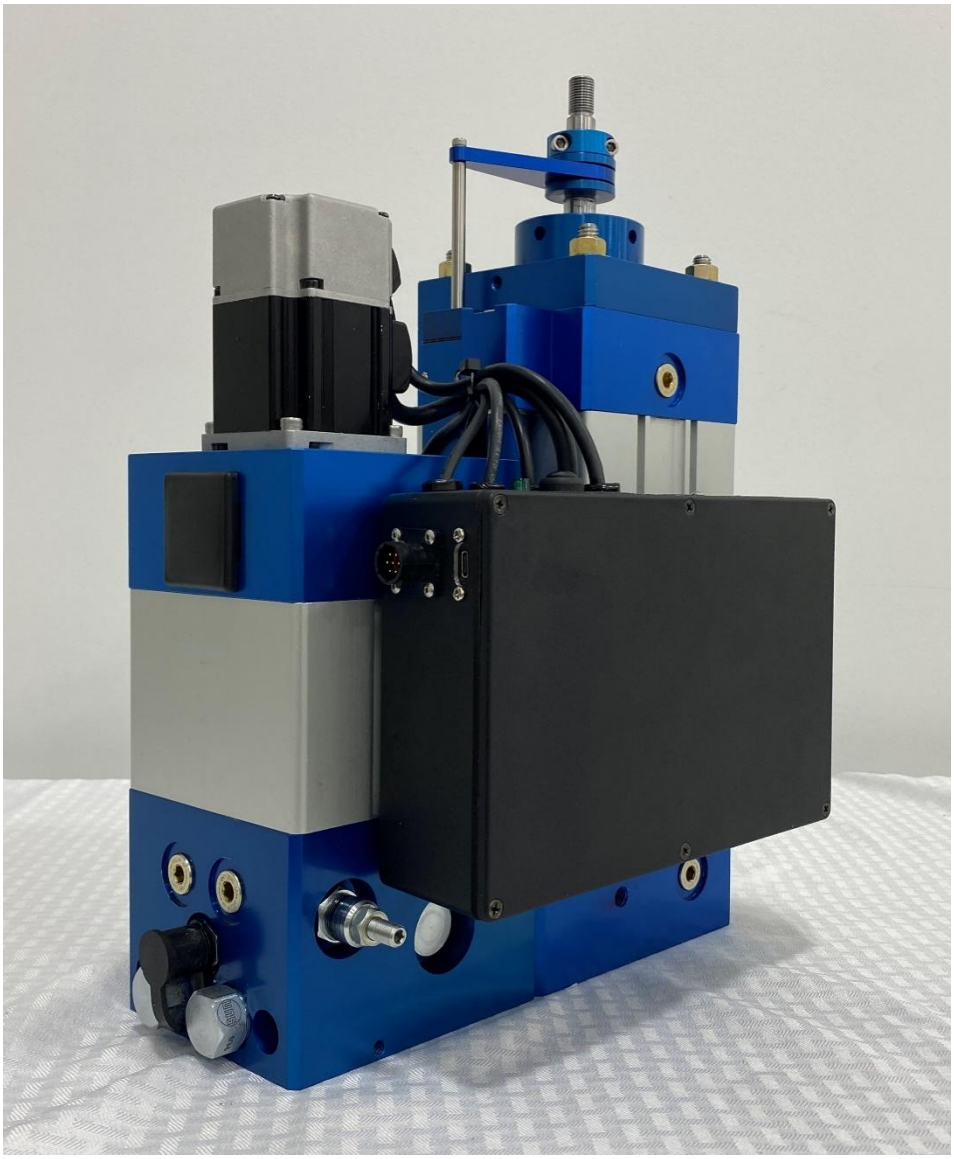
AOI for the Kyntronics SHA – User Document



REVISION HISTORY

Revision	Description of Change	Date (MM-DD-YYYY)	Revised by
A	Created	11-10-2021	SS
B	Add section 5.5 covering Solenoid Control Valves	12-9-2021	SS
C	Update to v1.2 SHA_Position, and add SHA_RelayId	1-17-2022	SS
D	Show 'Excessive Velocity Error' set to Ignore on SHA Axis.	2-2-2022	SS
E	V1.3 update: provide Position and TorqueLimit SHA AOIs.	3-17-2022	SS
F	V1.3 F update: add analog I/O and advanced tuning topics.	9-18-2023	SS
G	V1.4 G update: Enhance error conditions.	5-28-2024	SS
H	V1.4H update: zero offset on position sensor.	5-29-2025	SS

Contents

1. Assumptions

This document assumes the user is familiar with the AB software, understands motion and Add-On Instructions, and drive commissioning.

1.1 Kyntronics Smart Hydraulic Actuator (SHA) and the AOI

The Kyntronics AOI can be used with any AB motion command including synchronization utilizing the Rockwell software. The virtual axis is utilized so all available motion planner instructions can be used for the desired motion profile.

The Kyntronics AOI can only be used with Kyntronics products, any other use is strictly unauthorized.

