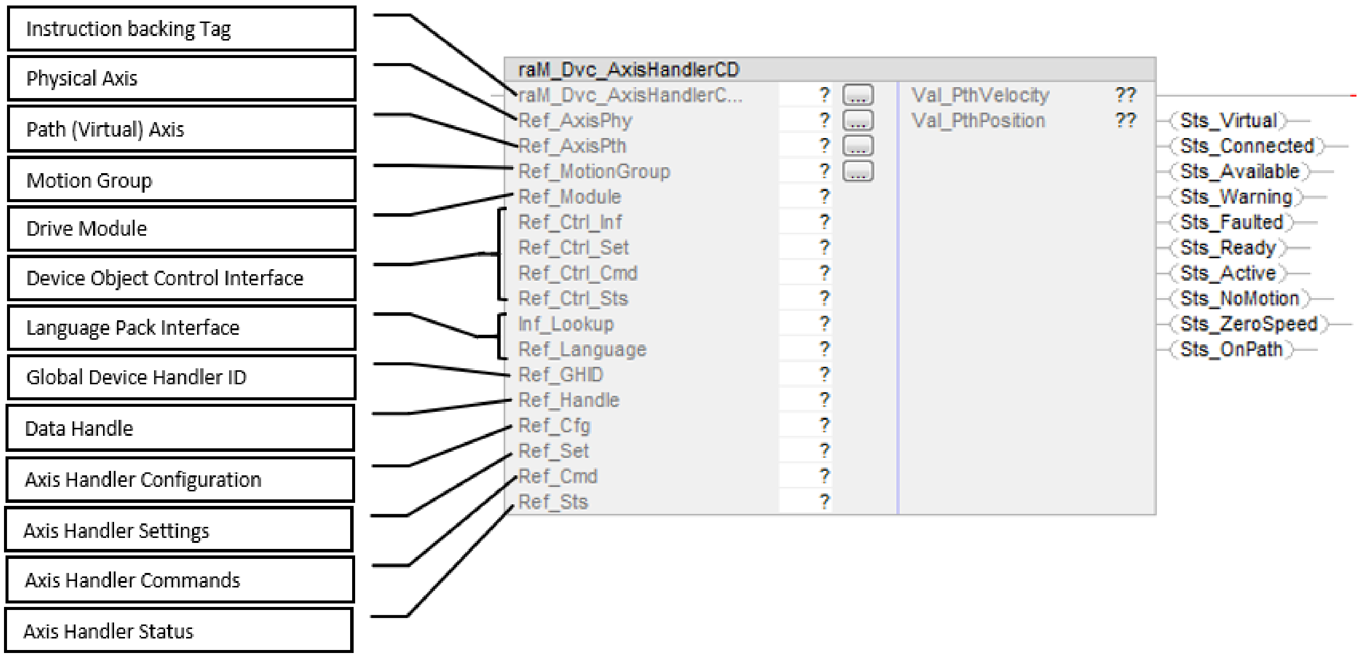


2.1.2.1 Public Program Parameters Tags

Tag Name	Usage	Data Type	Connection	Description
Ref_Language	Public	DINT	Alias connection is made at the axis handler instance	Language in Use
Inf_Lookup	Public	raM_UDT_LookupMember_STR0081[37]	Alias connection is made at the axis handler instance	Events Message in the Selected Language • Handler State • Handler Status • CIP Axis Faults • CIP Axis State • Motion Status

2.1.3 Axis Handler CIP Drive (raM_Dvc_AxisHandlerCD)

- Program with Add-on Instruction
- One instance per CIP Motion Axis
- Provides Device Management for CIP Motion Axes



2.1.3.1 InOut Data

InOut	Function / Description	Data Type
Ref_AxisPhy	Reference to CIP Axis in Motion Group	AXIS_CIP_DRIVE
Ref_AxisPth	Reference to Virtual Axis in Motion Group	AXIS_VIRTUAL
Ref_MotionGroup	Reference to Motion Group in Project	MOTION_GROUP
Ref_Module	Reference to Hardware Module associated with Physical Axis	MODULE
Ref_Ctrl_Inf	Device Control Interface - Power Motion – Information. Used for Axis Handler and Device Object Interface.	raC_UDT_ItfAD_PwrMotion_Inf
Ref_Ctrl_Set	Device Control Interface - Power Motion – Settings. Used for Axis Handler and Device Object Interface.	raC_UDT_ItfAD_PwrMotion_Set
Ref_Ctrl_Cmd	Device Control Interface - Power Motion – Command. Used for Axis Handler and Device Object Interface.	raC_UDT_ItfAD_PwrMotion_Cmd
Ref_Ctrl_Sts	Device Control Interface - Power Motion – Status. Used for Axis Handler and Device Object Interface.	raC_UDT_ItfAD_PwrMotion_Sts
Inf_Lookup	Event Message in selected Language	raC_UDT_LookupMember_STR0082
Ref_Language	Language in Use	DINT
Ref_GHID	Global Axis Handler ID	DINT
Ref_Handle	Axis Handler – Data Handle. Primary Interface between methods and axis handler for method registry and even logging.	raM_UDT_Dvc_xADH_DataHndl
Ref_Cfg	Axis Handler - Configuration	raM_UDT_Dvc_xADH_Configuration
Ref_Set	Axis Handler – Settings	raM_UDT_Dvc_xADH_Setting
Ref_Cmd	Axis Handler – Command	raM_UDT_Dvc_xADH_Command
Ref_Sts	Axis Handler - Status	raM_UDT_Dvc_xADH_Status

2.1.3.1.1 Ref_Cfg Members

Members	Function / Description	Data Type
MethodError	Method Error Interpretation – Enumerated. 0 = Warning Event. 1 = Fault Event	DINT
bCfg	Configuration (Bit Overlay)	DINT
LogHandlerState	1= Enable logging of handler state events into the queue	BOOL
LogMotionStatus	1 = Enable logging of Motion status events into the queue	BOOL
LogAxisState	1 = Enable logging of physical axis state events into the queue	BOOL

2.1.3.1.2 Ref_Set Members

Members	Function / Description	Data Type
ZeroSpeedTol	Tolerance for Zero Speed status bit	REAL
OnPathTol	Command Speed tolerance for OnPath status bit	REAL
Language	Language Number	DINT
Best	Settings (Bit Overlay)	DINT
InhibitCmd	1 = Inhibit user Commands from external sources; 0 = Allow	BOOL
InhibitSet	1 = Inhibit user settings from external sources; 0 = Allow	BOOL
InihibtCfg	1 = Inhibit user Configuration from external sources; 0 = Allow	BOOL

2.1.3.1.3 Ref_Cmd Members

Members	Function / Description	Data Type
bCmd	Commands (Bit Overlay)	DINT
Physical	1= Request Physical Device operation	BOOL
Virtual	1 = Request Virtual Device operation	BOOL
Energize	1 = Request Energize (activate) axis output power structure	BOOL
DeEnergize	1 = Request DeEnergize (de-activate) axis output power structure	BOOL
ResetFault	1 = Request Fault reset of axis and drive	BOOL

2.1.3.1.4 Ref_Sts Members

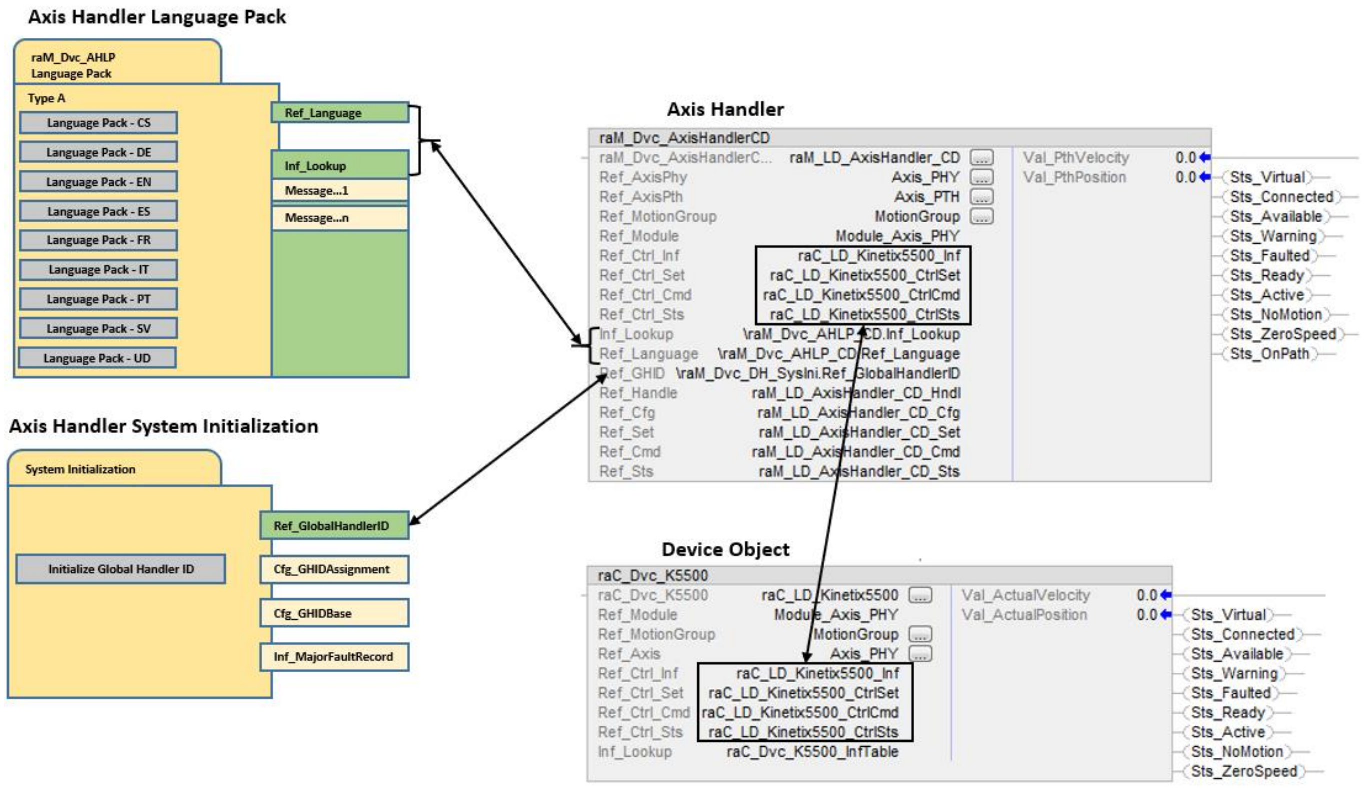
Members	Function / Description	Data Type
bSts	Status (Bit Overlay)	DINT
Physical	1 = Controlling Physical Device	BOOL
Virtual	1 = Controlling Virtual Device	BOOL
Connected	1 = Handler connection to device made	BOOL
Available	1 = Device is available for user interaction	BOOL
Ready	1 = Axis is ready for motion	BOOL
Faulted	1 = Device is in a faulted state	BOOL
Energized	1 = Drive output power structure is active	BOOL
Referenced	1 = Axis position has been referenced	BOOL
ZeroSpeed	1 = Axis is not moving. Velocity is 0 +/- zero velocity tolerance	BOOL
NoMotion	1 = Axis is at ZeroSpeed and no motion instructions active	BOOL
Recoverable	1 = Physical axis can synchronize to path Axis	BOOL
OnPath	1 = Physical axis is synchronized to path axis	BOOL
FirstFault	Axis Handler First-Out Fault	raM_UDT_OprEventCreate_Members
EventLog	Axis Handler Event Log	raM_UDT_OprEventList_Members
FaultLog	Axis Handler Fault Log	raM_UDT_OprEventList_Members

2.1.3.2 Output Data

Output	Function / Description	Data Type
Sts_Virtual	Device Status – Virtual	BOOL
Sts_Connected	Device is connected to the Programmable Controller	BOOL
Sts_Available	Device is available for interaction with user code	BOOL
Sts_Warning	Active Alarm or Warning Exists	BOOL
Sts_Faulted	Active Fault exists	BOOL
Sts_Ready	Device is ready to be enabled	BOOL
Sts_Active	Device Power Structure is enabled	BOOL
Sts_NoMotion	Device is active with no active motion instruction	BOOL
Sts_ZeroSpeed	Device is active and at zero velocity	BOOL
Sts_OnPath	Physical Axis is in synch with Path Axis (Gearing engaged)	BOOL
Val_PthVelocity	Path Axis Actual Velocity	REAL
Val_PthPosition	Path Axis Actual Position	REAL
raM_Dvc_AxisHandler_CIP	Device Identifier Bit	BOOL

2.2 Connecting Axis Handler Components

2.2.1 Single Instance



raM_Dvc_AxisHandlerCD

raM_Dvc_AxisHandlerC...	raM_LD_AxisHandler_CD
Ref_AxisPhy	Axis_PHY
Ref_AxisPth	Axis_PTH
Ref_MotionGroup	MotionGroup
Ref_Module	Module_Axis_PHY
Ref_Ctrl_Inf	raC_LD_Kinetix5500_Inf
Ref_Ctrl_Set	raC_LD_Kinetix5500_CtrlSet
Ref_Ctrl_Cmd	raC_LD_Kinetix5500_CtrlCmd
Ref_Ctrl_Sts	raC_LD_Kinetix5500_CtrlSts
Inf_Lookup	\raM_Dvc_AHLP_CD.Inf_Lookup
Ref_Language	\raM_Dvc_AHLP_CD.Ref_Language
Ref_GHID	\raM_Dvc_DH_SysIni.Ref_GlobalHandlerID
Ref_Handle	raM_LD_AxisHandler_CD_Hndl
Ref_Cfg	raM_LD_AxisHandler_CD_Cfg
Ref_Set	raM_LD_AxisHandler_CD_Set
Ref_Cmd	raM_LD_AxisHandler_CD_Cmd
Ref_Sts	raM_LD_AxisHandler_CD_Sts

Val_PthVelocity	0.0
Val_PthPosition	0.0

Sts_Virtual
Sts_Connected
Sts_Available
Sts_Warning
Sts_Faulted
Sts_Ready
Sts_Active
Sts_NoMotion
Sts_ZeroSpeed
Sts_OnPath

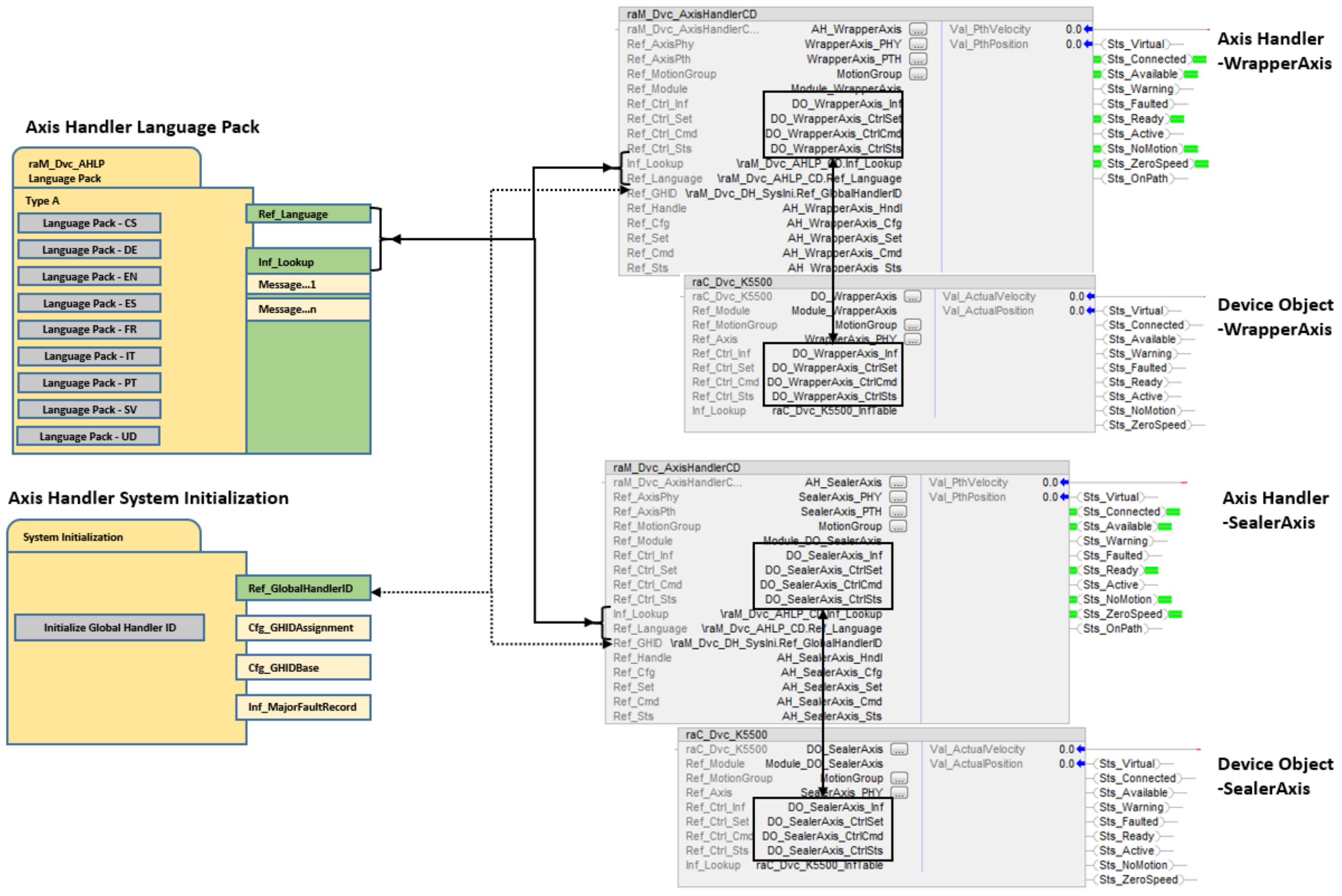
raC_Dvc_K5500

raC_Dvc_K5500	raC_LD_Kinetix5500
Ref_Module	Module_Axis_PHY
Ref_MotionGroup	MotionGroup
Ref_Axis	Axis_PHY
Ref_Ctrl_Inf	raC_LD_Kinetix5500_Inf
Ref_Ctrl_Set	raC_LD_Kinetix5500_CtrlSet
Ref_Ctrl_Cmd	raC_LD_Kinetix5500_CtrlCmd
Ref_Ctrl_Sts	raC_LD_Kinetix5500_CtrlSts
Inf_Lookup	raC_Dvc_K5500_InfTable

Val_ActualVelocity	0.0
Val_ActualPosition	0.0

Sts_Virtual
Sts_Connected
Sts_Available
Sts_Warning
Sts_Faulted
Sts_Ready
Sts_Active
Sts_NoMotion
Sts_ZeroSpeed

2.2.2 Multiple Instances (Motion Axes / Group Not Shown)

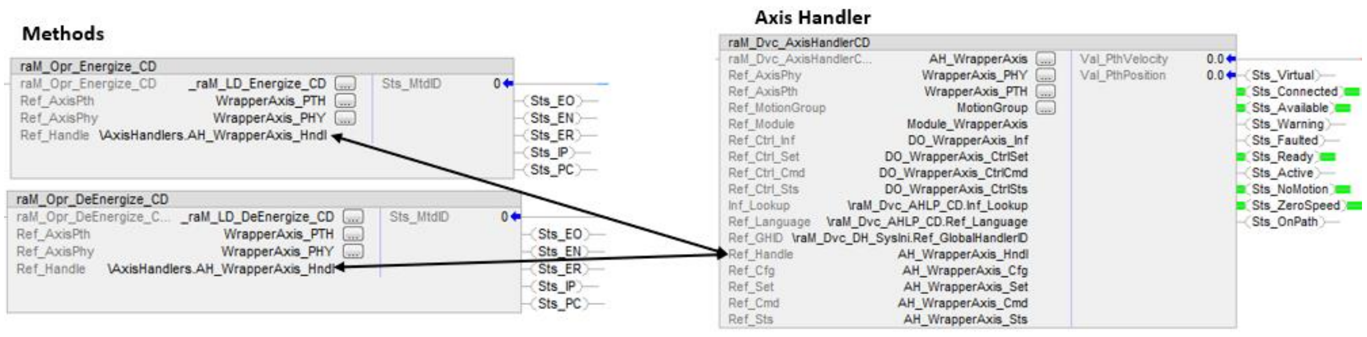


2.3 Connecting to the Application

An Axis Handler for an axis must also connect to the following:

- Methods designed for use with the axis handler
- User Application Code
- Axis Handler Status and Command bits to be used in user code

These connections are made using Direct Access Parameters.

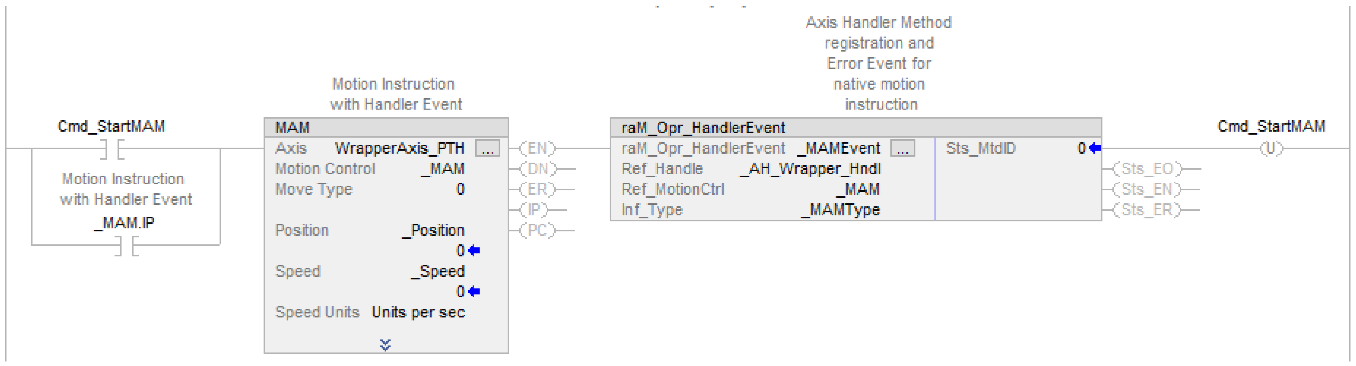


```
raM_Opr_Energize_CD
raM_Opr_Energize_CD   _raM_LD_Energize_CD          Sts_MtdID   0
Ref_AxisPth           WrapperAxis_PTH
Ref_AxisPhy           WrapperAxis_PHY
Ref_Handle   \AxisHandlers.AH_WrapperAxis_Hndl

raM_Opr_DeEnergize_CD
raM_Opr_DeEnergize_C...  _raM_LD_DeEnergize_CD     Sts_MtdID   0
Ref_AxisPth           WrapperAxis_PTH
Ref_AxisPhy           WrapperAxis_PHY
Ref_Handle   \AxisHandlers.AH_WrapperAxis_Hndl

raM_Dvc_AxisHandlerCD
raM_Dvc_AxisHandlerC...          AH_WrapperAxis    Val_PthVelocity   0.0
Ref_AxisPhy                      WrapperAxis_PHY   Val_PthPosition   0.0
Ref_AxisPth                      WrapperAxis_PTH
Ref_MotionGroup                  MotionGroup
Ref_Module                       Module_WrapperAxis
Ref_Ctrl_Inf                     DO_WrapperAxis_Inf
Ref_Ctrl_Set                     DO_WrapperAxis_CtrlSet
Ref_Ctrl_Cmd                     DO_WrapperAxis_CtrlCmd
Ref_Ctrl_Sts                     DO_WrapperAxis_CtrlSts
Inf_Lookup                       \raM_Dvc_AHLP_CD.Inf_Lookup
Ref_Language                     \raM_Dvc_AHLP_CD.Ref_Language
Ref_GHID                         \raM_Dvc_DH_SysIni.Ref_GlobalHandlerID
Ref_Handle                       AH_WrapperAxis_Hndl
Ref_Cfg                          AH_WrapperAxis_Cfg
Ref_Set                          AH_WrapperAxis_Set
Ref_Cmd                          AH_WrapperAxis_Cmd
Ref_Sts                          AH_WrapperAxis_Sts
```

Apart from Methods, native motion instructions can also be used along with Axis Handler. The handler event instruction allows user to log invocation and error events in the handler queue when used with an accompanying Logix native motion instruction. Additionally, the Handler Event will register the motion instruction as if it was a regular handler method. As shown below the HandlerEvent instruction is used along with native MAM instruction. Event in Axis Handler Event queue will be logged when the MAM instruction is triggered and when MAM instruction has error.


Axis Handler Method registration and Error Event for native motion instruction
Motion Instruction with Handler Event
Cmd_StartMAM
Motion Instruction with Handler Event
_MAM.IP
EN
DN
ER
IP
PC
Sts_EO
Sts_EN
Sts_ER
Cmd_StartMAM
U

```
MAM
Axis            WrapperAxis_PTH
Motion Control  _MAM
Move Type       0
Position        _Position
                0
Speed           _Speed
                0
Speed Units     Units per sec

raM_Opr_HandlerEvent
raM_Opr_HandlerEvent   _MAMEvent       Sts_MtdID   0
Ref_Handle             _AH_Wrapper_Hndl
Ref_MotionCtrl         _MAM
Inf_Type               _MAMType
```

raM_LD_MotionInstructionGeneral and raM_ST_MotionInstructionGeneral are the library objects which consists of implement code for raM_Opr_HandlerEvent and native motion instruction.

Note: OnPath status of Axis Handler must be monitored in user application code to ensure that Physical Axis is linked to the Path Axis. The OnPath status is set when Physical Axis is energized and in synchronized with Path Axis. The OnPath status will reset if the Path Axis to Physical Axis gearing is terminated or if the difference in position of Path Axis o Physical Axis is greater than the defined limit. If OnPath status is lost while the Path Axis is in motion, the application code must have an independent mechanism to stop the Path Axis and resynchronize the axes.

3 Handler Operation

3.1 Axis Handler Modes

The axis handler and device operation are defined by the following modes:

- Physical
- Virtual

Handler Operation:

- Program: If available, commands other than Mode commands are accepted through the Ref_Cmd parameter
- Operator: Operator commands accepted through the HMI faceplate

PAC Power-Up Defaults:

- Program -Physical

Device Operation:

- Physicalize
  - Axis Handler and Device Object are configured to operate in a 'physical' capacity. I/O modules must be connected, and valid configuration of the physical axis are required before operation is allowed.
- Virtualize
  - Axis Handler and Device Object configured to 'virtualize' the physical device. I/O module connection is not required.
  - Support for Axis Virtualization requires the Handler to be used in conjunction with axis methods for any function that affects device state or operate directly on the device.
    - Ex. Servo On, Servo Off, and Axis Registration.
  - Support for Axis Virtualization requires the Handler Path Axis to be used as the primary path planning axis.

Switching Modes:

- Both handler operation and device operation can be selected through the Ref_Cmd parameter
- Requests for both handler operation and device operation are made independently
- Mode changes can only be made when the handler Sts_Active is 0, meaning drive power structure is de-energized.

3.2 Axis Handler Transition Rules

The following table describes the rules that define when the mode transition is allowed:

Current Mode	Next Mode	Conditions
Physicalize	Virtualize	➔ Zero Speed on both Physical and Path Axis ➔ Physical Axis DeEnergized
Virtualize	Physicalize	➔ Zero Speed on both Physical and Path Axis ➔ Physical Axis DeEnergized

3.3 Axis Handler States

The following table describes the states that define the axis handler behavior:

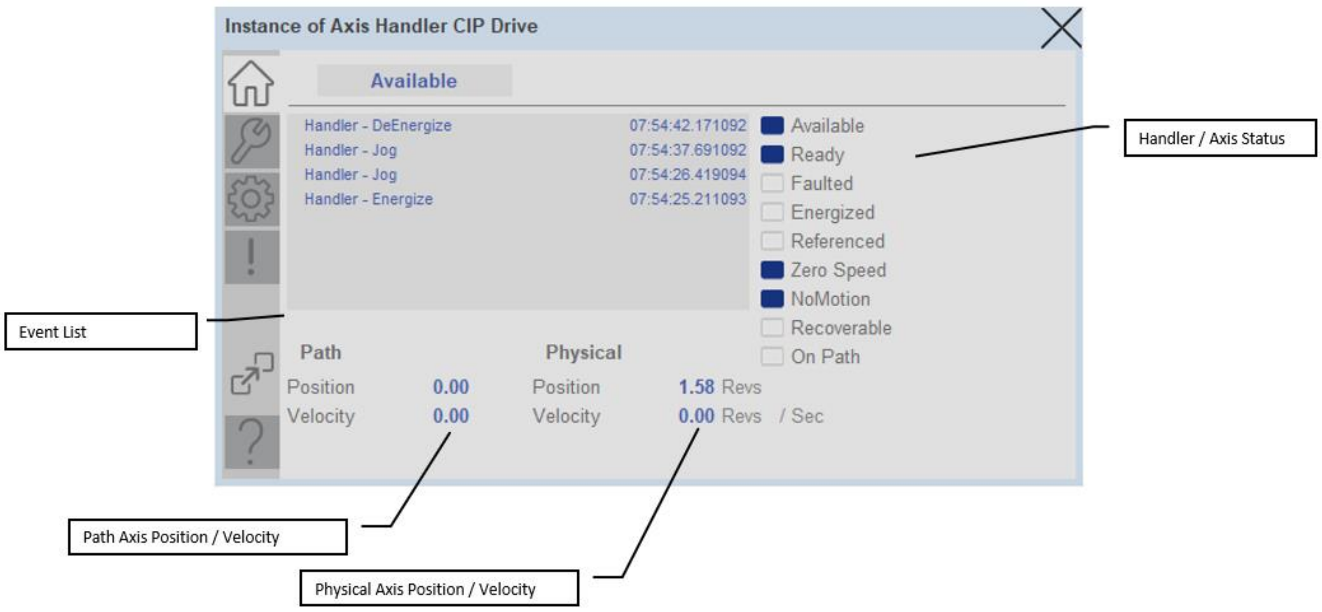
Name	Value	Description
Initializing	1	Initial state at controller power up. ➔ Acquire Handler ID ➔ Clear Method Registry ➔ Configure Initial Module State ➔ Initialize Handler Data Interface
Disconnected	2	Axis Handler is disconnected ➔ No Action
Disconnecting	3	Axis Handler and Device Object changing mode from Physical to Virtual or Virtual to Physical. Or Device object has lost connection to module.
Connecting	4	Device Object connecting to Module. ➔ Verify Device Network Connection
Idle	5	Device is connected. ➔ No Action
Configuring	6	Axis Handler is configuring device for operation. ➔ Motion Group Synchronization Confirmation ➔ Axis Configuration (system) ➔ Path Axis Configuration
Available	7	Axis Handler is Running. ➔ Managed Device Connected and Configured

4 User Interface

Tab descriptions:

- Home – Displays the Event List, Handler and Axis status, and both physical and path axis position and velocity.
- Settings – Fault clearing, Energize/De-energize, Axis jogging, and Language selection.
- Configuration – Device operation selects (Physicalized/Virtualized), detailed Event List, Method registry, Event List customization, jog dynamics, fault clearing, and Event List clearing.
- Alarm/Faults – Timestamped and consolidated alarms and faults.

4.1 Home



Handler - DeEnergize 07:54:42.171092
Handler - Jog 07:54:37.691092
Handler - Jog 07:54:26.419094
Handler - Energize 07:54:25.211093

Path Position 0.00, Velocity 0.00; Physical Position 1.58 Revs, Velocity 0.00 Revs / Sec

The Home tab offers a variety of status indicators that are connected directly to the Axis Handler data structure. These indicators can be utilized for quick reference while troubleshooting during a post-commissioning scenario.

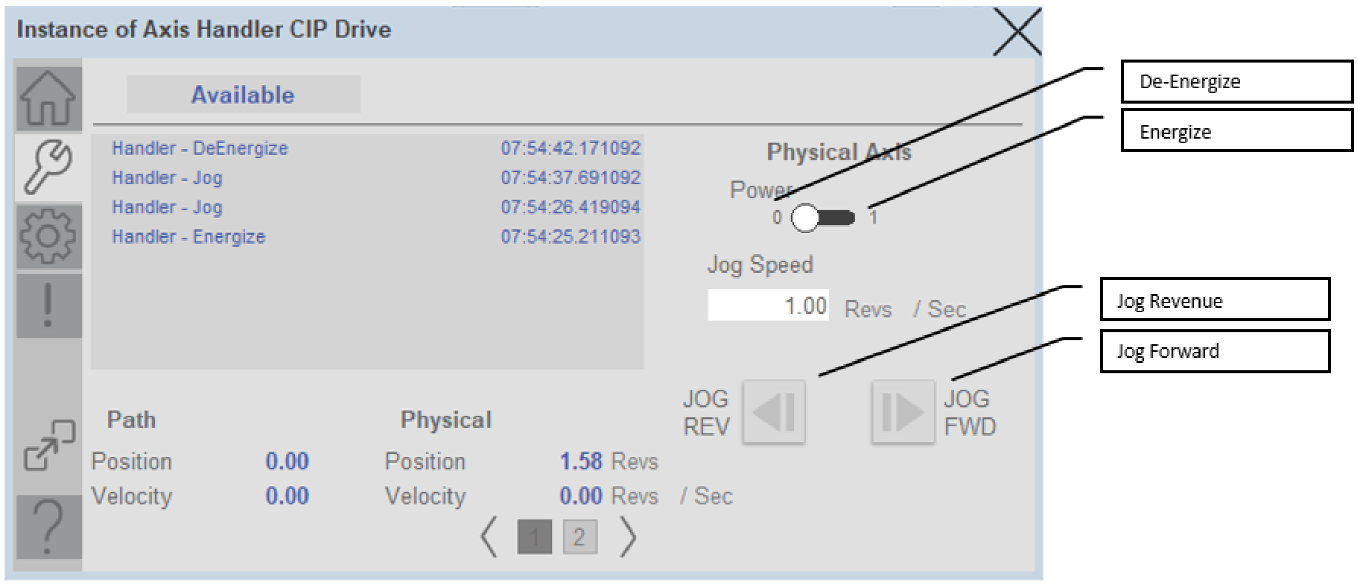
Below is an explanation of each indicator for quick reference:

Indicator Name	Meaning
Ready	Device is connected, configured, and able to receive user commands.
Available	The device is ready for command and ready to execute motion commands. The Motion Group is synched, DC bus is up, device is not inhibited, it is not faulted, and not shut down.
Faulted	The device is faulted.
Energized	The device control loop and power structure is enabled.
Referenced	The physical axis has been homed.

Zero Speed	The device velocity is within user specified stand still limits. This can be changed on page 2 of the Configuration tab.
NoMotion	No motion instruction is currently being executed upon device.
Recoverable	The path axis has been synchronized to the physical axis during execution of the raM_Opr_SyncPthPhyAx_CD.
On Path	On Path : *-If AH Physicalized:* physical axis is geared to its path axis and its command position is matched to the path axis position (axis is energized) *-If AH Virtualized:* axis energized. <u>NOT</u> On Path : *-If AH Physicalized:* physical axis is not geared to its path axis or the axis is de-energized *-If AH Virtualized:* axis is de-energized.

4.2 Settings

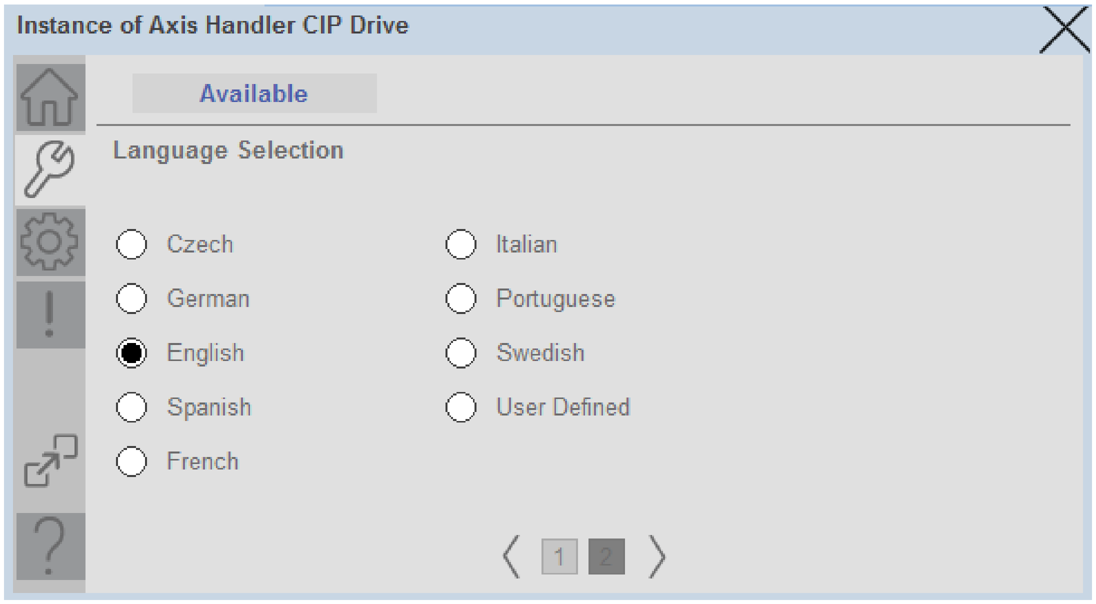
4.2.1 Page 1



Handler - DeEnergize	07:54:42.171092
Handler - Jog	07:54:37.691092
Handler - Jog	07:54:26.419094
Handler - Energize	07:54:25.211093

	Path	Physical
Position	0.00	1.58 Revs
Velocity	0.00	0.00 Revs / Sec

4.2.2 Page 2

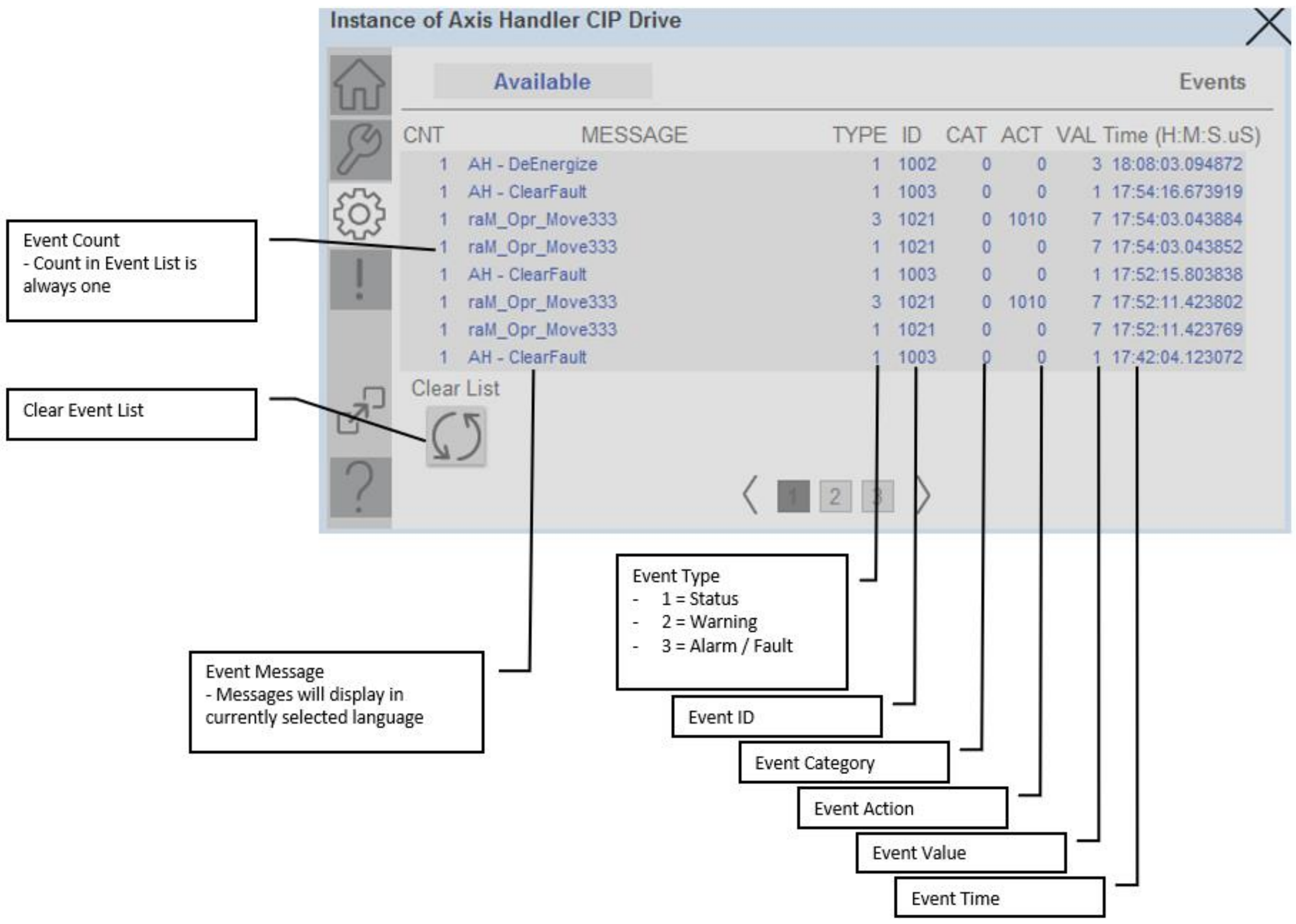


Czech [ ] Italian [ ]
German [ ] Portuguese [ ]
[✓ English] Swedish [ ]
Spanish [ ] User Defined [ ]
French [ ]

Note that the available languages can only be included from Application Code Manager during library object instantiation. If additional languages are desired at a later time, then Application Code Manager code regeneration is required. There is no configuration required on the faceplate to enable this functionality, only the languages included from Application Code Manager will be selectable on the display.