It is assumed that proper safety standards and precautions will be followed when using the AOI.

1.2 Files Included in the Kyntronics_SHA_AOI_PosCntrl_V1_x_x.zip

This ZIP includes the following files:

SHA_PositionControl.L5X –Asset file with Position Control outer loop.

SHA_PositionControl_TL.L5X –With Torque Limited outer Control loop (added in v1.2).

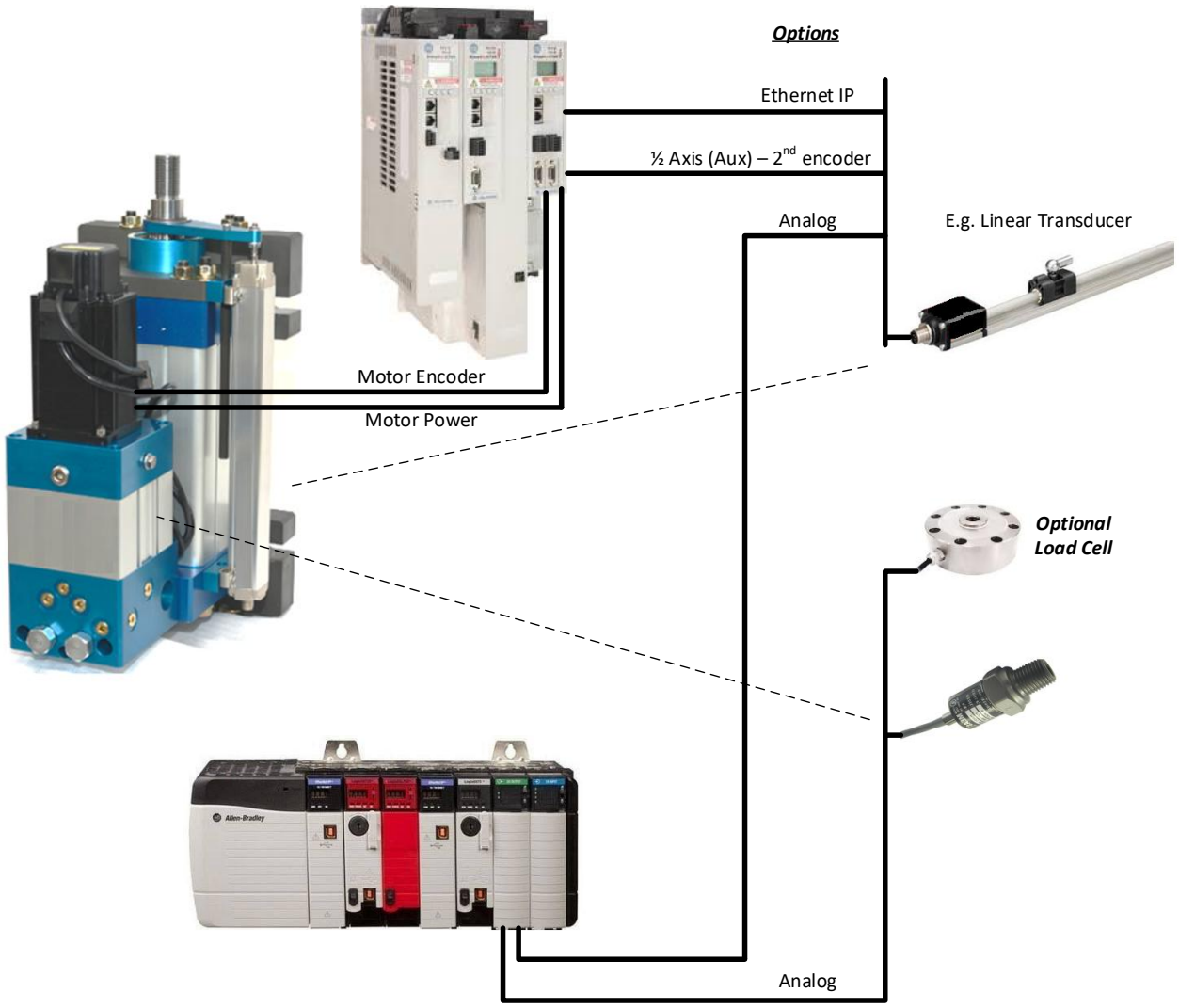
SHA_Config.L5X and SHA_Controller.L5X –The SHA dependency Asset files.

License.txt –The license information file.

SHA_022822_CmpLgx_AOI_V1_x_TorqueLimit.ACD –Sample RS Studio project file.
SHA_022822_CtlLgx_AOI_V1_x_PosnLimit.ACD –Sample RS Studio project file.

Version 1.4 is the latest AOI, with an update to correct a case where disabling the AOI did not clear I-gain windup, due to a known problem with AOIs written in Structured Text, which could cause an unexpected motion when the AOI is re-enabled. The v1.4 update detects and corrects this issue and clears the windup.

2. Hardware and Connectivity Options



2.1 Hardware Recommendations

- Use proper control methodology (i.e.—efficient wiring and networking).
- 1-2 millisecond Logix scan rates are recommended.
- Use the higher speed analog input card to achieve ~1ms RPI and 0.3 ms sample rates.
  - [1756-IF4FXOF2F, 4 Current/Volt Inputs, 2 Current/Volt Outputs Fast Analog.] **
- RA Rule of Thumb in CIP architecture: 32 axis/ms of group update rate.
- Use the ½ axis TTL input for the linear position (for higher precision requirements): **
  a. Expected Position repeatability is ~0.0005 – 0.002 inch.
  b. Expected Synchronization (using suggested AB sync methodology) is 0.002 – 0.004 inches, between axes.

** --Repeatability and accuracy depend on many variables, e.g.—mechanics, stroke, pump/cylinder, valves, etc.

2.1.1 Alternate Analog Input Modules

While the 1756/1769-IF4FXOF2F modules are preferred, as these are capable of 1 milli-second conversion times (which is the specification that matters). The RPI/Requested Packet Interval is simply the data exchange rate with the module, but it has no guarantee that new data is available. For example, if the RPI for a 5069-IY4 module is set to 1.0 milli-second, but the IY4 analog input takes 2 milli-seconds to convert the value, then the module data packet will be delivered to the PLC every 1 milli-second, but the data inside the packet will only change every 2 milli-seconds (every other data packet will have a new value for the analog input).

Point I/O modules, analog inputs:

If these must be used, for best update time do NOT allow packet optimization. From a conversion speed standpoint, a remote I/O rack on 100MBS Ethernet with an IY4 analog input module is faster (2 msec updates if 2 channels used). The Point I/O analog input speed is significantly slower:

Analog Input Modules

1734 Analog Input Modules Technical Specifications

	1734-IE2C	1734-IE2V	1734-IE4C	1734-IE8C
Number of inputs	2		4	8
Input signal range	4…20 mA 0…20 mA	0…10V ±10V	4…20 mA 0…20 mA	4…20 mA 0…20 mA
Input resolution	16 bits - over 21 mA 0.32 µA/cnt	15 bits plus sign 320 µV/cnt in unipolar or bipolar mode	16 bits - 0…21 mA 0.32 µA/cnt	
Data format	Signed integer			
Accuracy	Current Input: 0.1% Full Scale @ 25 °C(1)	Voltage Input: 0.1% Full Scale @ 25 °C(1)	Current Input: 0.1% Full Scale @ 25 °C(1)	
Accuracy drift w/temp.	Current Input: 30 ppm/°C	5 ppm/°C	30 ppm/°C	
Step response, per channel	70 ms @ Notch = 60 Hz (default) 80 ms @ Notch = 50 Hz 16 ms @ Notch = 250 Hz 8 ms @ Notch = 500 Hz		50 ms @ Notch = 60 Hz (default) 60 ms @ Notch = 50 Hz 30 ms @ Notch = 100 Hz 25 ms @ Notch = 120 Hz 15 ms @ Notch = 200 Hz 12.5 ms @ Notch = 240 Hz 10 ms @ Notch = 300 Hz 7.5 ms @ Notch = 400 Hz 6.25 ms @ Notch = 480 Hz	
Input conversion type	Delta Sigma		Sigma Delta	

Slower analog position and pressure feedback means the PI gains of the AOI will need to be reduced, and the AOI will be less responsive, and will take longer to reach a precise position or force setpoint.

2.2 Position Feedback and Control

The AOI supports position via:

- EtherNet/IP
- ½ Axis (Auxiliary Encoder)
- 0-10 Vdc
- -10 Vdc to +10 Vdc
- 4-20mA
- SSI (via AMCI module)