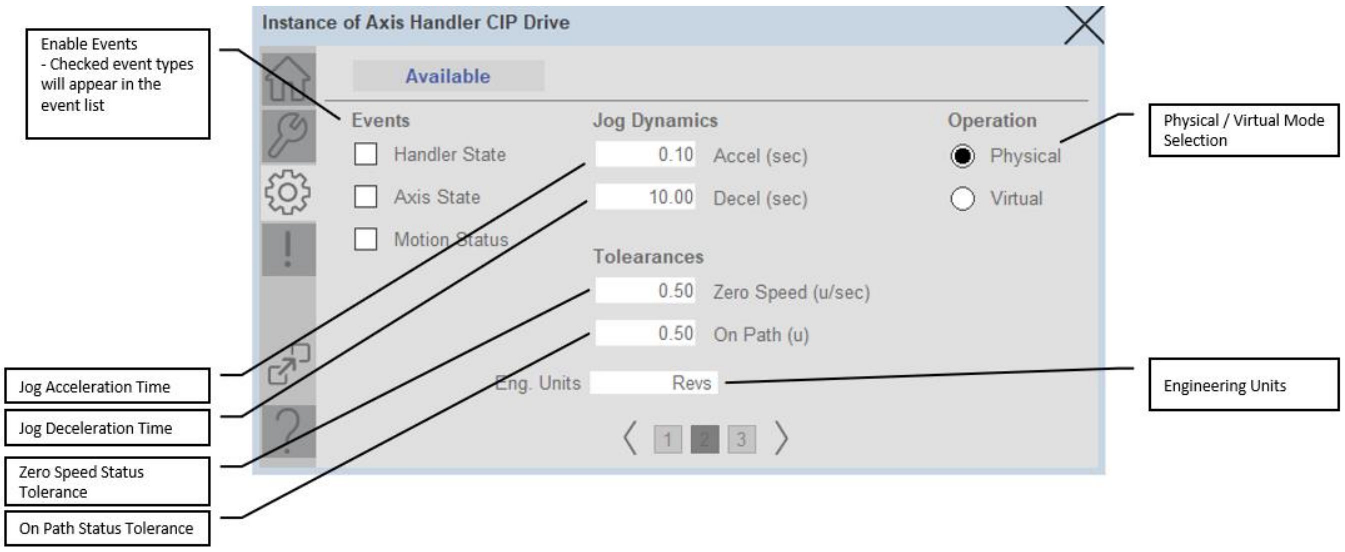
4.3 Configuration

4.3.1 Page 1



CNT	MESSAGE	TYPE	ID	CAT	ACT	VAL	Time (H:M:S.uS)
1	AH - DeEnergize	1	1002	0	0	3	18:08:03.094872
1	AH - ClearFault	1	1003	0	0	1	17:54:16.673919
1	raM_Opr_Move333	3	1021	0	1010	7	17:54:03.043884
1	raM_Opr_Move333	1	1021	0	0	7	17:54:03.043852
1	AH - ClearFault	1	1003	0	0	1	17:52:15.803838
1	raM_Opr_Move333	3	1021	0	1010	7	17:52:11.423802
1	raM_Opr_Move333	1	1021	0	0	7	17:52:11.423769
1	AH - ClearFault	1	1003	0	0	1	17:42:04.123072

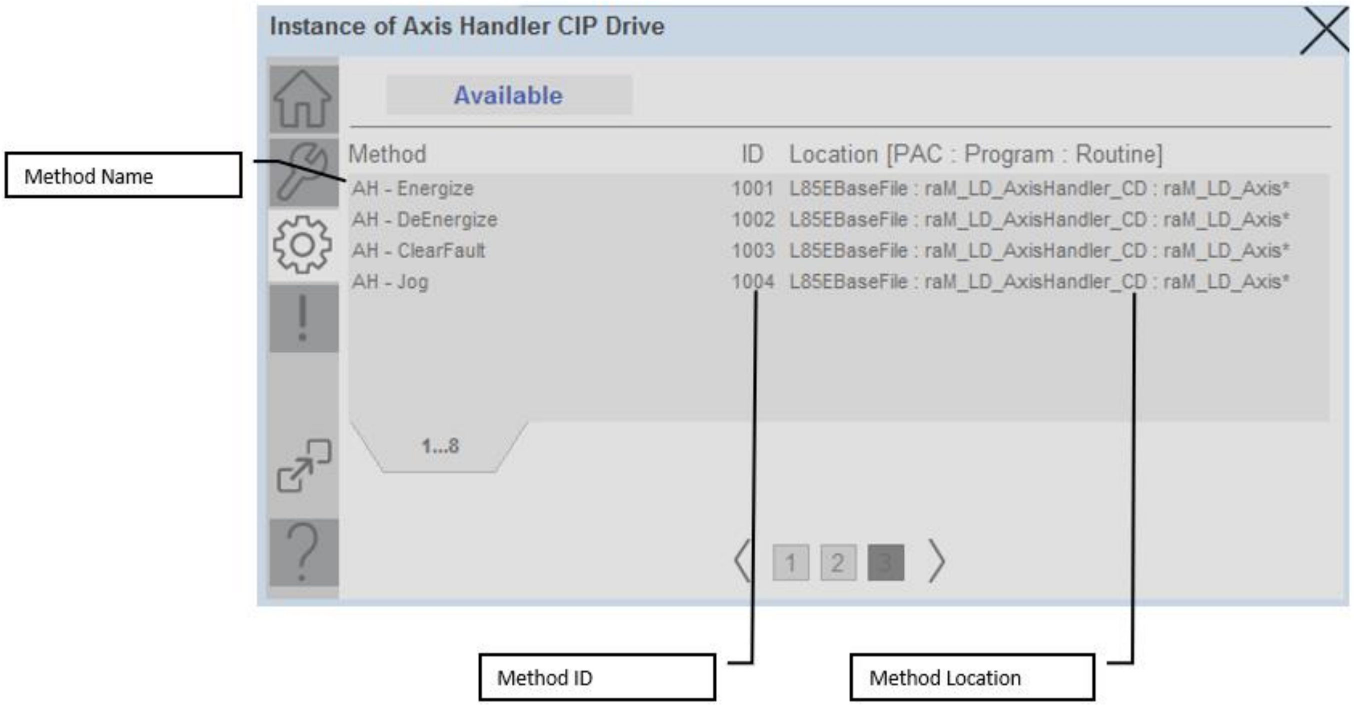
4.3.2 Page 2



Event category descriptions:

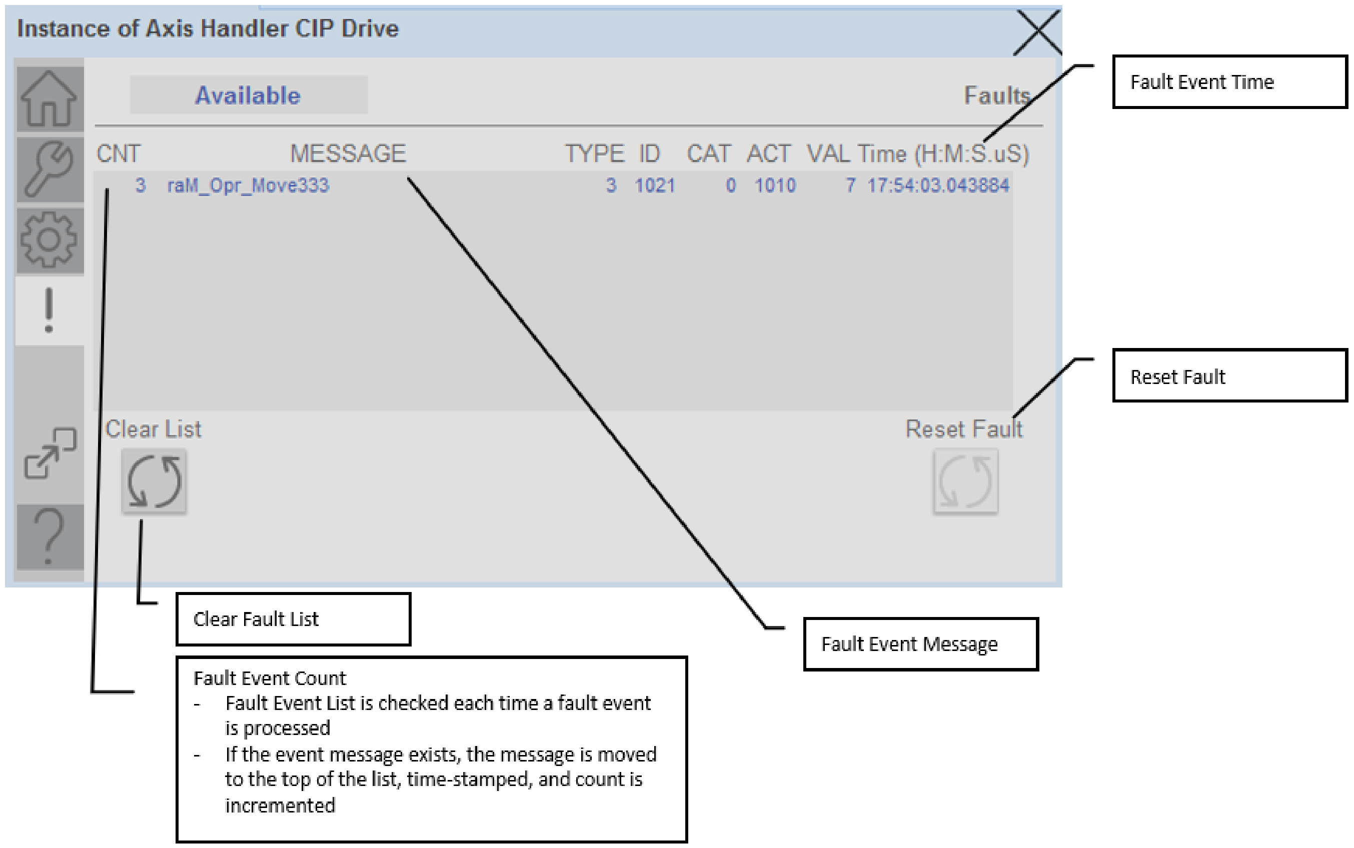
- <u>Handler Status</u> – Axis Handler Status change. Includes Method Registry Full.
- <u>Handler State</u> – Axis Handler State change. Exists as part of the Axis Handler data structure.
- <u>Axis State</u> – Captures changes in CIP Axis. This is independent of the Axis Handler data structure. If using AH methods these events are redundant. If using standard motion instructions, make sure this is selected.
- <u>Motion Status</u> – Motion Status of CIP axis. Like the CIP Axis State, this is independent of the Axis Handler data structure.
- Faults and Methods are always listed

4.3.3 Page 3



Method	ID	Location [PAC : Program : Routine]
AH - Energize	1001	L85EBaseFile : raM_LD_AxisHandler_CD : raM_LD_Axis*
AH - DeEnergize	1002	L85EBaseFile : raM_LD_AxisHandler_CD : raM_LD_Axis*
AH - ClearFault	1003	L85EBaseFile : raM_LD_AxisHandler_CD : raM_LD_Axis*
AH - Jog	1004	L85EBaseFile : raM_LD_AxisHandler_CD : raM_LD_Axis*

4.4 Fault



CNT	MESSAGE	TYPE	ID	CAT	ACT	VAL	Time (H:M:S.uS)
3	raM_Opr_Move333	3	1021	0	1010	7	17:54:03.043884

Fault Event Count
- Fault Event List is checked each time a fault event is processed
- If the event message exists, the message is moved to the top of the list, time-stamped, and count is incremented

4.5 <u>FactoryTalk View</u>

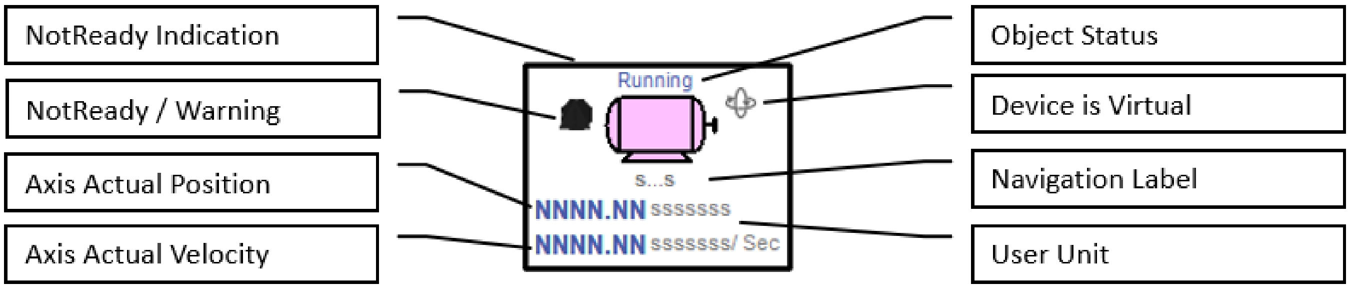
4.5.1 Images Files

The global objects and the display make heavy use of the image files developed for the Rockwell Automation Library. These image files must be made part of the HMI application if the pre-developed graphics are to be used.

4.5.2 Global Objects

4.5.2.1 (raM-xE) raM_Dvc_AxisHandlerCD_Global.ggfx

Global Object file with the "call-to-action" buttons available to go to the faceplate display.



4.6 View Designer

View Designer Faceplates are distributed as View Designer application files.
The visual components of the application are located in the User-Defined Screens folder as a Faceplate popup and a Tools screen.
This faceplate has ability to navigate to a secondary faceplate type. All faceplates in the supplied application and those in secondary faceplate application should be added to into user application. Following this, specific configuration must occur.

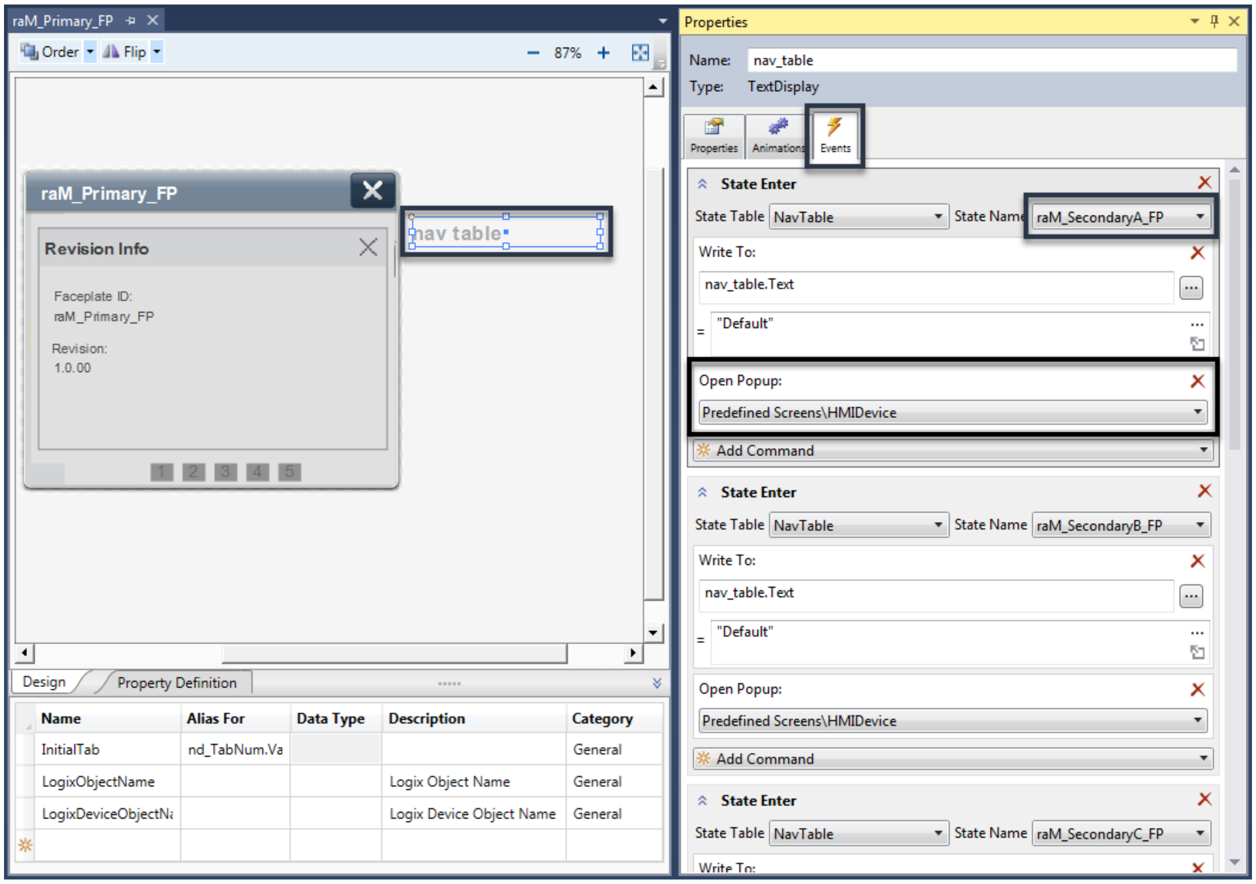
4.6.1 Add Faceplate and Tools

To add the faceplate into user application, copy it from the supplied application file and insert into user application by drag and drop or copy paste.
The Tools screen contains Faceplate launch button.
The supplied launch button is an example of how to navigate to the faceplate in a user application.
When copying a faceplate launch button into user application from the supplied application, the button loses the assigned Popup reference and Property configuration. These values must be manually restored for the button to operate properly.

4.6.2 Secondary Faceplate navigation setup

Navigation to a secondary faceplate requires identification of the secondary faceplate in “nav table” object.

1. Select the text labeled “nav_table” located to the right of the primary faceplate border.
2. In the Properties window select Events.
3. Select the State Enter Event with the State Name that corresponds to name of the secondary faceplate being configured.
   Note that the Open Popup configuration initially calls a predefined screen.



Name	Alias For	Data Type	Description	Category
InitialTab	nd_TabNum.Va			General
LogixObjectName			Logix Object Name	General
LogixDeviceObjectN:			Logix Device Object Name	General

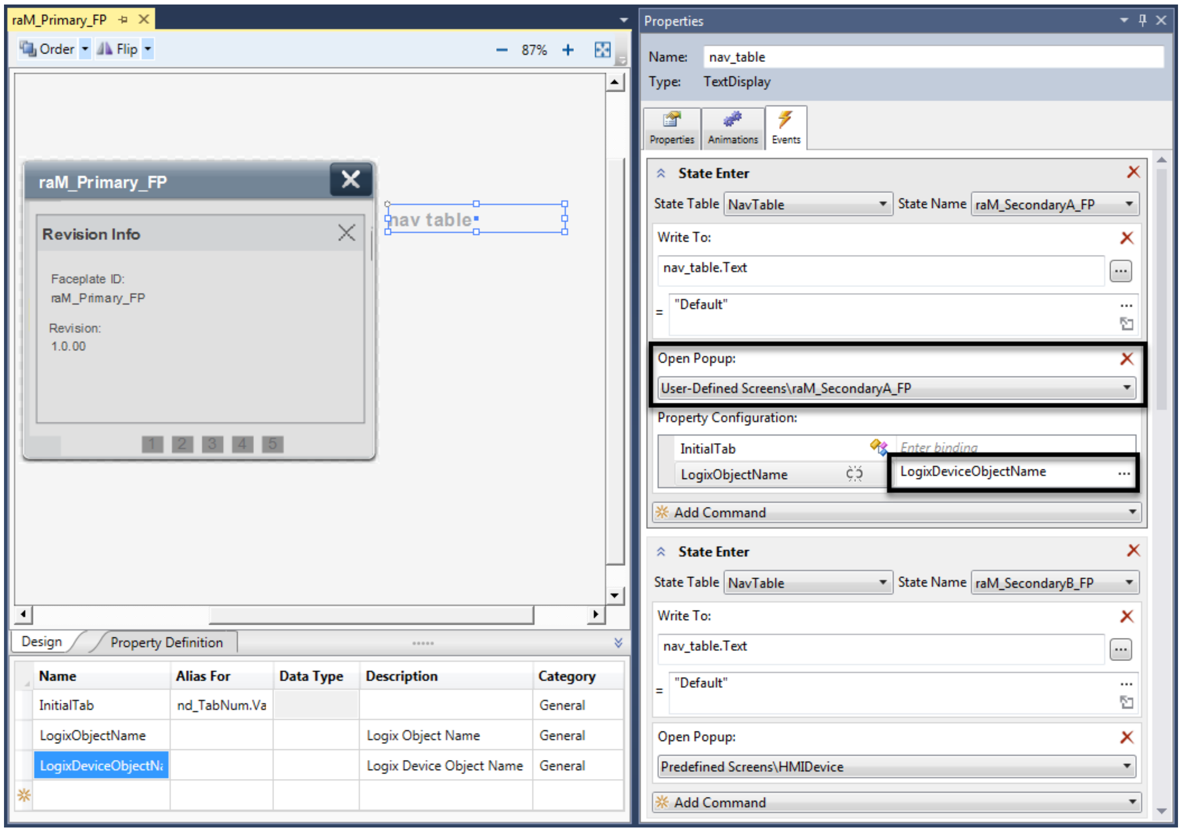
Name: nav_table
Type: TextDisplay

State Enter
State Table NavTable State Name raM_SecondaryA_FP
Write To:
nav_table.Text
= "Default"
Open Popup:
Predefined Screens\HMIDevice

State Enter
State Table NavTable State Name raM_SecondaryB_FP
Write To:
nav_table.Text
= "Default"
Open Popup:
Predefined Screens\HMIDevice

State Enter
State Table NavTable State Name raM_SecondaryC_FP
Write To:

4. Change the Open Popup configuration to reference the name of the secondary faceplate
5. In the Property Configuration, leave InitialTab blank and set LogixObjectName to “LogixDeviceObjectName”.



Name	Alias For	Data Type	Description	Category
InitialTab	nd_TabNum.Va			General
LogixObjectName			Logix Object Name	General
LogixDeviceObjectNa			Logix Device Object Name	General

Name: nav_table
Type: TextDisplay

State Enter
State Table: NavTable   State Name: raM_SecondaryA_FP
Write To: nav_table.Text
= "Default"
Open Popup: User-Defined Screens\raM_SecondaryA_FP
Property Configuration:

InitialTab	Enter binding
LogixObjectName	LogixDeviceObjectName

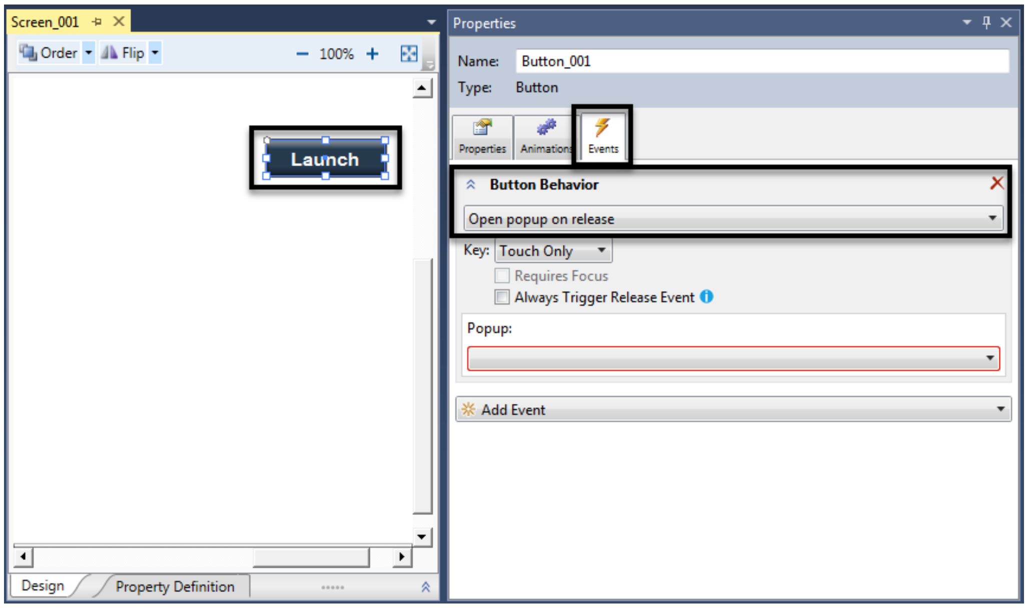
State Enter
State Table: NavTable   State Name: raM_SecondaryB_FP
Write To: nav_table.Text
= "Default"
Open Popup: Predefined Screens\HMIDevice

6. If multiple secondary faceplate options were added to the application, repeat steps 3 through 5 for each.

4.6.3 Configure Faceplate and Tools

To configure the Faceplate Launch Button:

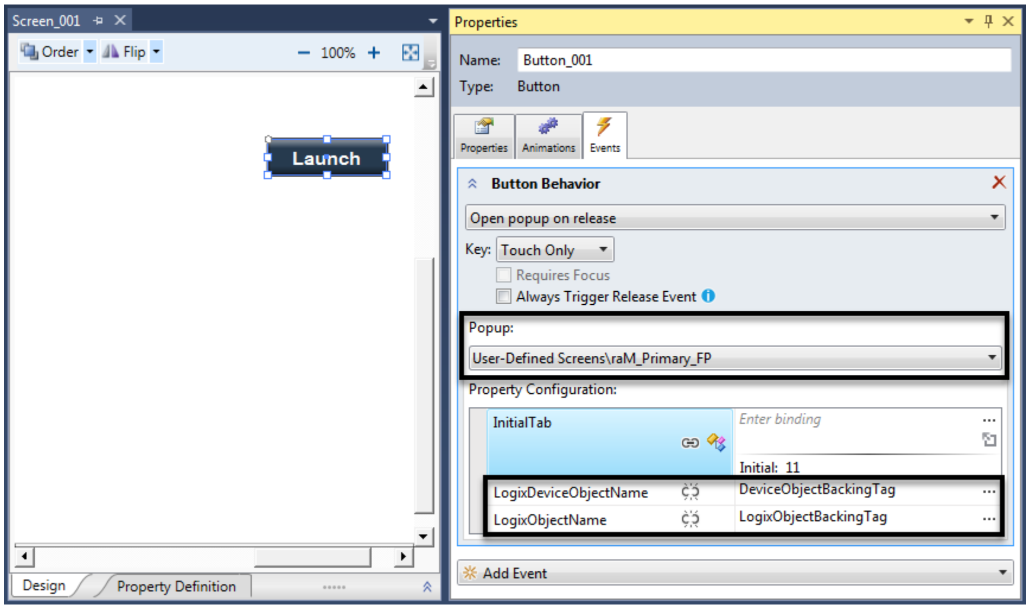
1. Select the Launch Button.
2. In the Properties window select Events.
3. Configure an Event as “Button Behavior” with “Open popup on release” option.



Name: Button_001
Type: Button
Open popup on release
Key: Touch Only
Requires Focus
Always Trigger Release Event
Popup:

4. Use dropdown menu under "Popup" to select the desired faceplate popup.

   Selecting the faceplate popup will make its Property Definitions appear under the Property Configuration section.
5. InitialTab binding must be left blank and initial value must be 11.
6. Use the ellipsis buttons in the LanguagePack, LogixDeviceObjectName and LogixObjectName properties and select Axis Handler Language Pack program, Logix Device Object backing tag and Logix Object backing tag respectively from the list of available tags.



Name: Button_001
Type: Button
Open popup on release
Key: Touch Only
Requires Focus
Always Trigger Release Event
Popup:
User-Defined Screens\raM_Primary_FP
Property Configuration:

InitialTab	Enter binding Initial: 11
LogixDeviceObjectName	DeviceObjectBackingTag
LogixObjectName	LogixObjectBackingTag

Notes:

1. View Designer Faceplates are designed for minimum 800x480 resolution. If screen is smaller, the faceplate will not fit on the display properly
2. User is responsible for resolving image name conflicts should any occur.
3. To remove navigation to secondary faceplate, delete the following items:
    a. "nav_toDeviceObject" PanelDeviceGroup
    b. "nav_table" TextDisplay
    c. "LogixDeviceObjectName" Property Definition from primary faceplate

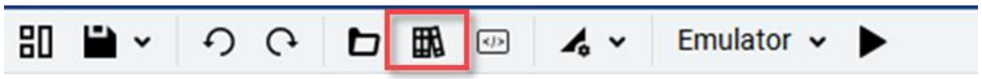
4.7 FactoryTalk Optix

FactoryTalk Optix faceplates are distributed as a library file, which is installed automatically when running Setup.cmd script in the Machine Builder Libraries download package.

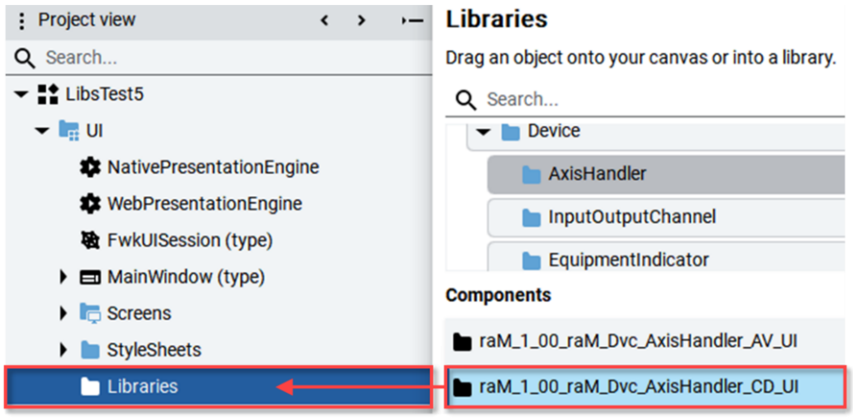
4.7.1 Add library into application

FactoryTalk Optix enables users to organize their projects according to their preferences. Typically, library files are inserted in the UI folder, adhering to the user's organizational structure.

To access libraries, click on the Template Libraries button in the Main toolbar.



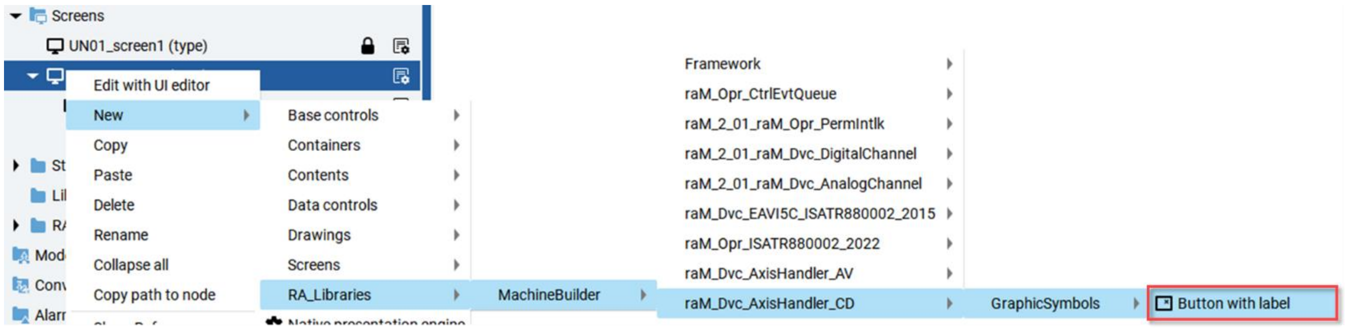
Select library to import from list of libraries in the popup screen, then drag it into the folder in the Project view. In the image below, the selected library is being dragged into the previously created 'Libraries' folder.



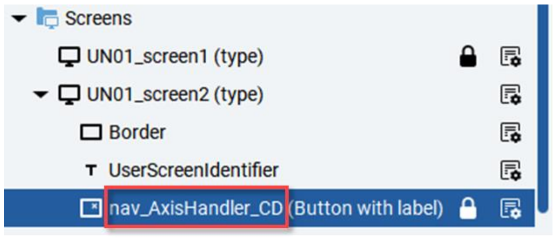
Note that because many libraries use common types, newly added components may include types that already exist in your application. If prompted for a type conflict, choose "Skip All" to use the existing type definitions.

4.7.2 Add faceplate graphic symbol into application

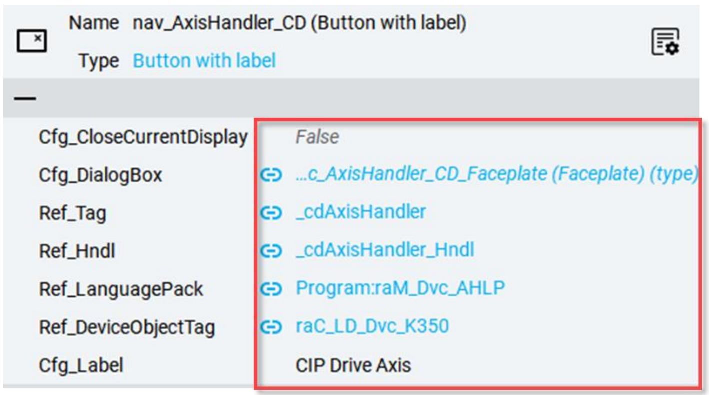
In Project view select container that will hold faceplate graphic symbol, right click on it, select New, then navigate to desired faceplate's graphic symbol.



Rename the instance of Graphic symbol as desired.



Double click on the container in Project view to open it in Editor pane, then position the graphic symbol as desired. Selecting the graphic symbol in Project view or in Editor pane shows its configuration parameters at the top of Properties pane.

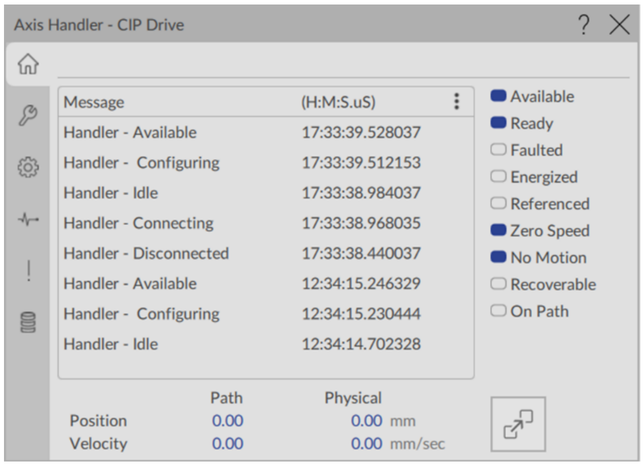


Name nav_AxisHandler_CD (Button with label)
Type Button with label

Cfg_CloseCurrentDisplay	False
Cfg_DialogBox	...c_AxisHandler_CD_Faceplate (Faceplate) (type)
Ref_Tag	_cdAxisHandler
Ref_Hndl	_cdAxisHandler_Hndl
Ref_LanguagePack	Program:raM_Dvc_AHLP
Ref_DeviceObjectTag	raC_LD_Dvc_K350
Cfg_Label	CIP Drive Axis

Enter configuration parameters and run your application to test.

Use Home tab to view current values and statuses, and recent events



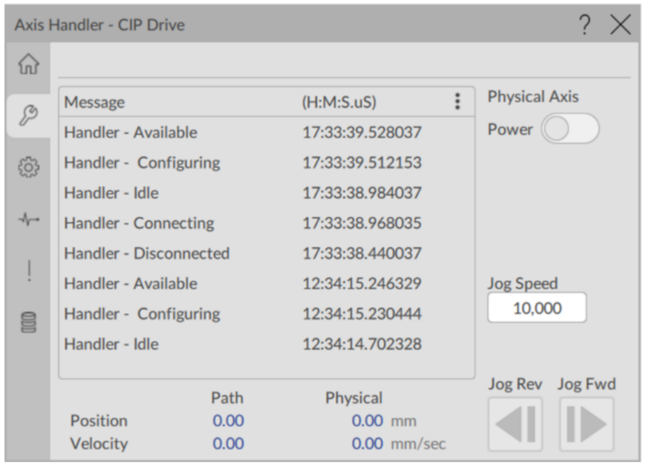
Axis Handler - CIP Drive

Message	(H:M:S.uS)
Handler - Available	17:33:39.528037
Handler - Configuring	17:33:39.512153
Handler - Idle	17:33:38.984037
Handler - Connecting	17:33:38.968035
Handler - Disconnected	17:33:38.440037
Handler - Available	12:34:15.246329
Handler - Configuring	12:34:15.230444
Handler - Idle	12:34:14.702328

- ● Available
- ● Ready
- ○ Faulted
- ○ Energized
- ○ Referenced
- ● Zero Speed
- ● No Motion
- ○ Recoverable
- ○ On Path

	Path	Physical
Position	0.00	0.00 mm
Velocity	0.00	0.00 mm/sec

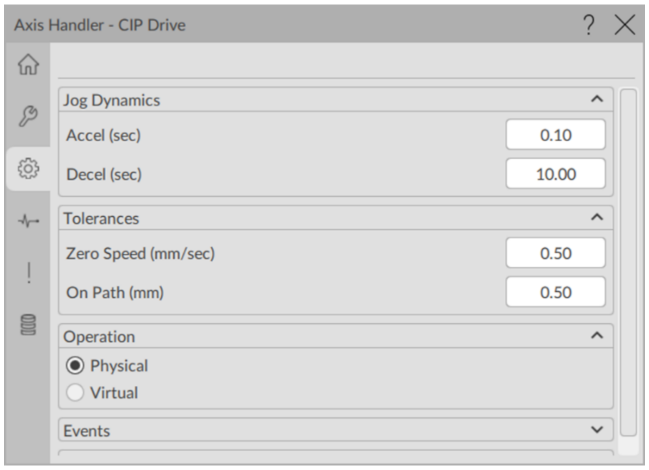
Use Settings tab to manually manipulate axis



Message	(H:M:S.uS)
Handler - Available	17:33:39.528037
Handler - Configuring	17:33:39.512153
Handler - Idle	17:33:38.984037
Handler - Connecting	17:33:38.968035
Handler - Disconnected	17:33:38.440037
Handler - Available	12:34:15.246329
Handler - Configuring	12:34:15.230444
Handler - Idle	12:34:14.702328

	Path	Physical	
Position	0.00	0.00	mm
Velocity	0.00	0.00	mm/sec

Use Engineering tab to configure object operation.



Jog Dynamics

Accel (sec)	0.10
Decel (sec)	10.00

Tolerances

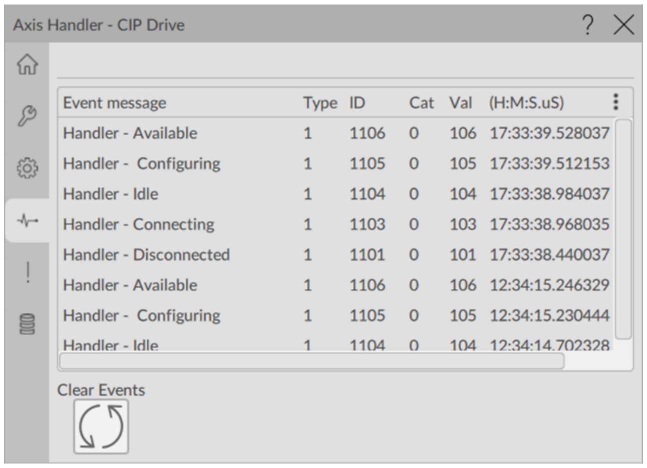
Zero Speed (mm/sec)	0.50
On Path (mm)	0.50

Operation

[✓ Physical] Virtual

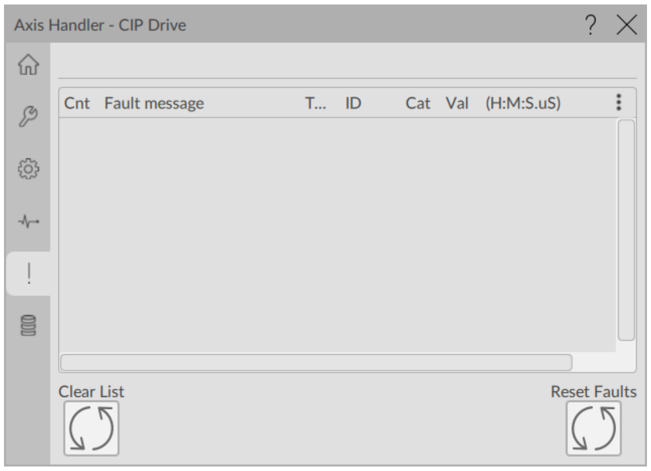
Events

Use Diagnostics tab to view recent events in sequential order.



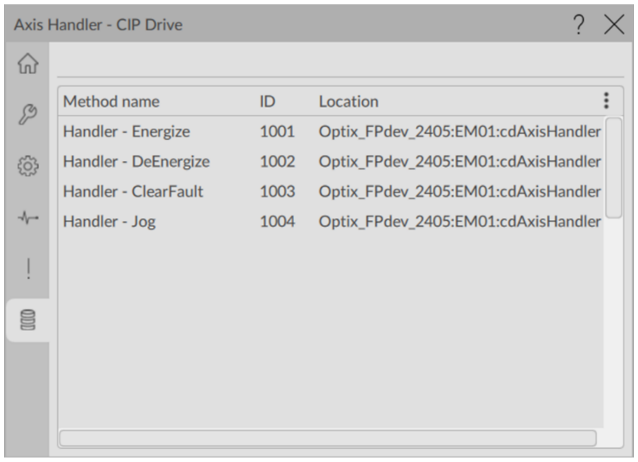
Event message	Type	ID	Cat	Val	(H:M:S.uS)
Handler - Available	1	1106	0	106	17:33:39.528037
Handler - Configuring	1	1105	0	105	17:33:39.512153
Handler - Idle	1	1104	0	104	17:33:38.984037
Handler - Connecting	1	1103	0	103	17:33:38.968035
Handler - Disconnected	1	1101	0	101	17:33:38.440037
Handler - Available	1	1106	0	106	12:34:15.246329
Handler - Configuring	1	1105	0	105	12:34:15.230444
Handler - Idle	1	1104	0	104	12:34:14.702328

Use Faults tab to view latest faults showing first fault on top.



Cnt	Fault message	T...	ID	Cat	Val	(H:M:S.uS)

Use Method Registry tab to find location of each method in Logix program.



Method name	ID	Location
Handler - Energize	1001	Optix_FPdev_2405:EM01:cdAxisHandler
Handler - DeEnergize	1002	Optix_FPdev_2405:EM01:cdAxisHandler
Handler - ClearFault	1003	Optix_FPdev_2405:EM01:cdAxisHandler
Handler - Jog	1004	Optix_FPdev_2405:EM01:cdAxisHandler

5 Events

Note: All Event IDs will be displayed with the Handler ID preceding the values listed.

Event ID = [Handler ID][Event Value]

Example:
Handler ID = 7
Event Value = 501
Event ID = 7501

5.1 Status

Axis Handler – State	Event Message	Event Type	Event Value
*.eState = 1	Handler – Initializing	1 (Status)	101
*.eState = 2	Handler – Disconnected	1 (Status)	102
*.eState = 3	Handler – Disconnecting	1 (Status)	103
*.eState = 4	Handler – Connecting	1 (Status)	104
*.eState = 5	Handler – Idle	1 (Status)	105
*.eState = 6	Handler – Configuring	1 (Status)	106
*.eState = 7	Handler - Available	1 (Status)	107

Axis Handler – Status	Event Message	Event Type	Event Value
N/A	Handler - ConnectionLoss	1 (Status)	120
N/A	Handler - MethodRegistryFull	1 (Status)	122

Axis – Motion Status	Event Message	Event Type	Event Value
*.StoppingStatus	Axis Stop	1 (Status)	400
*.MoveStatus	Axis Move	1 (Status)	401
*.JogStatus	Axis Jog	1 (Status)	402
*.GearingStatus	Axis Gear	1 (Status)	403
*.HomingStatus	Axis Home	1 (Status)	404
*.PositionCamStatus	Axis PCAM	1 (Status)	405
*.TimeCamStatus	Axis TCAM	1 (Status)	406

Axis – CIP Axis State	Event Message	Event Type	Event Value
*.CIPAxisState = 1	AxisState - Pre-Charge	1 (Status)	501
*.CIPAxisState = 2	AxisState - Stopped	1 (Status)	502
*.CIPAxisState = 3	AxisState - Starting	1 (Status)	503
*.CIPAxisState = 4	AxisState - Running	1 (Status)	504
*.CIPAxisState = 5	AxisState - Testing	1 (Status)	505
*.CIPAxisState = 6	AxisState - Stopping	1 (Status)	506
*.CIPAxisState = 7	AxisState - Aborting	1 (Status)	507
*.CIPAxisState = 8	AxisState - Faulted	1 (Status)	508
*.CIPAxisState = 9	AxisState - Start Inhibited	1 (Status)	509
*.CIPAxisState = 10	AxisState - Shutdown	1 (Status)	510
*.CIPAxisState = 11	AxisState - Axis Inhibited	1 (Status)	511
*.CIPAxisState = 12	AxisState - Not Grouped	1 (Status)	512

Axis – CIP Axis State	Event Message	Event Type	Event Value
*.CIPAxisState = 13	AxisState - No Module	1 (Status)	513

5.2 Methods

When the event is a method execution, the Method ID will be the Event ID. All Method ID's will be displayed with the Handler ID preceding the values listed.

Method ID = [Handler ID][Method Registry ID]

Each method registers itself against its associated handler. The sequence that methods register themselves against the handler will determine the method registry ID. If a method is the third one to register itself against the handler, it will acquire the number 3 as the method registry ID.

Event ID = [MethodID]

Example:
Handler ID = 7
Method is the third one to register itself against handler.
Method Registry Location = 3
Method ID = 7003
Event ID = 7003

Each method is assigned a unique 'type' identifier based on function that is displayed in the Event Value field on method invocation or error. If user-defined methods are developed, they must not use utilize of these Event Values.

Method	Event ID	Event Value
raM_Opr_Clear_xx	Method ID (*.Val_MtdID)	01
raM_Opr_Energize_xx	Method ID (*.Val_MtdID)	02
raM_Opr_DeEnergize_xx	Method ID (*.Val_MtdID)	03
raM_Opr_JogMan	Method ID (*.Val_MtdID)	04
raM_Opr_Stop	Method ID (*.Val_MtdID)	05
raM_Opr_SyncPthPhyAx_xx	Method ID (*.Val_MtdID)	06
raM_Opr_Move333	Method ID (*.Val_MtdID)	07
raM_Opr_Cam	Method ID (*.Val_MtdID)	08
raM_Opr_Phase333	Method ID (*.Val_MtdID)	09
raM_Opr_Home_xx	Method ID (*.Val_MtdID)	10
raM_Opr_HomeToReg_xx	Method ID (*.Val_MtdID)	11
raM_Opr_Gear	Method ID (*.Val_MtdID)	12
raM_Opr_NoProductNoBag	Method ID (*.Val_MtdID)	13
raM_Tec_RotaryKnife	Method ID (*.Val_MtdID)	14
raM_Opr_SoftReg_xx	Method ID (*.Val_MtdID)	15
raM_Opr_DriveReg_xx	Method ID (*.Val_MtdID)	16
raM_Opr_1732IB8Reg_xx	Method ID (*.Val_MtdID)	17
raM_Opr_RegDistance	Method ID (*.Val_MtdID)	18
raM_Opr_FlexGear	Method ID (*.Val_MtdID)	19
raM_Opr_TensionZoneFD	Method ID (*.Val_MtdID)	20
raM_Opr_TensionZoneFD_DN	Method ID (*.Val_MtdID)	21
raM_Opr_TensionZoneFD_LC	Method ID (*.Val_MtdID)	22

Method	Event ID	Event Value
raM_Opr_TensionZoneVD	Method ID (*.Val_MtdID)	23
raM_Opr_TensionZoneVD_DN	Method ID (*.Val_MtdID)	24
raM_Opr_TensionZoneVD_LC	Method ID (*.Val_MtdID)	25

6 Application Code Manager

6.1 Definition Object: raM_Dvc_AxisHandler_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.

6.2 Implement Object: raM_LD_AxisHandler_CD

Implement Language: Ladder Diagram

Parameter Name	Default Value	Instance Name	Definition	Description
ObjectName	raM_LD_AxisHandler_CD	-	Object Name	Instantiation Object Name
PRGName	{ObjectName}		Program Name	Enter the program name where handler AOI is generated
RoutineName	{ObjectName}	{RoutineName}	Routine	Name of the routine where the object will be placed
TagName	{ObjectName}	-	Backing Tag	Instruction Backing Tag
PathAxisName	{AxisName}_PTH		Axis Tag Name	Path Axis Tag Name

Linked Library

Link Name	Catalog Number	Revision	Solution	Category
Device Object		>=3.0	(RA-LIB) Device	Power Motion
raM_Dvc_AxisHandler_CD	raM_Dvc_AxisHandler_CD	>=3.0	(RA-LIB) Machine	DvcHdlr – Handler
raM_LD_AHLP	raM_LD_AHLP	>=2.0	(RA-LIB) Machine	DvcHdlr – Language
raM_LD_DH_SysIni	raM_LD_DH_SysIni	>=1.0	(RA-LIB) Machine	DvcHdlr - Common

Configured HMI Content

HMI Content	Instance Name	Description
Callup Button	{ObjectName}_GrpName	Global Object configured callout instance

Input Interface

Interface Name	Linked Library	Revision
raC_Itf_PowerMotion	PowerMotion Devices	1.0

MethodInterface Members

Member Name	Description
PrgName	Program name where Device Object resides
TagName	Name of Device Object instruction
ModuleName	Name of Module Device Object references
AxisName_PHY	Name of Physical Axis associated with Device Object

Output Interface

Interface Name	Linked Library	Revision
MethodInterface	raM_LD_AxisHandler_CD	1.0

MethodInterface Members

Member Name	Description
AHPrgName	Program name where Axis Handler resides
AxisName_PHY	Name of Physical Axis attached to Handler
AxisName_PTH	Name of Path Axis attached to Handler
AHTagName	Handle Tag name of Axis Handler

6.3 Implement Object: raM_LD_AHLP

This object contains the axis handler language pack program. User can select required language packs while instantiating this object.