2.3 Force Feedback and Control

Pressure

- 0-10 Vdc

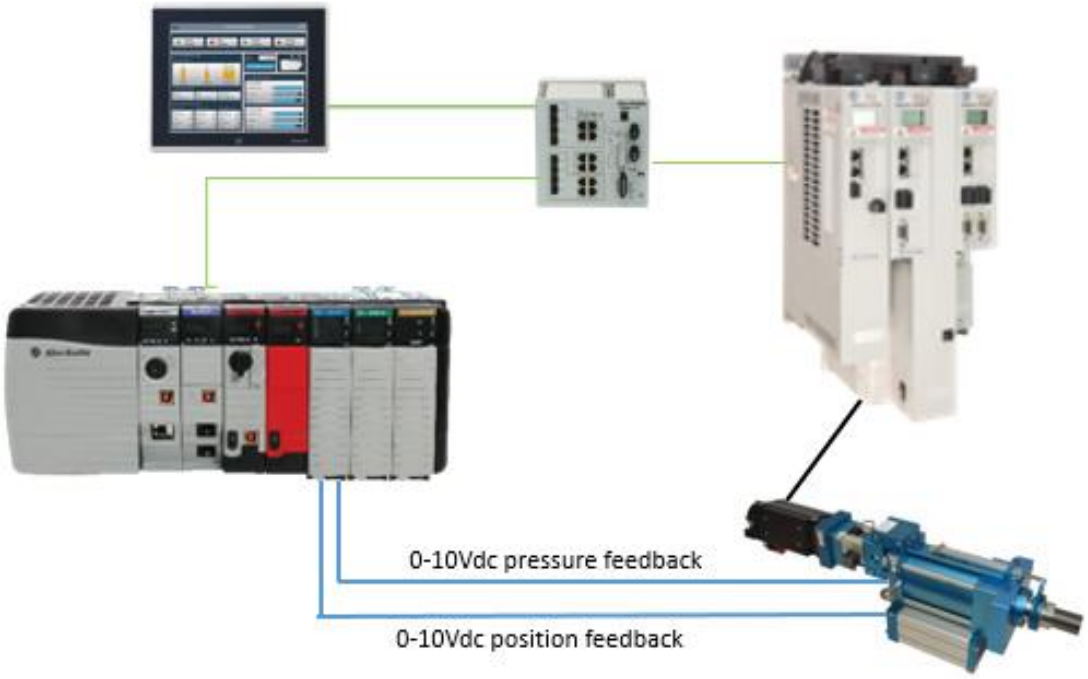
Load Cell

- 0-10 Vdc
- -10 Vdc to +10 Vdc
- 4-20mA
- Ethernet IP

2.4 Motor Connections

- AB Motor Power
- AB Encoder

3. Typical AB Motion Setup



3.1 Drives

Kinetix CIP (EtherNet/IP) Drive

- 5700 (optional Feedback Half Axis Available)
- 5500
- 5300
- Others

3.2 PLC

- Control Logix Controller, Motion rated
- Compact Logix Controller (64 bit controller versions), Motion rated

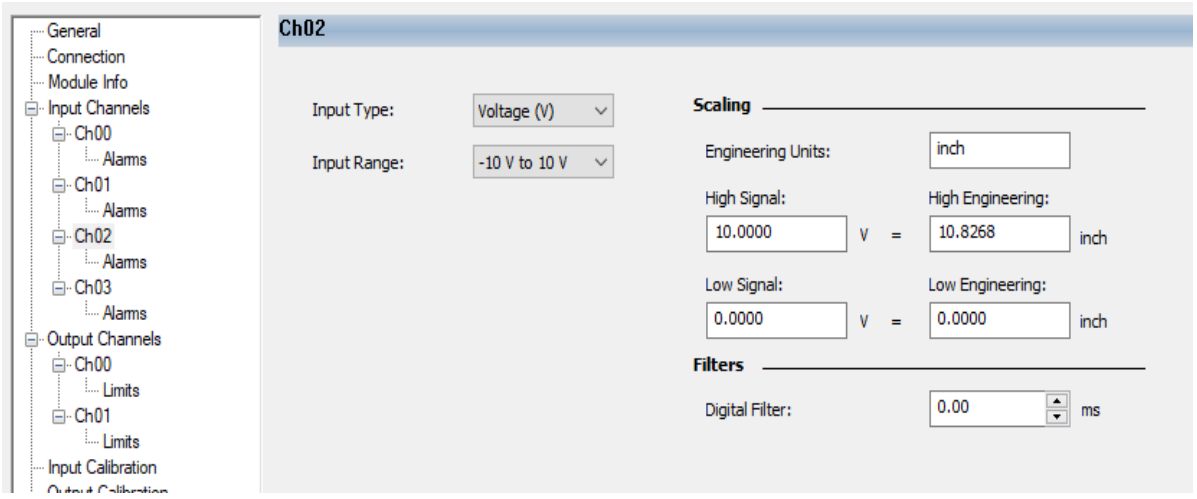
3.3 Network/Connectivity

Appropriate EtherNet/IP Switch recommended for motion or direct daisy chain per Rockwell recommendations.