Parameter Name	Default Value	Instance Name	Definition	Description
Include_EN	Yes	-		Include English in AH Event languages
Include_CS	No	-		Include Chez in AH Event languages
Include_DE	No	-		Include German in AH Event languages
Include_ES	No	-		Include Spanish in AH Event languages
Include_FR	No	-		Include French in AH Event languages
Include_IT	No	-		Include Italian in AH Event languages
Include_PT	No	-		Include Portugese in AH Event languages
Include_SV	No	-		Include Sweden in AH Event languages
Include_UD	No	-		Include User Defied in AH Event languages

6.4 Implement Object: raM_LD_DH_SysIni

This object contains code to assign Global Handler Device ID. The code is placed in the controller Power-Up Handler.

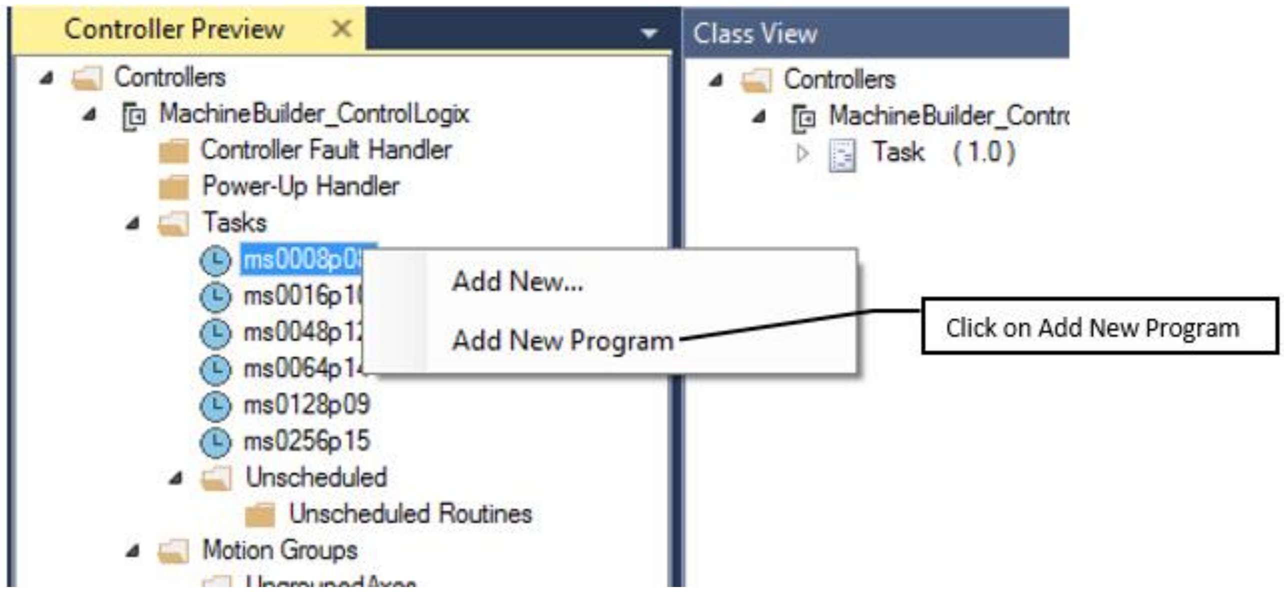
6.5 Attachments

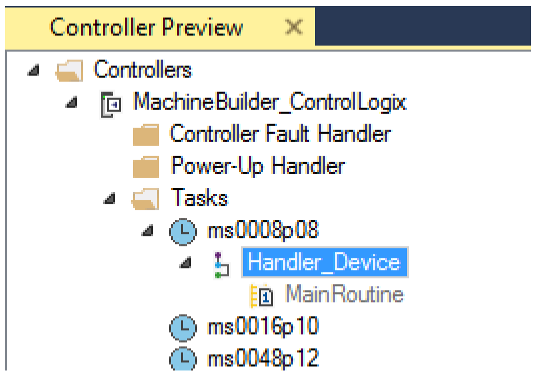
Name	Description	File Name	Extraction path
V3_{LibraryName}	Reference Manual	RM-{LibraryName}.pdf	{ProjectName}\Documentation
V3_{LibraryName}	Faceplate ME	(raM-ME) {LibraryName}-Faceplate.gfx	{ProjectName}\Visualization\FTViewME\Displays
V3_{LibraryName}	Global Object ME	(raM-ME) {LibraryName}-Global.gfx	{ProjectName}\Visualization\FTViewME\GlobalObjects
V3_{LibraryName}	Faceplate SE	(raM-SE) {LibraryName}-Faceplate.gfx	{ProjectName}\Visualization\FTViewME\Displays
V3_{LibraryName}	Global Object SE	(raM-SE) {LibraryName}-Global.gfx	{ProjectName}\Visualization\FTViewME\GlobalObjects
V3_{LibraryName}	View Designer	{LibraryName}.vpd	{ProjectName}\Visualization\ViewDesigner
V2_Images	Images	Images.zip	{ProjectName}\Visualization\Images

7 Application

7.1 Instantiate Axis Handler and Method in Application Code Manager

7.1.1 Add New Program and name as "Handler_Device"





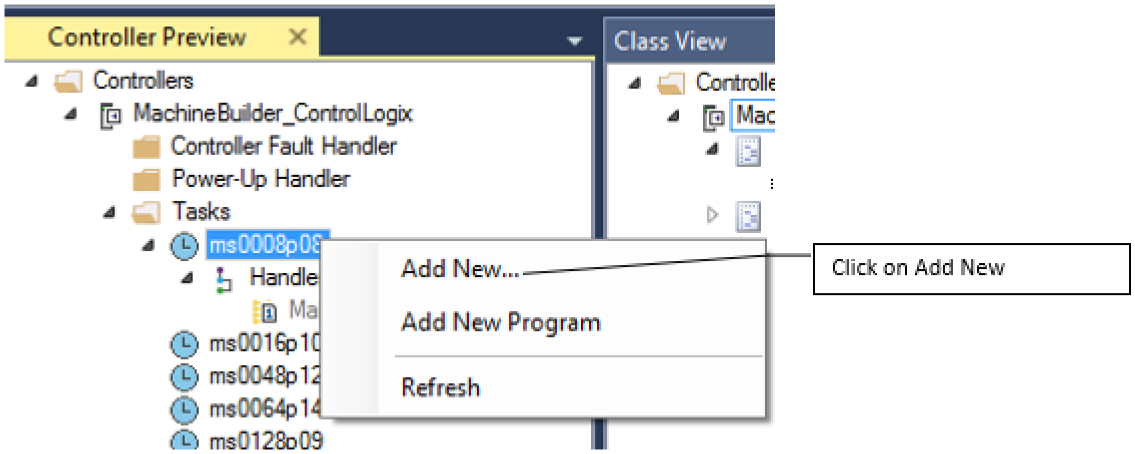
7.1.2 Add Axis Handler

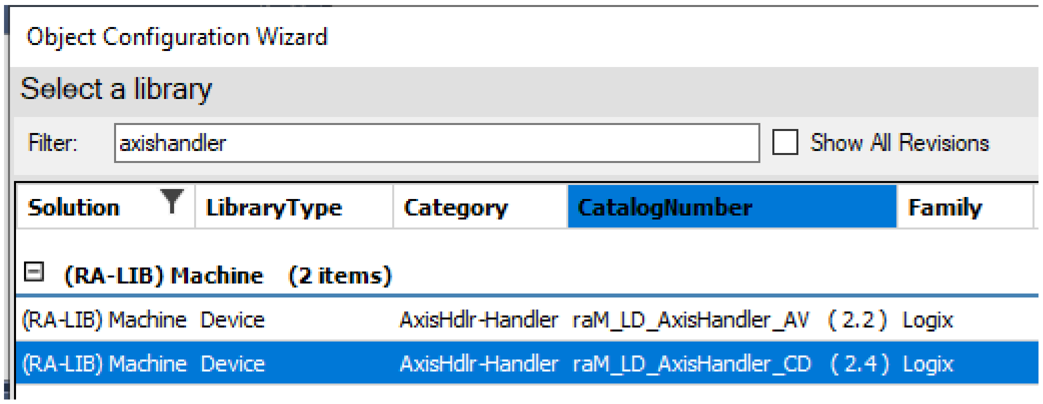
Components required for Axis Handler

- Axis Handler Implement
- Axis Handler Definition (raM_Dvc_AxisHandler_CD) – one time addition
- Device Object Implement: Select appropriate library as per Drive hardware used. This example is using Kinetix5500.
- Device Object Definition (raC_Dvc_K5500) – one time addition
- raM_LD_AHLP: Axis Handler Language pack – one-time addition
- raM_LD_DH_SysIni : Axis System Initialization program – one time addition

As is shown, with the exception of Axis Handler Implement, all other components are just onetime additions. Although the below steps may look like a long process, since most of the onetime addition objects have already been added, subsequent axis handler additions can skip these steps.

Right click ms0008p08 task and select Add New…





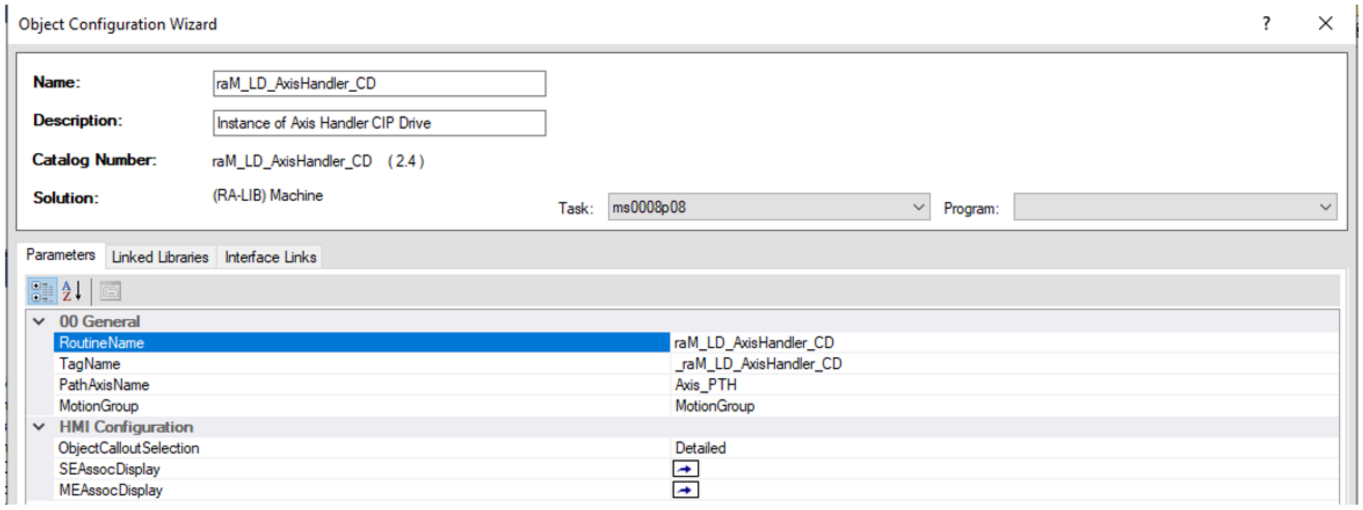
Object Configuration Wizard

Select a library

Filter: axishandler   ☐ Show All Revisions

Solution	LibraryType	Category	CatalogNumber	Family
(RA-LIB) Machine (2 items)				
(RA-LIB) Machine	Device	AxisHdlr-Handler	raM_LD_AxisHandler_AV (2.2)	Logix
(RA-LIB) Machine	Device	AxisHdlr-Handler	raM_LD_AxisHandler_CD (2.4)	Logix

Double click on the raM_LD_AxisHandler_CD (this is the implement object of raM_Dvc_AxisHandler_CD AOI).



Object Configuration Wizard

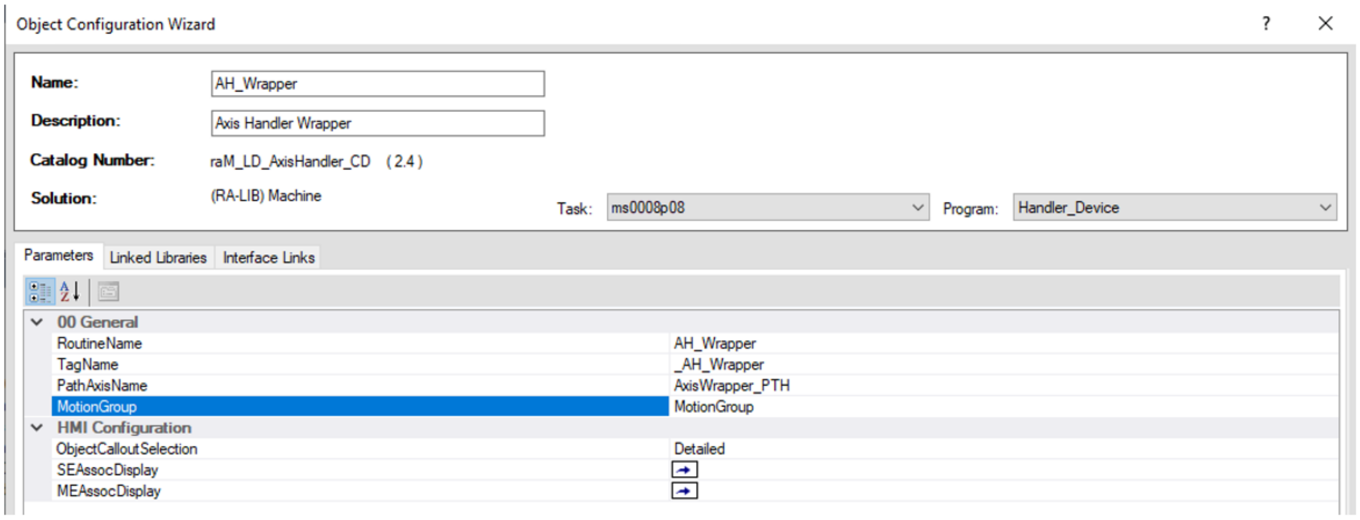
Name:	raM_LD_AxisHandler_CD
Description:	Instance of Axis Handler CIP Drive
Catalog Number:	raM_LD_AxisHandler_CD (2.4)
Solution:	(RA-LIB) Machine
Task:	ms0008p08
Program:	

Parameters | Linked Libraries | Interface Links

00 General	
RoutineName	raM_LD_AxisHandler_CD
TagName	_raM_LD_AxisHandler_CD
PathAxisName	Axis_PTH
MotionGroup	MotionGroup
HMI Configuration	
ObjectCalloutSelection	Detailed
SEAssocDisplay	
MEAssocDisplay	

Enter the following:

- Object Name: AH_Wrapper
- Description: Axis Handler Wrapper
- Program from Drop down list: Handler_Device
- PathAxisName: AxisWrapper_PTH
- MotionGroup: Motion Group name is derived from Controller Properties.



Object Configuration Wizard

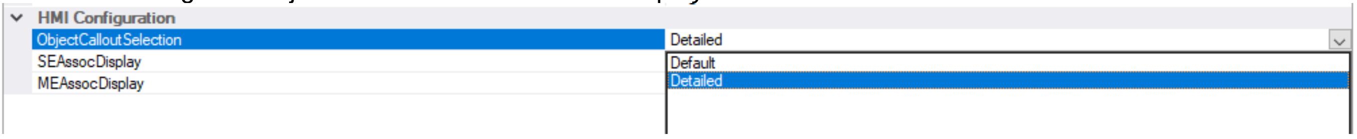
Name:	AH_Wrapper
Description:	Axis Handler Wrapper
Catalog Number:	raM_LD_AxisHandler_CD (2.4)
Solution:	(RA-LIB) Machine

Task: ms0008p08 Program: Handler_Device

Parameters | Linked Libraries | Interface Links

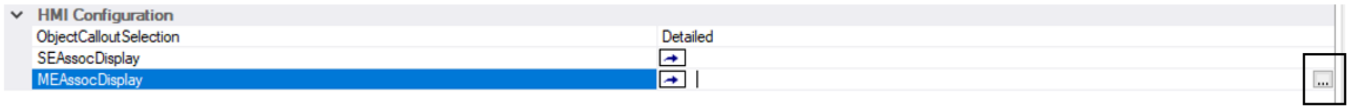
00 General	
RoutineName	AH_Wrapper
TagName	_AH_Wrapper
PathAxisName	AxisWrapper_PTH
MotionGroup	MotionGroup
HMI Configuration	
ObjectCalloutSelection	Detailed
SEAssocDisplay	
MEAssocDisplay	

Select the Navigation object to be instantiated in the display.

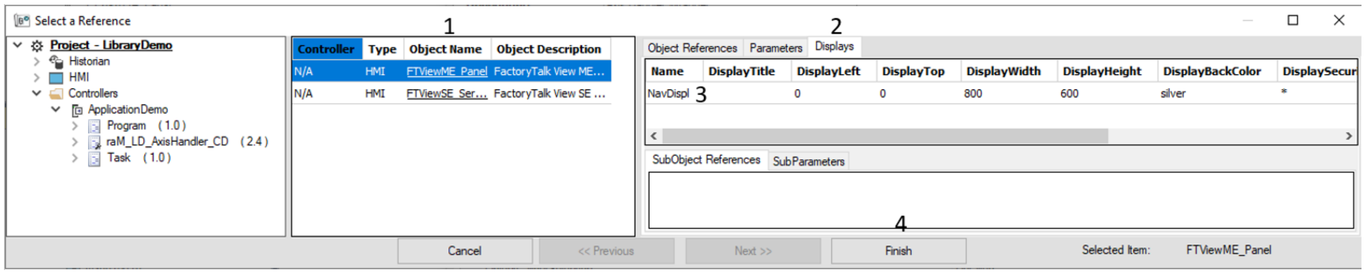


HMI Configuration	
ObjectCalloutSelection	Detailed
SEAssocDisplay	Default
MEAssocDisplay	Detailed

Select the Display where the navigation object is to be instantiated.



HMI Configuration	
ObjectCalloutSelection	Detailed
SEAssocDisplay	
MEAssocDisplay	



Select a Reference

- Project - LibraryDemo
  - Historian
  - HMI
  - Controllers
    - ApplicationDemo
      - Program (1.0)
      - raM_LD_AxisHandler_CD (2.4)
      - Task (1.0)

1

Controller	Type	Object Name	Object Description
N/A	HMI	FTViewME_Panel	FactoryTalk View ME…
N/A	HMI	FTViewSE_Ser…	FactoryTalk View SE …

2

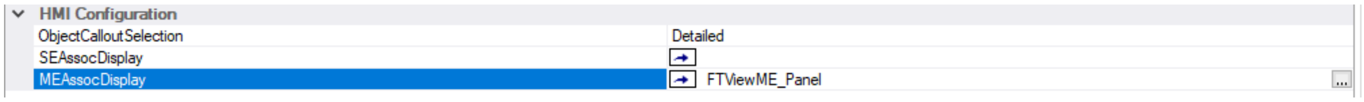
Object References | Parameters | Displays

Name	DisplayTitle	DisplayLeft	DisplayTop	DisplayWidth	DisplayHeight	DisplayBackColor	DisplaySecur
NavDispl 3		0	0	800	600	silver	*

SubObject References | SubParameters

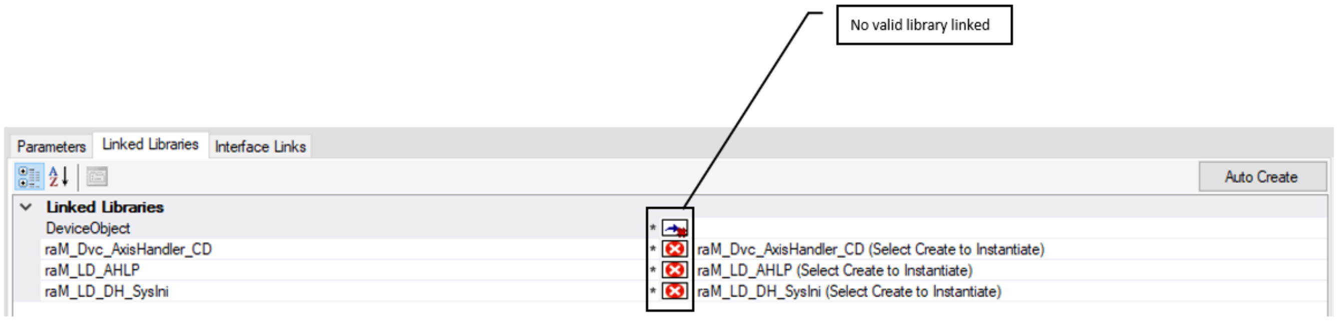
4

Selected Item: FTViewME_Panel



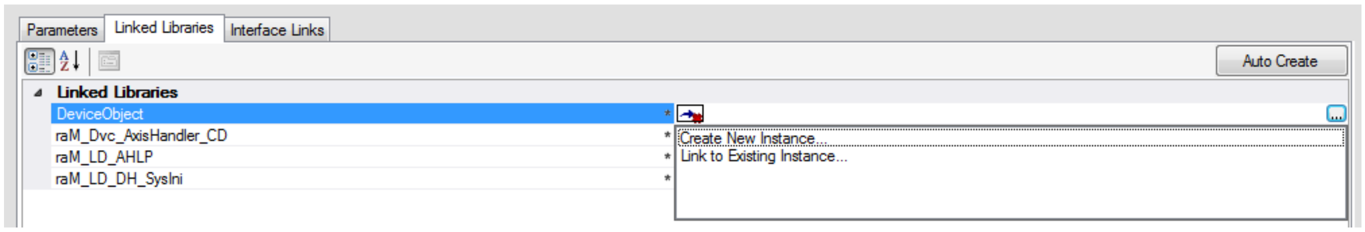
HMI Configuration	
ObjectCalloutSelection	Detailed
SEAssocDisplay	
MEAssocDisplay	FTViewME_Panel

Select the Linked Libraries tab

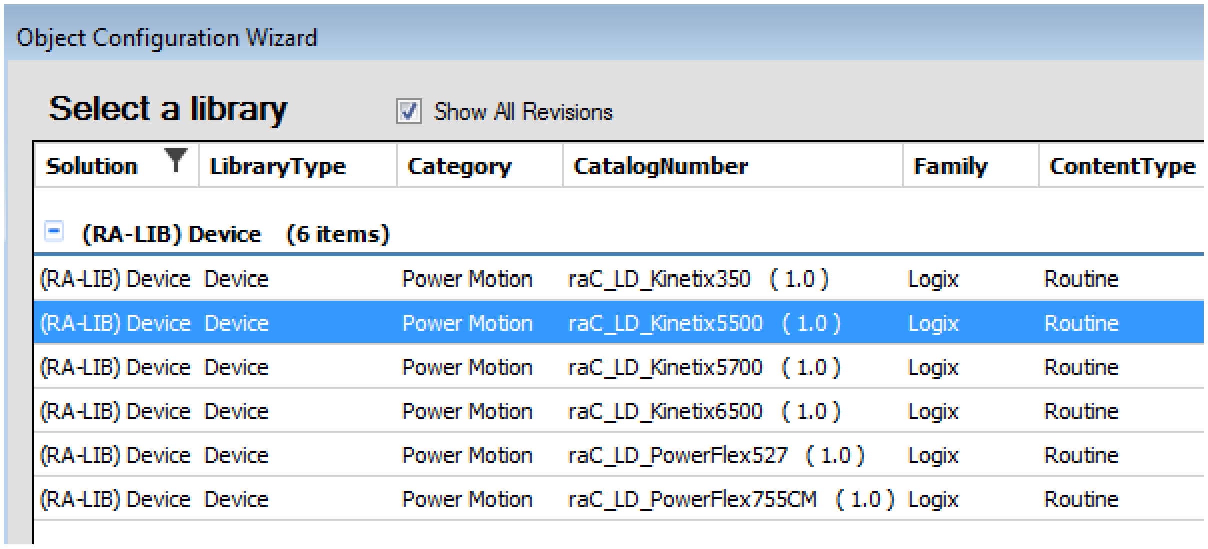


Linked Libraries	
DeviceObject	
raM_Dvc_AxisHandler_CD	raM_Dvc_AxisHandler_CD (Select Create to Instantiate)
raM_LD_AHLP	raM_LD_AHLP (Select Create to Instantiate)
raM_LD_DH_SysIni	raM_LD_DH_SysIni (Select Create to Instantiate)

The first Linked Library that is needed for Axis Handler is the Device Object. In this case, a Kinetix5500 drive with associated Axis is required, so add an raC_Dvc_K5500 device object. Click in DeviceObject box and select Create New Instance.

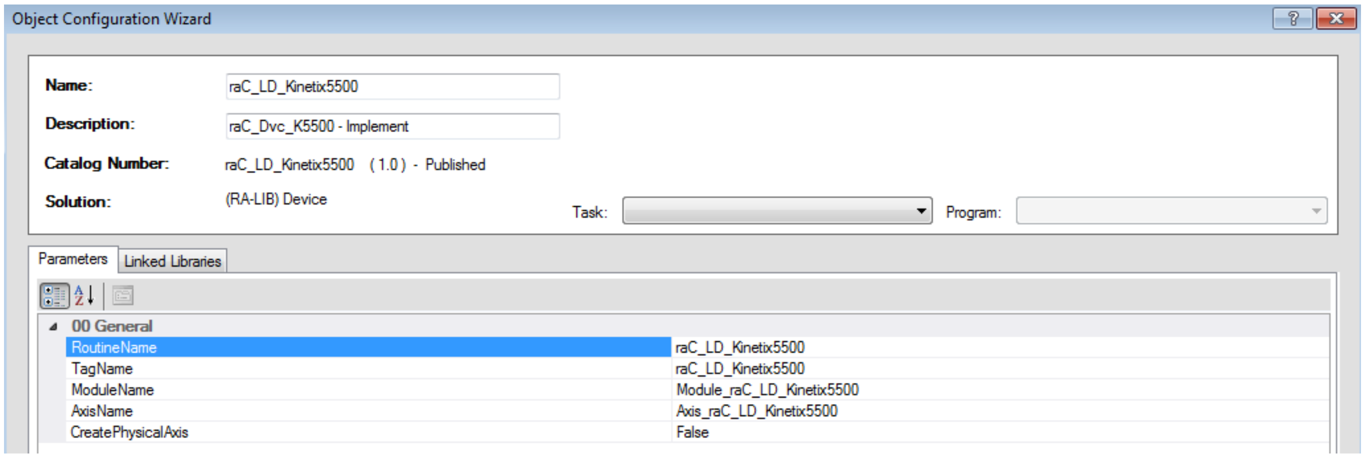


Select the raC_Dvc_K5500 Device object from the available list. The filter has been set so only relevant objects will be available to be selected. This prevents the user from selecting incompatible objects. As shown below, only CIP device objects are shown.



Solution	LibraryType	Category	CatalogNumber	Family	ContentType
(RA-LIB) Device (6 items)					
(RA-LIB) Device	Device	Power Motion	raC_LD_Kinetix350 (1.0)	Logix	Routine
(RA-LIB) Device	Device	Power Motion	raC_LD_Kinetix5500 (1.0)	Logix	Routine
(RA-LIB) Device	Device	Power Motion	raC_LD_Kinetix5700 (1.0)	Logix	Routine
(RA-LIB) Device	Device	Power Motion	raC_LD_Kinetix6500 (1.0)	Logix	Routine
(RA-LIB) Device	Device	Power Motion	raC_LD_PowerFlex527 (1.0)	Logix	Routine
(RA-LIB) Device	Device	Power Motion	raC_LD_PowerFlex755CM (1.0)	Logix	Routine

This opens Object Configuration Wizard of raC_Dvc_K5500


Object Configuration Wizard

Name: raC_LD_Kinetix5500
Description: raC_Dvc_K5500 - Implement
Catalog Number: raC_LD_Kinetix5500 (1.0) - Published
Solution: (RA-LIB) Device
Task:
Program:

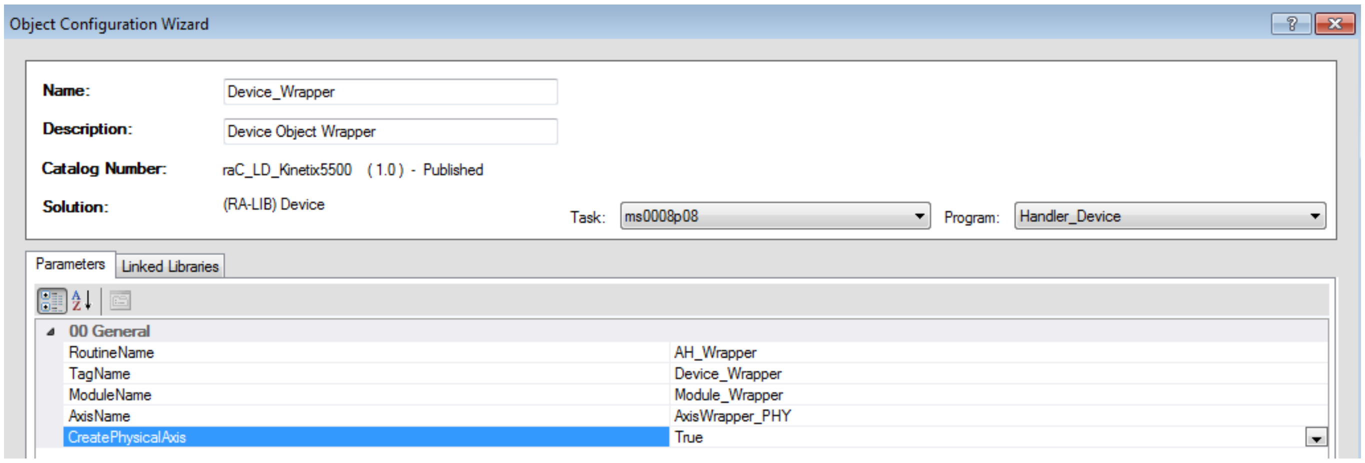
Parameters | Linked Libraries

00 General

RoutineName	raC_LD_Kinetix5500
TagName	raC_LD_Kinetix5500
ModuleName	Module_raC_LD_Kinetix5500
AxisName	Axis_raC_LD_Kinetix5500
CreatePhysicalAxis	False

Enter the following

- Object Name: Device_Wrapper
- Description: Device Object Wrapper
- Task: ms0008p08
- Program: Handler_Device
- RoutineName: AH_Wrapper (so Handler and Device are in same routine)
- Module: Module_Wrapper
- AxisName: AxisWrapper_PHY (this is Physical Axis, what was added in axis handler is Path Axis)
- CreatePhysicalAxis: True


Object Configuration Wizard

Name: Device_Wrapper
Description: Device Object Wrapper
Catalog Number: raC_LD_Kinetix5500 (1.0) - Published
Solution: (RA-LIB) Device
Task: ms0008p08
Program: Handler_Device

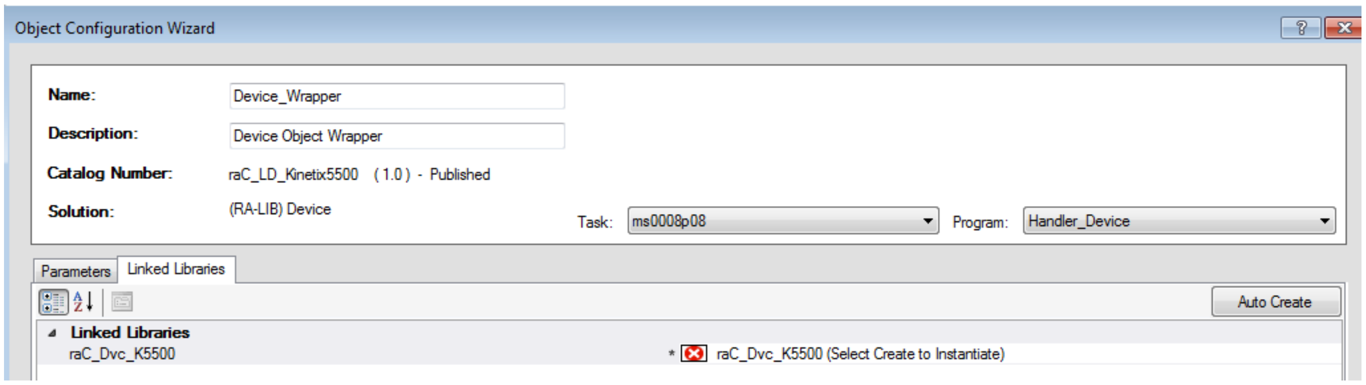
Parameters | Linked Libraries

00 General

RoutineName	AH_Wrapper
TagName	Device_Wrapper
ModuleName	Module_Wrapper
AxisName	AxisWrapper_PHY
CreatePhysicalAxis	True

Click Next to add the definition object raC_Dvc_K5500.

Note: Definition object is required to be added only once. If another raC_LD_K5500 object is added, then the linked library is automatically filled in as the definition is already available. This simplifies the instantiation steps.



Object Configuration Wizard

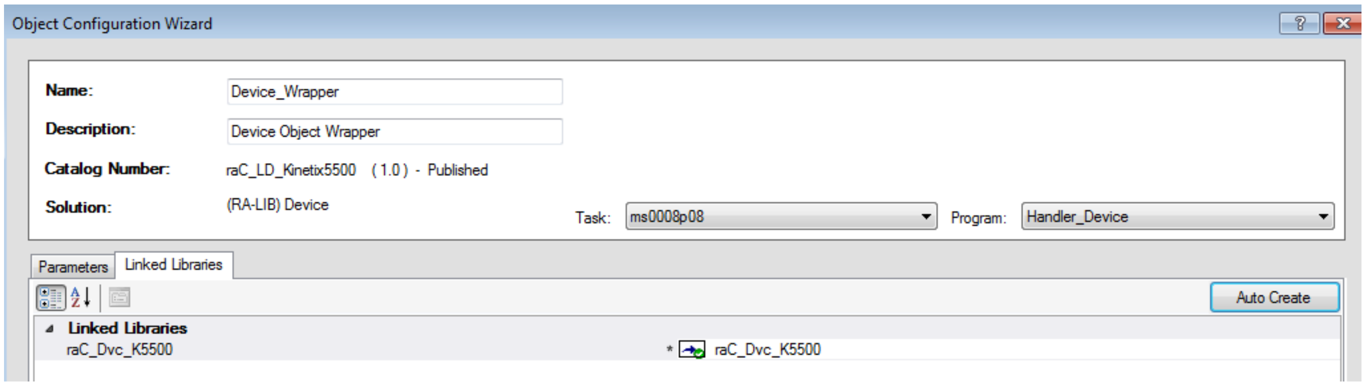
Name:	Device_Wrapper
Description:	Device Object Wrapper
Catalog Number:	raC_LD_Kinetix5500 (1.0) - Published
Solution:	(RA-LIB) Device

Task: ms0008p08 Program: Handler_Device

Linked Libraries

raC_Dvc_K5500	* raC_Dvc_K5500 (Select Create to Instantiate)

Click Auto Create to auto add the definition.



Object Configuration Wizard

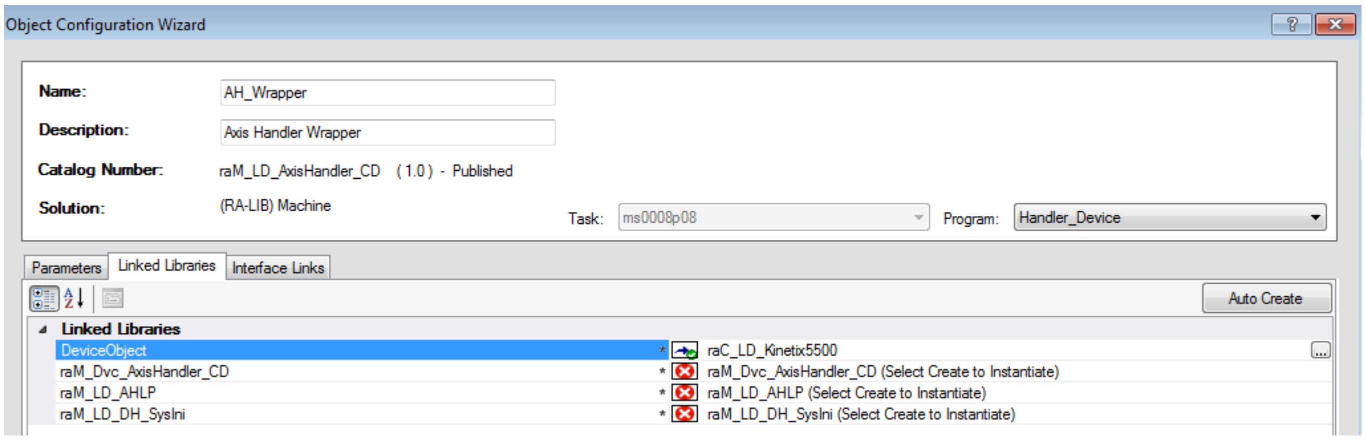
Name:	Device_Wrapper
Description:	Device Object Wrapper
Catalog Number:	raC_LD_Kinetix5500 (1.0) - Published
Solution:	(RA-LIB) Device

Task: ms0008p08 Program: Handler_Device

Linked Libraries

raC_Dvc_K5500	* raC_Dvc_K5500

Click Finish to close the Object Configuration Wizard of raC_LD_K5500 object. The active window has now returned to the Axis Hander Object Configuration Wizard window.



Object Configuration Wizard

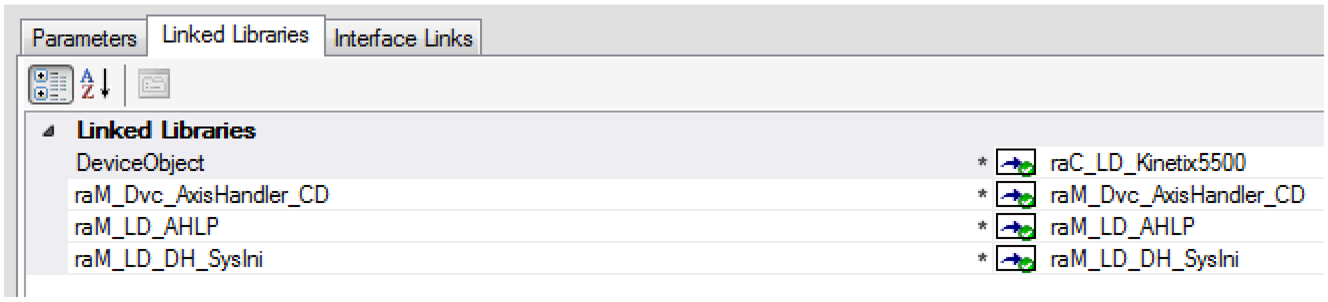
Name:	AH_Wrapper
Description:	Axis Handler Wrapper
Catalog Number:	raM_LD_AxisHandler_CD (1.0) - Published
Solution:	(RA-LIB) Machine

Task: ms0008p08 Program: Handler_Device

Linked Libraries

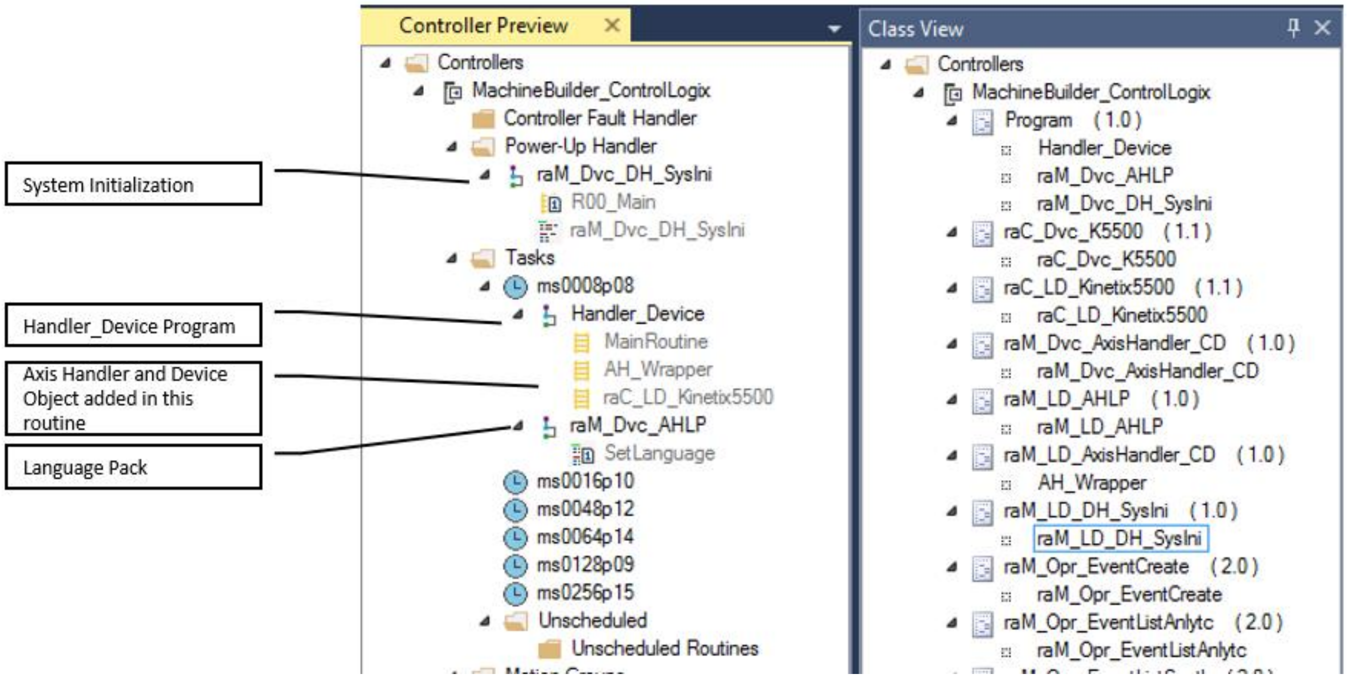
DeviceObject	* raC_LD_Kinetix5500
raM_Dvc_AxisHandler_CD	* raM_Dvc_AxisHandler_CD (Select Create to Instantiate)
raM_LD_AHLP	* raM_LD_AHLP (Select Create to Instantiate)
raM_LD_DH_SysIni	* raM_LD_DH_SysIni (Select Create to Instantiate)

The rest of the objects can be instantiated by clicking Auto Create.