4. SHA Specific Inputs

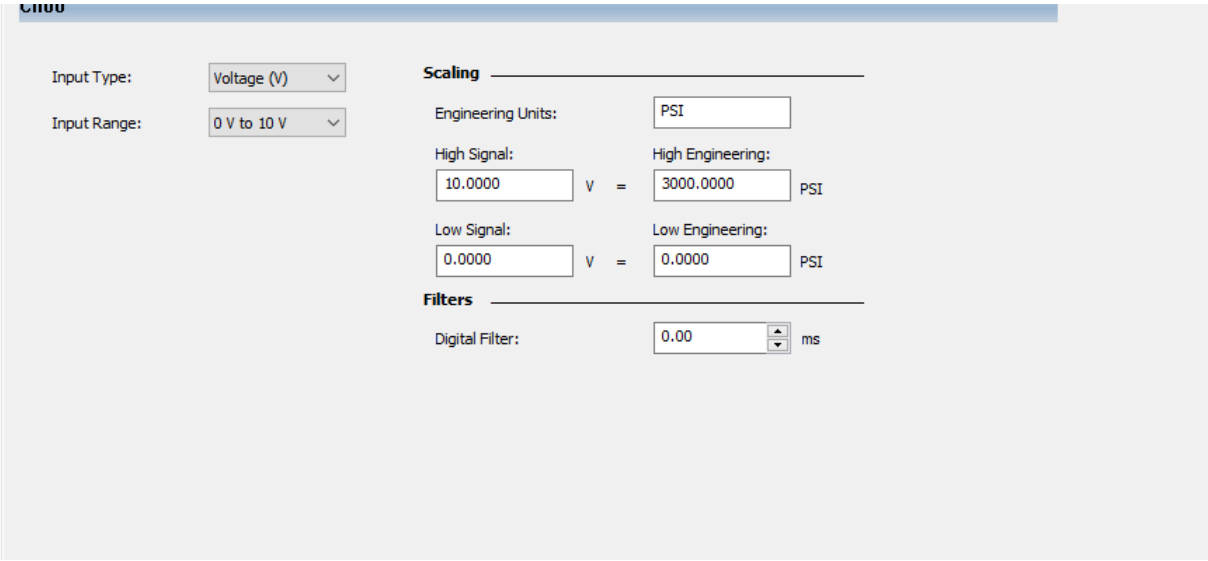
4.1 Position Feedback

- Required for Position Control or Position Limiting.
- Convention requires increasing position feedback for extension of the actuator.
- High resolution applications can use Hiperface or CIP Encoders as half axis (feedback only).
  - This is available on drives that support an additional “half axis” such as the Kinetix 5700. CIP Encoders are also available. These devices are simply added to the Motion Group and the feedback can be directly fed to the AOI.
  - Analog position feedback devices are also supported.
  - User provides scaling and calibration into engineering units (mm/inches).
  - Example program uses an IF4FXOF2F. Fast Analog Input card to achieve 1ms RPI (requested packet interval).
  - The screen below shows an example of an analog input configured for a linear transducer to give position feedback. Set the Engineering Units as desired: inch or mm typically. Then set the 0.00V value to 0.0 inch (or mm). The 10.000V reading is then set to the maximum value of the position sensor, adjusted to the specified engineering units. For example, for a 275mm position sensor, the inch equivalent is 10.8268 inches, as shown below. Since the actuator operators inboard of the sensor’s range, a Zero Offset is typically applied to the ”raw” feedback position. It is recommended to set the Zero Offset so that the fully retracted position reports as a slightly negative position (say, -0.08” or -2.00mm). In this way, the 0.0 inch (mm) position is then that far off of the retract internal hard stop, so that commanding the 0.0 inch (mm) position does not strike the internal hard stop.



4.2 Force Feedback

- Required for Force Control or Position with Force Limiting
- Convention assumes positive force is asserting in the extend direction
- Negative force is asserting in the retract direction
- Four quadrant operation is supported
  - Recommend working with Kyntronics to help size applications that require regenerative operation
- Analog feedback
  - Load Cell devices (Bi-polar voltage)
  - Example program uses an IFX4 Fast Analog Input card to achieve 1ms RPI
  -



5. SHA Project Prerequisites

5.1 Safety

- Ensure the necessary safety items are properly integrated.
- Ensure the designated maximum speed is not exceeded (usually 3000 rpm).
- Ensure the proper fault/alarm logic is implemented; see section 9.3

5.2 Kinetix Drive is configured in the Project

- Module is configured for the motor using the RA motor catalog number. The motor is used in velocity control, so no Autotune is needed. Likewise, the Load Observer is not needed and should not be used.

– The CIP Axis is added to the motion group, covered in section 6 below.

5.3Create a Motion Group

- Recommend 1millisecond update rate for best performance.

5.4 Confirm project can go online

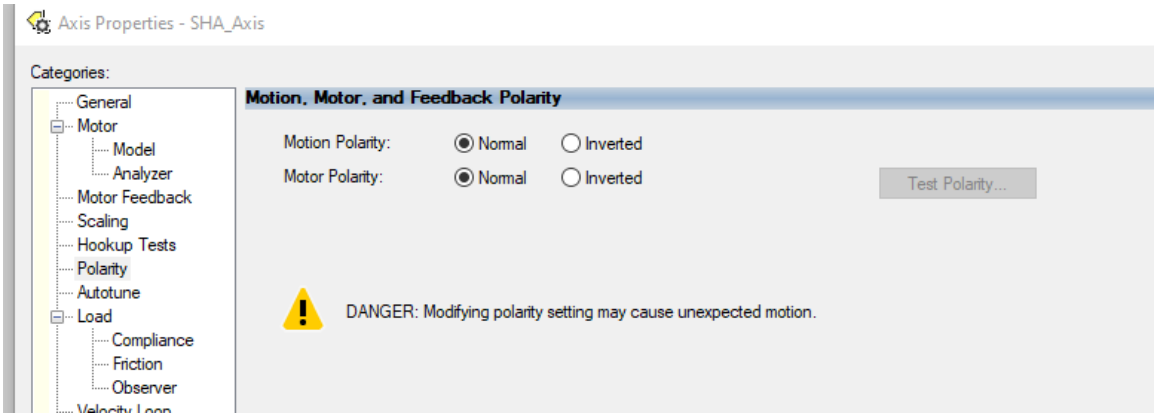
5.5 Retract Solenoid Control Valves and Other Control Valves

For SHAs that are used in a vertical orientation, a Retract Solenoid Control Valve is usually included. Prior to attempting any extend/retract motion, this solenoid should be powered, so that excessive pressure is not built up inside the SHA. Generally, the Retract Solenoid Control Valve will then act as a brake should power be lost, or an emergency stop occur.