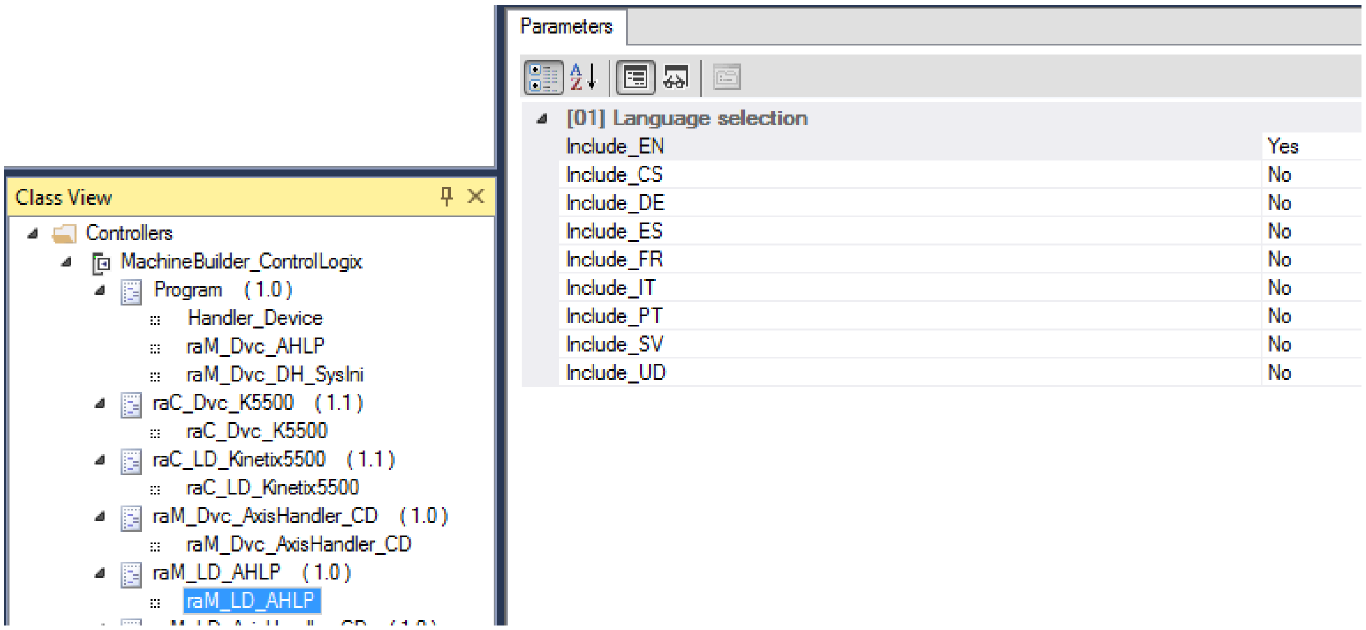


Linked Libraries		
DeviceObject	*	raC_LD_Kinetix5500
raM_Dvc_AxisHandler_CD	*	raM_Dvc_AxisHandler_CD
raM_LD_AHLP	*	raM_LD_AHLP
raM_LD_DH_SysIni	*	raM_LD_DH_SysIni

There is no need to do anything in the Interface Links tab. Click Finish will complete instantiation of Axis Handler Object



Since the Language pack was added by Auto Create in Axis Handler Linked Library, select the raM_LD_AHLP object to configure required languages.


Class View

- Controllers
  - MachineBuilder_ControlLogix
    - Program (1.0)
      - Handler_Device
      - raM_Dvc_AHLP
      - raM_Dvc_DH_SysIni
    - raC_Dvc_K5500 (1.1)
      - raC_Dvc_K5500
    - raC_LD_Kinetix5500 (1.1)
      - raC_LD_Kinetix5500
    - raM_Dvc_AxisHandler_CD (1.0)
      - raM_Dvc_AxisHandler_CD
    - raM_LD_AHLP (1.0)
      - raM_LD_AHLP

Parameters

[01] Language selection

Include_EN	Yes
Include_CS	No
Include_DE	No
Include_ES	No
Include_FR	No
Include_IT	No
Include_PT	No
Include_SV	No
Include_UD	No

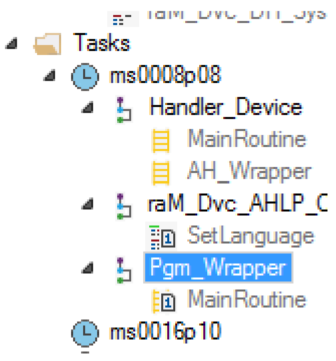
7.1.3 Adding a Method

A few of the most common methods are:

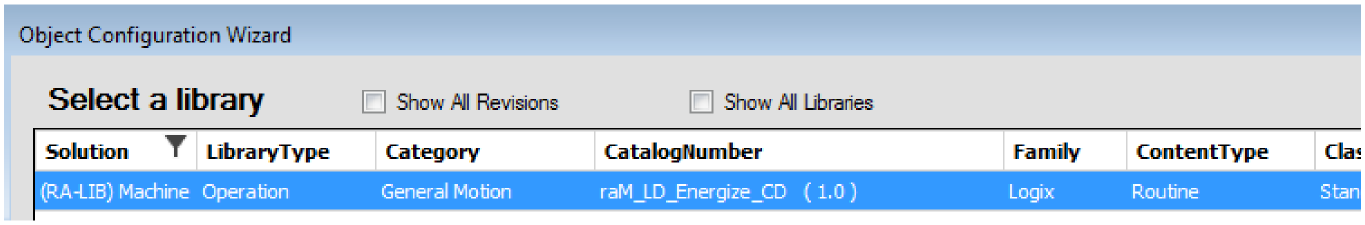
- raM_Opr_Energize_CD
- raM_Opr_DeEnergize_CD
- raM_Opr_SyncPthPhyAx_CD
- raM_Opr_Move333_CD

The following will demonstrate the addition of an raM_Opr_Energize_CD.

To add application code, create another program. Name the program as Pgm_Wrapper


- Tasks
  - ms0008p08
    - Handler_Device
      - MainRoutine
      - AH_Wrapper
    - raM_Dvc_AHLP_C
      - SetLanguage
    - Pgm_Wrapper
      - MainRoutine
  - ms0016p10

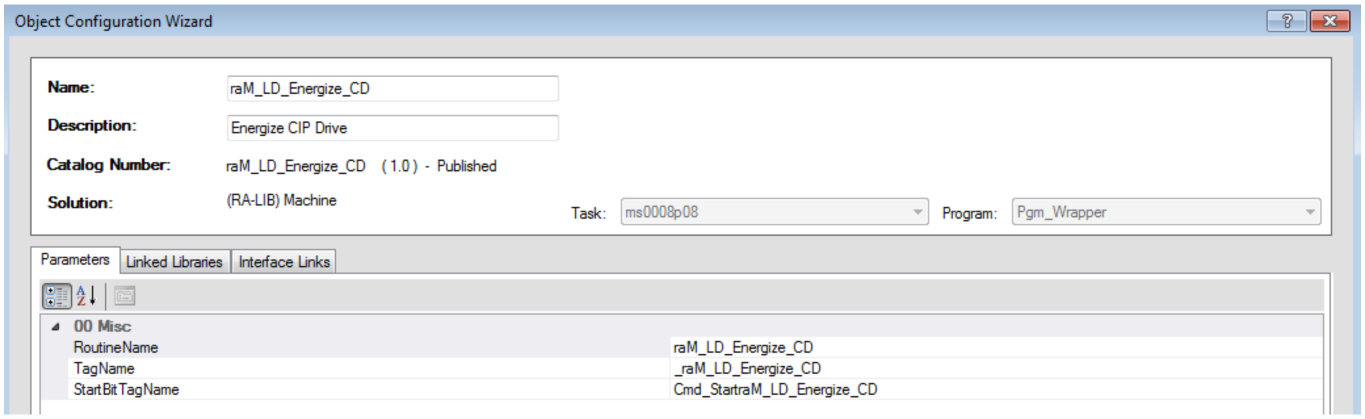
Right Click on Pgm_Wrapper and click Add New. Select raM_LD_Energize_CD (implement object of raM_Opr_Energize_CD)


Object Configuration Wizard

Select a library ☐ Show All Revisions ☐ Show All Libraries

Solution	LibraryType	Category	CatalogNumber	Family	ContentType	Clas
(RA-LIB) Machine	Operation	General Motion	raM_LD_Energize_CD (1.0)	Logix	Routine	Stan

.


Object Configuration Wizard

Name: raM_LD_Energize_CD
Description: Energize CIP Drive
Catalog Number: raM_LD_Energize_CD (1.0) - Published
Solution: (RA-LIB) Machine
Task: ms0008p08 Program: Pgm_Wrapper

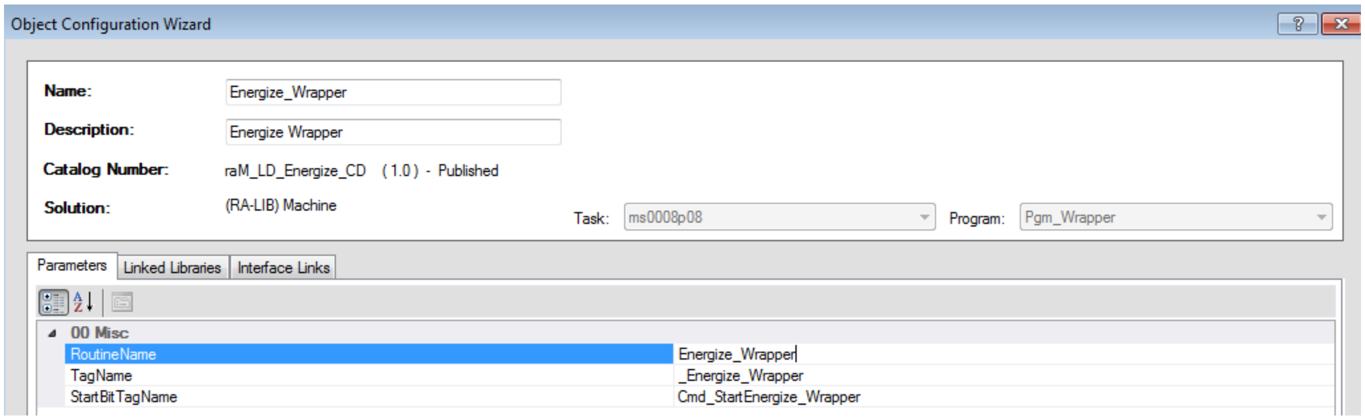
Parameters | Linked Libraries | Interface Links

00 Misc

RoutineName	raM_LD_Energize_CD
TagName	_raM_LD_Energize_CD
StartBitTagName	Cmd_StartraM_LD_Energize_CD

Enter the following

- Object Name: Energize_Wrapper
- Description: Energize_Wrapper


Object Configuration Wizard

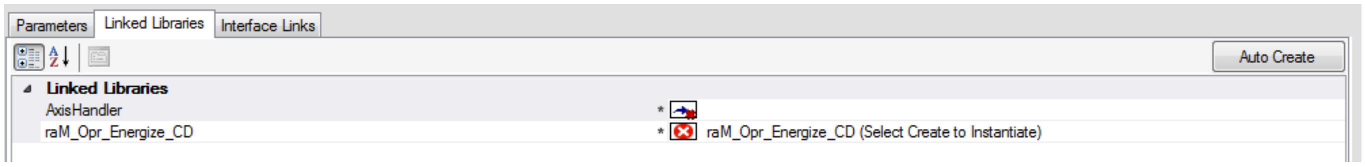
Name: Energize_Wrapper
Description: Energize Wrapper
Catalog Number: raM_LD_Energize_CD (1.0) - Published
Solution: (RA-LIB) Machine
Task: ms0008p08 Program: Pgm_Wrapper

Parameters | Linked Libraries | Interface Links

00 Misc

RoutineName	Energize_Wrapper
TagName	_Energize_Wrapper
StartBitTagName	Cmd_StartEnergize_Wrapper

Select the Linked Libraries tab

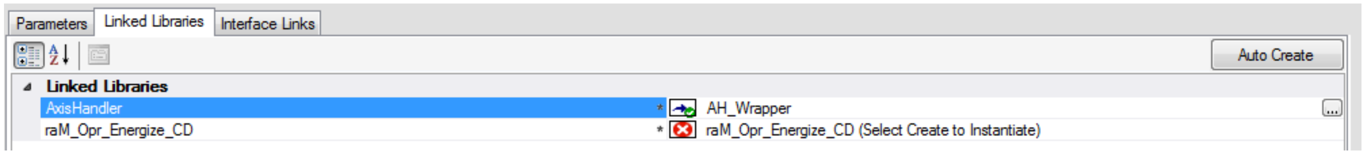

Parameters | Linked Libraries | Interface Links

Auto Create

Linked Libraries

AxisHandler	*
raM_Opr_Energize_CD	* raM_Opr_Energize_CD (Select Create to Instantiate)

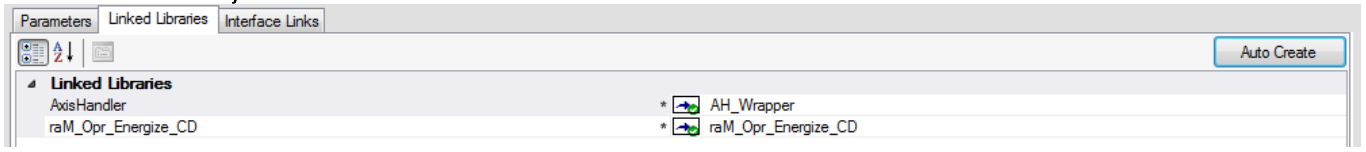
Click raM_LD_AxisHandler_CD and link to existing Axis Handler that was created earlier (AH_Wrapper)



Linked Libraries

AxisHandler	* AH_Wrapper
raM_Opr_Energize_CD	* raM_Opr_Energize_CD (Select Create to Instantiate)

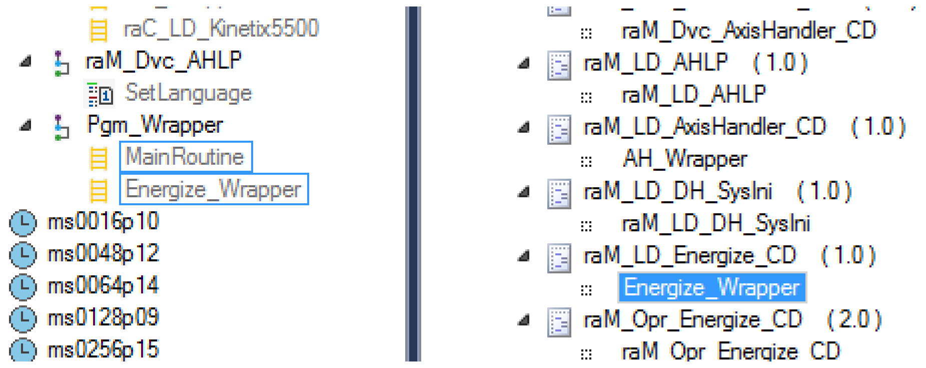
To add an asset object click Auto Create.



Linked Libraries

AxisHandler	* AH_Wrapper
raM_Opr_Energize_CD	* raM_Opr_Energize_CD

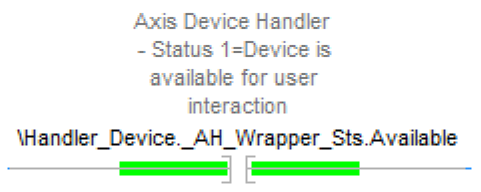
There is no need to make any changes in the Interface Links tab. Click Finish to complete the instantiation of raM_LD_Energize_CD Object.



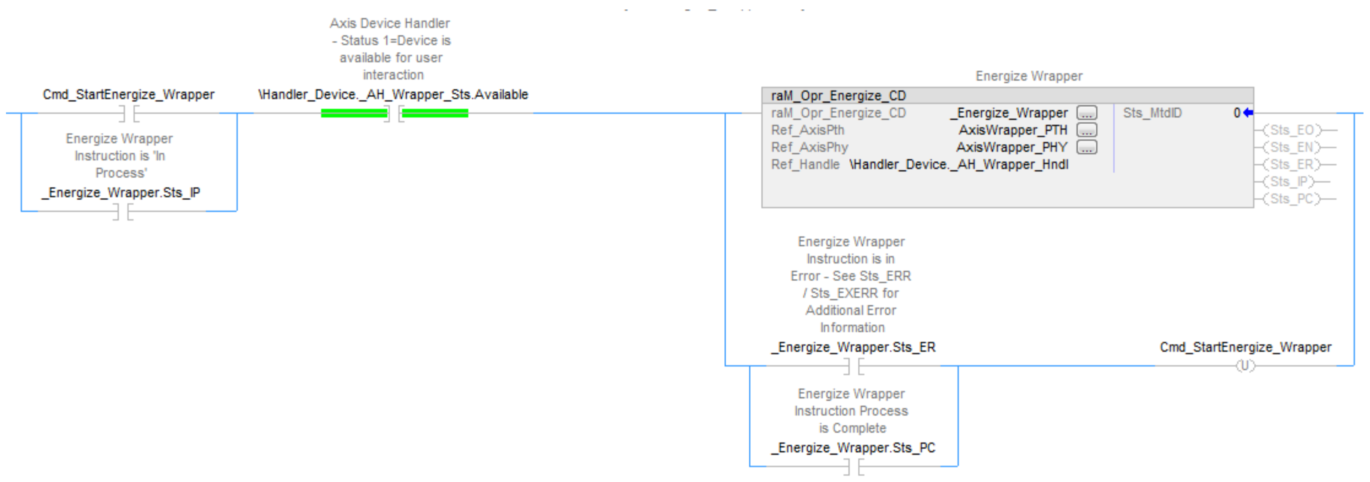
Generate the code and open the resultant file in Studio5000.

7.2 <u>Interfacing from Application Code</u>

When interfacing with the handler from the user application code, handler commands and status can be accessed using Direct Access parameters. The example shows how the Axis Handler Available status is used as an interlock to the Energize Method.


Axis Device Handler - Status 1=Device is available for user interaction
\Handler_Device._AH_Wrapper_Sts.Available

Example: Method linked to the data handle


Cmd_StartEnergize_Wrapper
Energize Wrapper Instruction is 'In Process'
_Energize_Wrapper.Sts_IP
Axis Device Handler - Status 1=Device is available for user interaction
\Handler_Device._AH_Wrapper_Sts.Available
Energize Wrapper
raM_Opr_Energize_CD
raM_Opr_Energize_CD _Energize_Wrapper
Ref_AxisPth AxisWrapper_PTH
Ref_AxisPhy AxisWrapper_PHY
Ref_Handle \Handler_Device._AH_Wrapper_Hndl
Sts_MtdID 0
Sts_EO
Sts_EN
Sts_ER
Sts_IP
Sts_PC
Energize Wrapper Instruction is in Error - See Sts_ERR / Sts_EXERR for Additional Error Information
_Energize_Wrapper.Sts_ER
Cmd_StartEnergize_Wrapper
U
Energize Wrapper Instruction Process is Complete
_Energize_Wrapper.Sts_PC

7.3 <u>Method Error Configuration</u>

This section illustrates the behavior of the Handler when the configuration for the method error is changed from Warning Event to Fault Event.

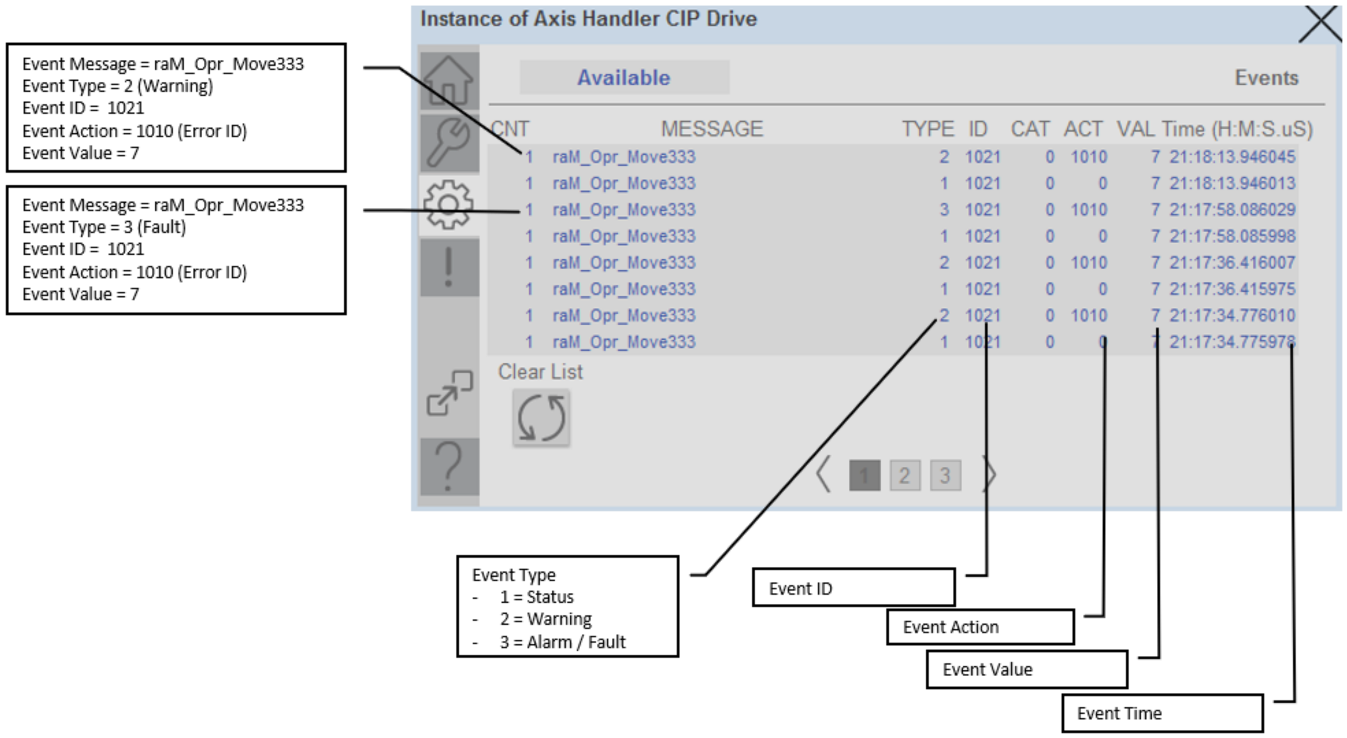
The method error configuration can be changed programmatically at any time. The user can set the standard desired behavior (warning or fault) for method error and individually treat a specific method's error that have a desired behavior different from the standard.

*.Cfg	Function / Description
AxisHandler_Cfg.MethodError	Method Error Interpretation Enumerated 0 = Warning Event 1 = Fault Event

In this example, consider the method raM_Opr_Move333 (method value assignment = 7).
An invalid parameter was entered in the method configuration to cause the Error 1010 when instruction is executed.