In other cases, solenoid control valves may be used to provide additional fluid paths under certain circumstances; for example, to allow for higher speed extend or retract operation areas.

5.6 Jog Test

– Using Motion Direct Commands for the CIP Axis
– Execute MAJ (Motion Axis Jog)
– Forward Direction
– Use +10 Hz then -10 Hz
– Verify positive jog extends the actuator
– If not, the direction can be inverted using the Axis Properties Menu -> Polarity, Motion Polarity Invert setting.



6. Override Default Options in General Configuration

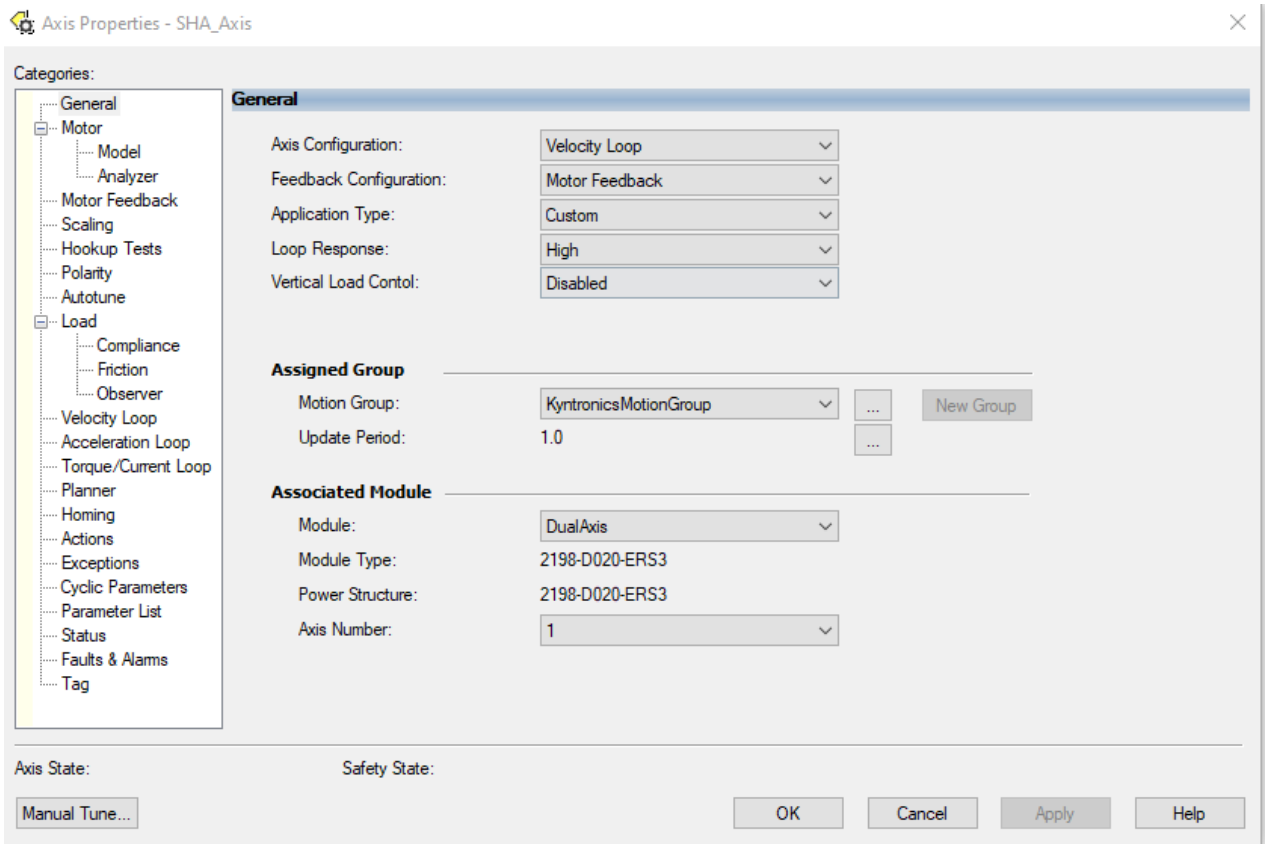
6.1 CIP Axis Configuration

Select:

Velocity Loop
Custom
High Loop Response

Note: Because the motor drives the pump to move the actuator, no Autotune is needed; the default parameters for the motor from its catalog number are sufficient. Likewise, the Load→Observer does not need to be used or run.