Error 1010 - Cfg_MoveType is not valid

Event Message (Method)	Event Type	Event ID	Event Action	Event Value
raM_Opr_Move333	3 (Fault)	1021	1010 (Error ID)	7
raM_Opr_Move333	1 (Status)	1021	0	7
raM_Opr_Move333	2 (Warning)	1021	1010 (Error ID)	7



Event Message = raM_Opr_Move333
Event Type = 2 (Warning)
Event ID = 1021
Event Action = 1010 (Error ID)
Event Value = 7

Event Message = raM_Opr_Move333
Event Type = 3 (Fault)
Event ID = 1021
Event Action = 1010 (Error ID)
Event Value = 7

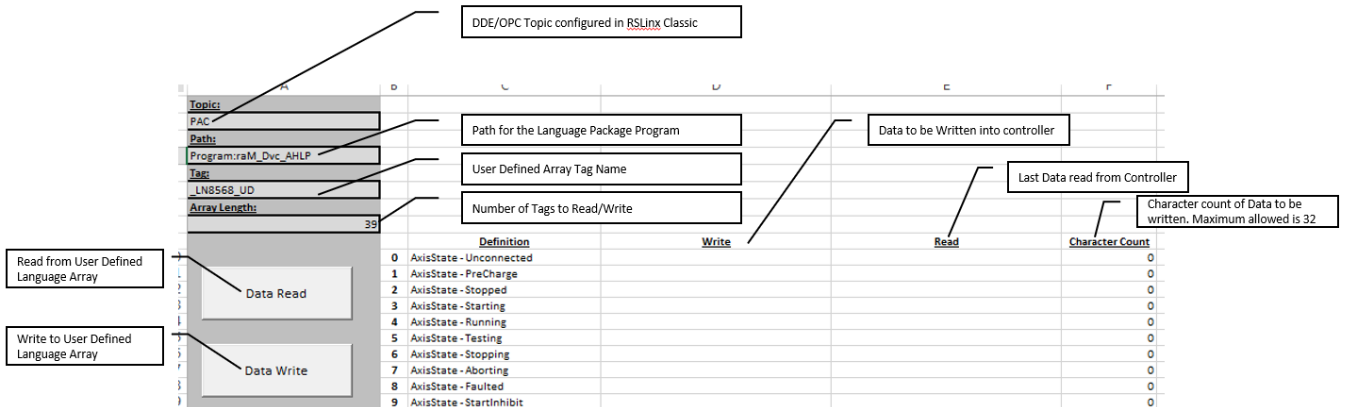
Event Type
- 1 = Status
- 2 = Warning
- 3 = Alarm / Fault

CNT	MESSAGE	TYPE	ID	CAT	ACT	VAL	Time (H:M:S.uS)
1	raM_Opr_Move333	2	1021	0	1010	7	21:18:13.946045
1	raM_Opr_Move333	1	1021	0	0	7	21:18:13.946013
1	raM_Opr_Move333	3	1021	0	1010	7	21:17:58.086029
1	raM_Opr_Move333	1	1021	0	0	7	21:17:58.085998
1	raM_Opr_Move333	2	1021	0	1010	7	21:17:36.416007
1	raM_Opr_Move333	1	1021	0	0	7	21:17:36.415975
1	raM_Opr_Move333	2	1021	0	1010	7	21:17:34.776010
1	raM_Opr_Move333	1	1021	0	0	7	21:17:34.775978

7.4 <u>User Defined Language</u>

If "User Defined Language" is included in the Axis Handler Language Pack, use the Excel file "UserDefinedLanguage_xADHv2.xlsx" included with the Machine Builder Library download package to write the user-defined messages to the Language Package Program. As text data in the PAC is stored in an ASCII string, text entered must be representable by an ASCII character code.

.

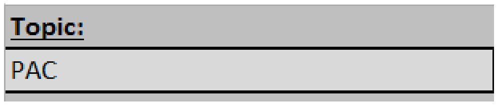

DDE/OPC Topic configured in RSLinx Classic
Path for the Language Package Program
User Defined Array Tag Name
Number of Tags to Read/Write
Data to be Written into controller
Last Data read from Controller
Character count of Data to be written. Maximum allowed is 32
Read from User Defined Language Array
Write to User Defined Language Array
Data Read
Data Write

Topic: PAC
Path: Program:raM_Dvc_AHLP
Tag: _LN8568_UD
Array Length: 39

	Definition	Write	Read	Character Count
0	AxisState - Unconnected			0
1	AxisState - PreCharge			0
2	AxisState - Stopped			0
3	AxisState - Starting			0
4	AxisState - Running			0
5	AxisState - Testing			0
6	AxisState - Stopping			0
7	AxisState - Aborting			0
8	AxisState - Faulted			0
9	AxisState - StartInhibit			0

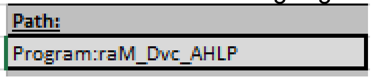
There are 2 workbooks (one for each AxisHandler type).
- UD – CIPDrive
- UD – Axis Virtual

1. Select the app-appropriate worksheet based on the Axis Handler type.
2. Set the DDE/OPC Topic Name as configured in RSLinx. In this example, the topic name is "PAC".



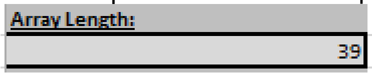
Topic:
PAC

3. Set the Path to the Language Package Program. In this example, the Path is “Program:raM_Dvc_AHLP”.



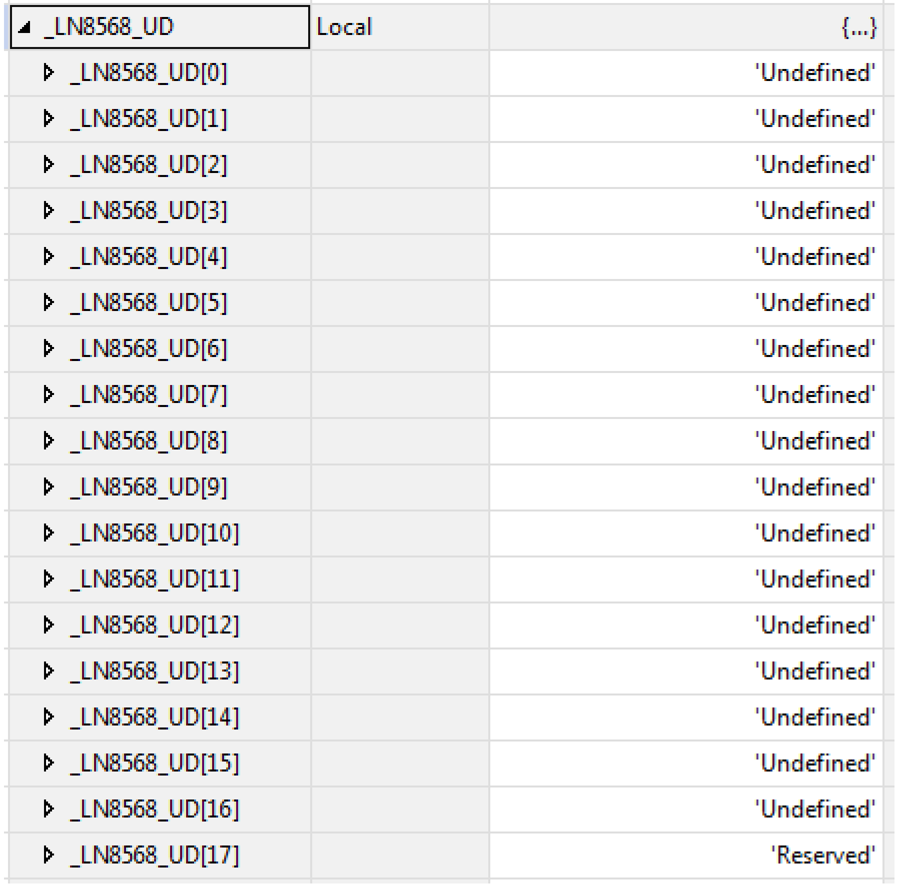
Path:
Program:raM_Dvc_AHLP

4. Confirm the number of tags to be written. The write length is static based on the number of strings that must be updated. In this example, the Length is “39”.



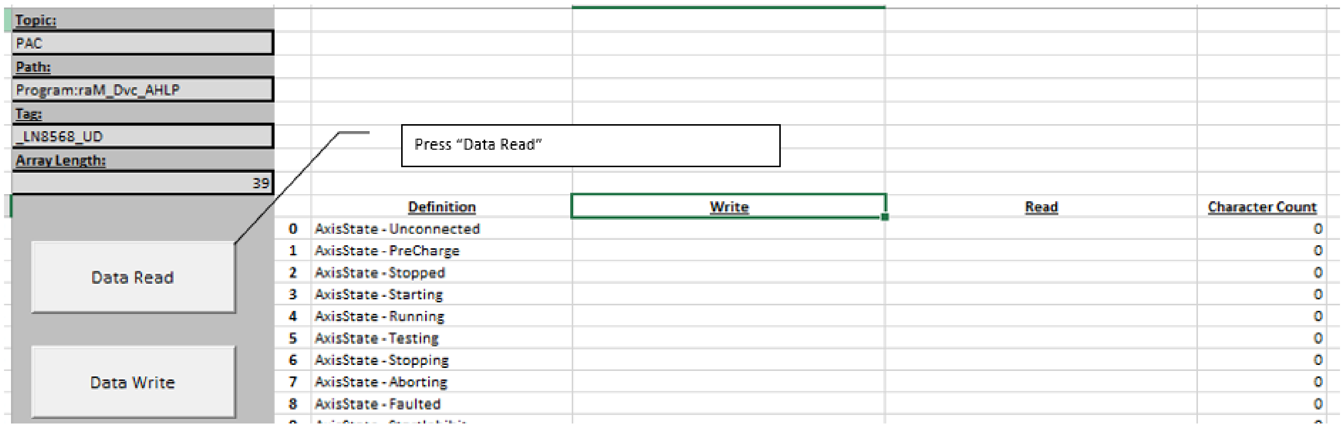
Array Length:
39

5. Look at the existing Data in the User Defined Language Tags in the controller



◢ _LN8568_UD	Local	{...}
▷ _LN8568_UD[0]		'Undefined'
▷ _LN8568_UD[1]		'Undefined'
▷ _LN8568_UD[2]		'Undefined'
▷ _LN8568_UD[3]		'Undefined'
▷ _LN8568_UD[4]		'Undefined'
▷ _LN8568_UD[5]		'Undefined'
▷ _LN8568_UD[6]		'Undefined'
▷ _LN8568_UD[7]		'Undefined'
▷ _LN8568_UD[8]		'Undefined'
▷ _LN8568_UD[9]		'Undefined'
▷ _LN8568_UD[10]		'Undefined'
▷ _LN8568_UD[11]		'Undefined'
▷ _LN8568_UD[12]		'Undefined'
▷ _LN8568_UD[13]		'Undefined'
▷ _LN8568_UD[14]		'Undefined'
▷ _LN8568_UD[15]		'Undefined'
▷ _LN8568_UD[16]		'Undefined'
▷ _LN8568_UD[17]		'Reserved'

6. The following displays the workbook before pressing “Data Read”

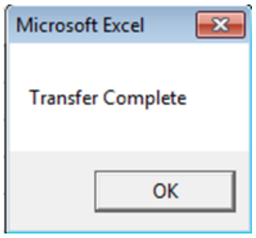


Topic:
PAC
Path:
Program:raM_Dvc_AHLP
Tag:
_LN8568_UD
Array Length:
39

Press “Data Read”

	Definition	Write	Read	Character Count
0	AxisState - Unconnected			0
1	AxisState - PreCharge			0
2	AxisState - Stopped			0
3	AxisState - Starting			0
4	AxisState - Running			0
5	AxisState - Testing			0
6	AxisState - Stopping			0
7	AxisState - Aborting			0
8	AxisState - Faulted			0

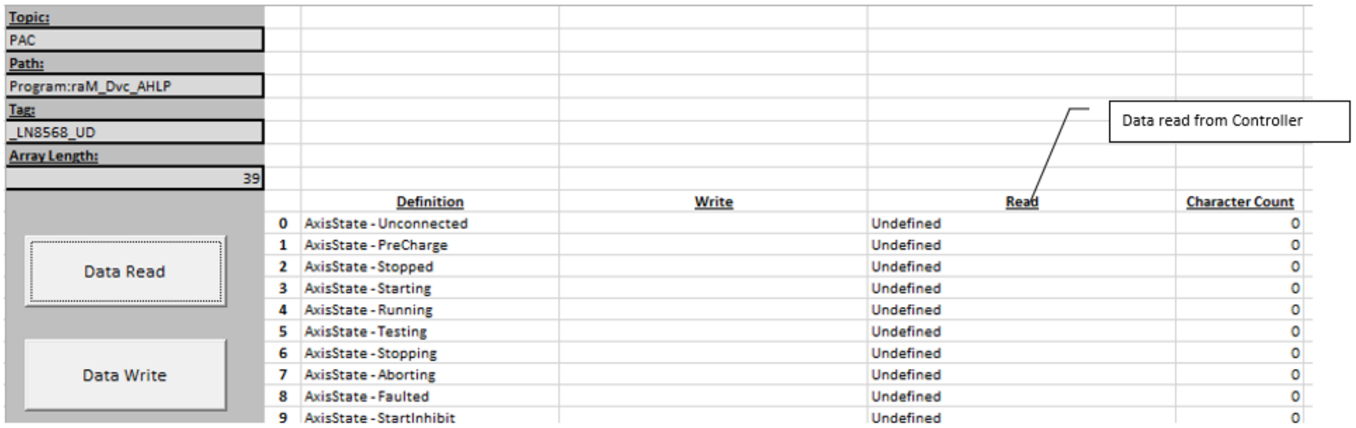
7. Press “Data Read” Push-button in the workbook to read information from the User Defined Language Tags in the controller. A message will inform that transfer has been completed.



Microsoft Excel

Transfer Complete

8. The following displays the workbook after pressing “Data Read”.

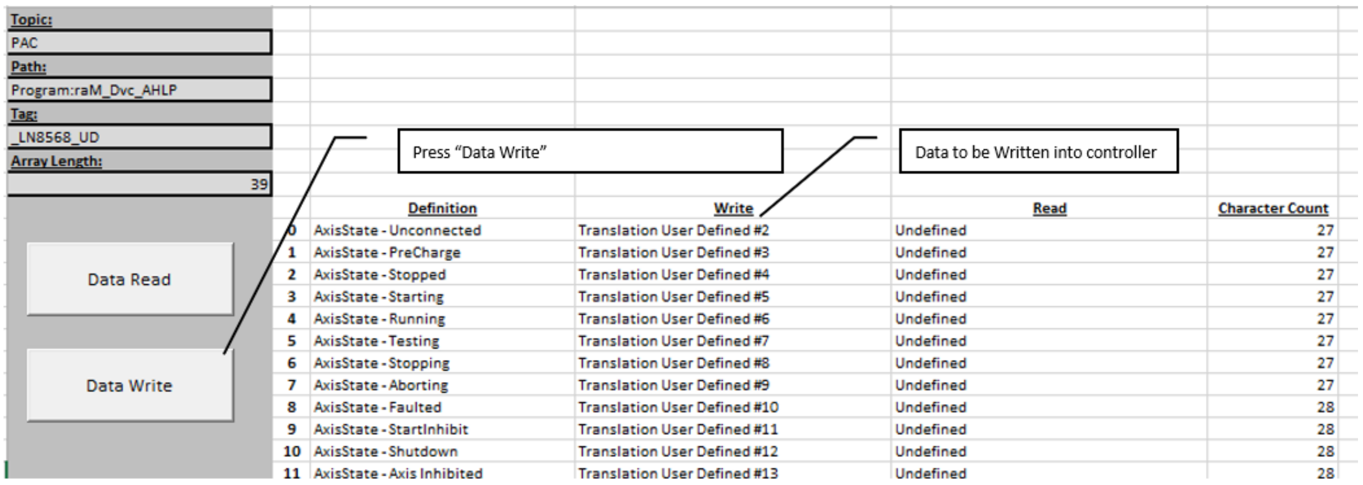


Topic:
PAC
Path:
Program:raM_Dvc_AHLP
Tag:
_LN8568_UD
Array Length:
39

Data read from Controller

	Definition	Write	Read	Character Count
0	AxisState - Unconnected		Undefined	0
1	AxisState - PreCharge		Undefined	0
2	AxisState - Stopped		Undefined	0
3	AxisState - Starting		Undefined	0
4	AxisState - Running		Undefined	0
5	AxisState - Testing		Undefined	0
6	AxisState - Stopping		Undefined	0
7	AxisState - Aborting		Undefined	0
8	AxisState - Faulted		Undefined	0
9	AxisState - StartInhibit		Undefined	0

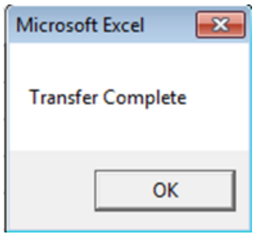
9. Enter data to be written to the controller in the workbook and press “Data Write” Push-button to write information to the User Defined Language Tags in the controller.



Topic: PAC
Path: Program:raM_Dvc_AHLP
Tag: _LN8568_UD
Array Length: 39

	Definition	Write	Read	Character Count
0	AxisState - Unconnected	Translation User Defined #2	Undefined	27
1	AxisState - PreCharge	Translation User Defined #3	Undefined	27
2	AxisState - Stopped	Translation User Defined #4	Undefined	27
3	AxisState - Starting	Translation User Defined #5	Undefined	27
4	AxisState - Running	Translation User Defined #6	Undefined	27
5	AxisState - Testing	Translation User Defined #7	Undefined	27
6	AxisState - Stopping	Translation User Defined #8	Undefined	27
7	AxisState - Aborting	Translation User Defined #9	Undefined	27
8	AxisState - Faulted	Translation User Defined #10	Undefined	28
9	AxisState - StartInhibit	Translation User Defined #11	Undefined	28
10	AxisState - Shutdown	Translation User Defined #12	Undefined	28
11	AxisState - Axis Inhibited	Translation User Defined #13	Undefined	28

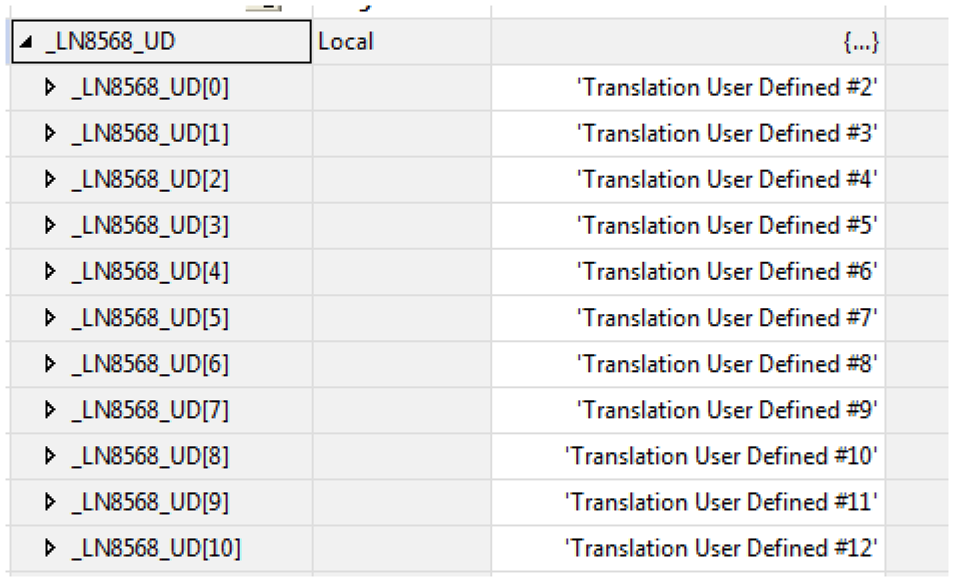
10. A message will be displayed that transfer has been completed.



Microsoft Excel

Transfer Complete

11. Data is available in the User Defined Language tags in the controller after data written command.



◢ _LN8568_UD	Local	{...}
▷ _LN8568_UD[0]		'Translation User Defined #2'
▷ _LN8568_UD[1]		'Translation User Defined #3'
▷ _LN8568_UD[2]		'Translation User Defined #4'
▷ _LN8568_UD[3]		'Translation User Defined #5'
▷ _LN8568_UD[4]		'Translation User Defined #6'
▷ _LN8568_UD[5]		'Translation User Defined #7'
▷ _LN8568_UD[6]		'Translation User Defined #8'
▷ _LN8568_UD[7]		'Translation User Defined #9'
▷ _LN8568_UD[8]		'Translation User Defined #10'
▷ _LN8568_UD[9]		'Translation User Defined #11'
▷ _LN8568_UD[10]		'Translation User Defined #12'

After writing new data online to the User Defined Tags, please switch to another language and then back to User Defined Language in order to load new User Defined values in the Inf_Text tag used by the HMI.

8 Learning Resources

8.1 Videos on Rockwell Automation YouTube Channel

Machine Builder Libraries – Axis Handler CIP Motion Drive

https://www.youtube.com/watch?v=_ayMGORxV38&list=PL3K_BigUXJ1PA9pt2WfmrPG5yPzPJrhRv&index=7

8.2 Labs available on Rockwell Automation Cloud Based Learning Environment (onCourse)

1. Applying Machine Builder Libraries to Develop ISA-TR88.00.02 (PackML) Based Machine Control
2. Applying Machine Builder Libraries to Develop ISA-TR88.00.02 Based Machine Control – Equipment Supervisor and EM Requests
3. Applying Machine Builder Libraries to Develop ISA-TR88.00.02 Based Machine Control – Control Events
4. Explore Machine Builder Libraries with Application Examples (VFFS)

Contact your local Rockwell Automation Sales/Support representative to schedule an instance of lab.

9 For More Information

Contact the Machine Builder Libraries team at *oemlibraries@ra.rockwell.com*.

10 Appendix

General

This document provides a programmer with details on this instruction for a Logix-based controller, its Application Code Manager library content, and visualization content, if applicable. This document assumes that the programmer is already 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 object includes a Logix Designer Asset for use with Version 30 or later of Studio 5000 Logix Designer.

Common Information for All Instructions

Rockwell Automation Application Content may 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

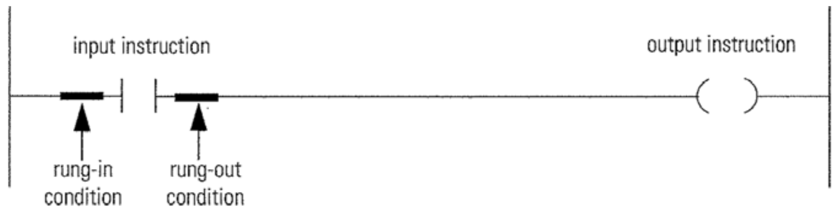
Instruction Execution - Edge and Continuous

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.