Axis Properties screen, for SHA_Axis:



6.2 Verify CIP Axis Motion Polarity

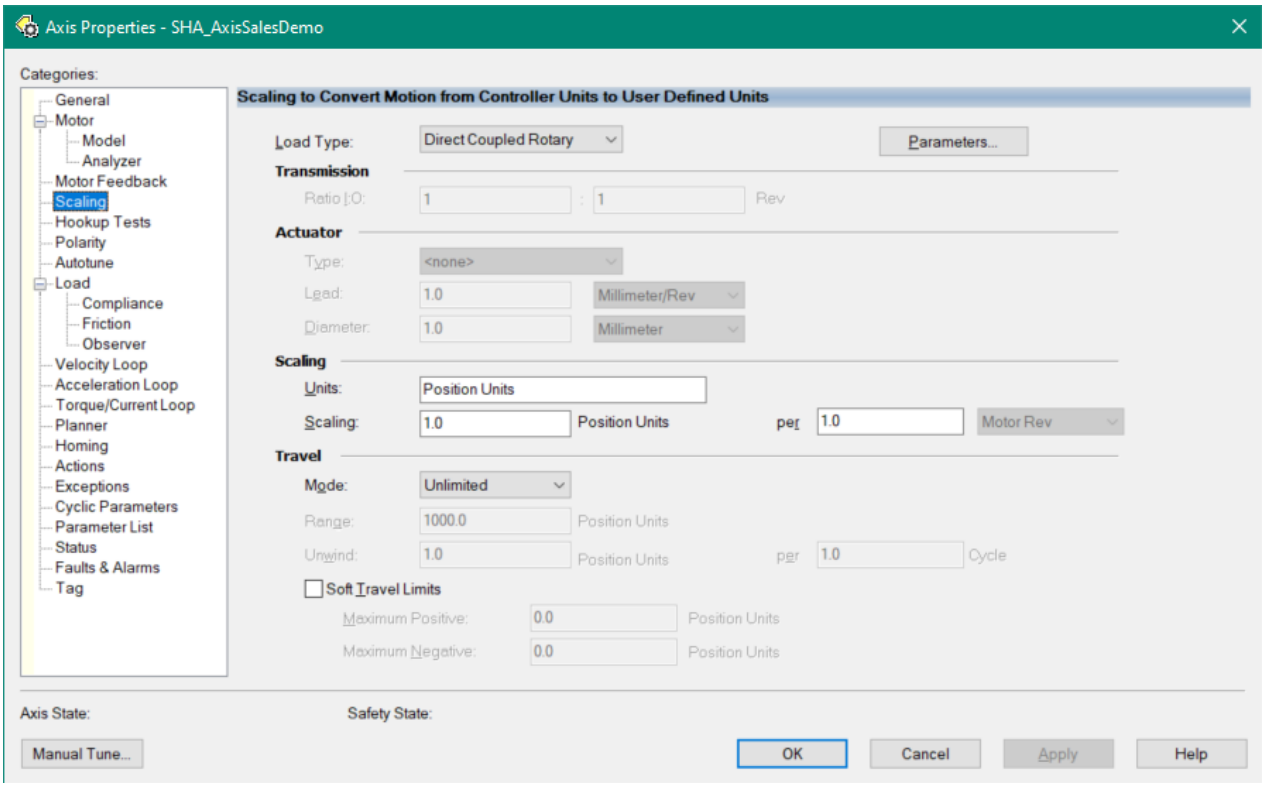
- **A positive jog reference should extend the actuator**
- **A negative jog reference should retract the actuator**

If a positive jog reference retracts the actuator, then invert the motion polarity of the SHA_Axis, using Axis→Properties→Motor→Polarity.

6.3 Units for the Kinetix Axis

User units for the Kinetix Axis will remain 1:1 such that all speeds to the drive are in units of Hertz. Set the Travel Mode to Unlimited.

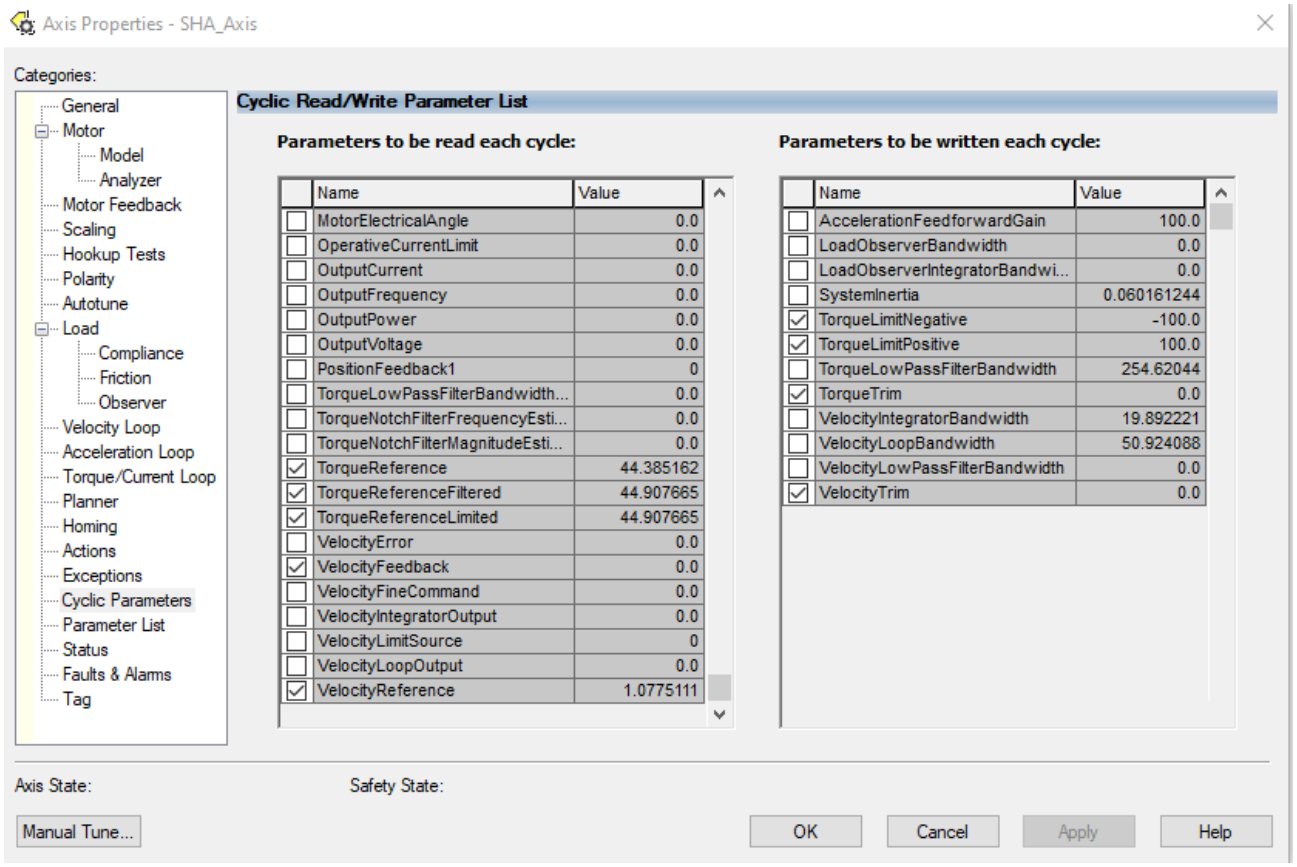
The minimum and maximum pump speed parameters in the SHA_CONFIG tags are also in units of Hertz. (So 5 Hz => 5 cycles/sec * 60 seconds/minute => 300 RPM.)



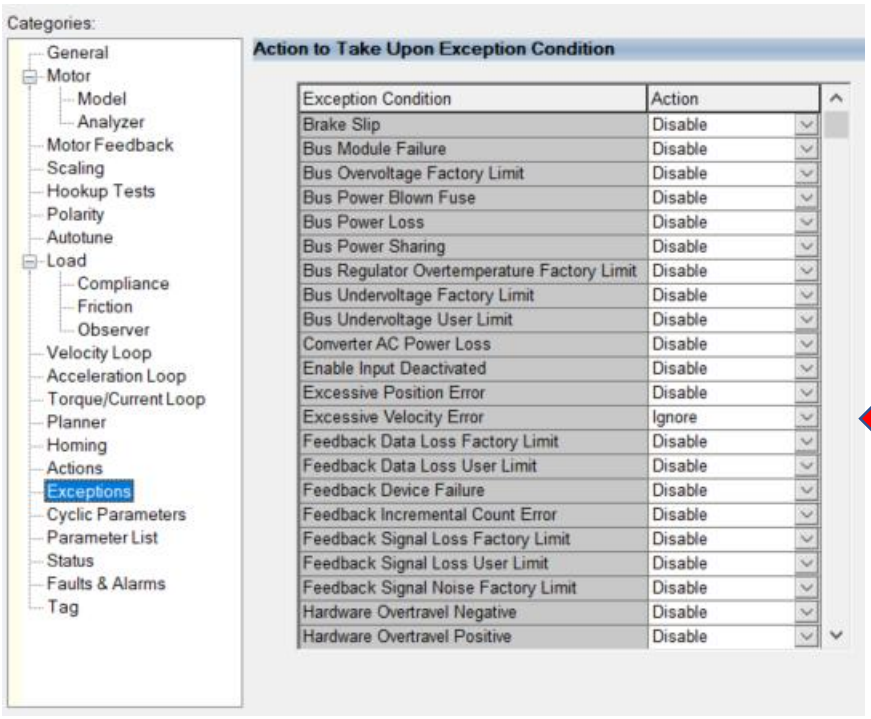
6.4 Additional CIP Drive Parameters are Required

The AOI requires non-default feedback and control parameters to be exchanged between the CIP Axis backing tag and the Kinetix drive.

Enable the parameters as shown below:



In the Exceptions table, set the Excessive Velocity Error to Ignore:

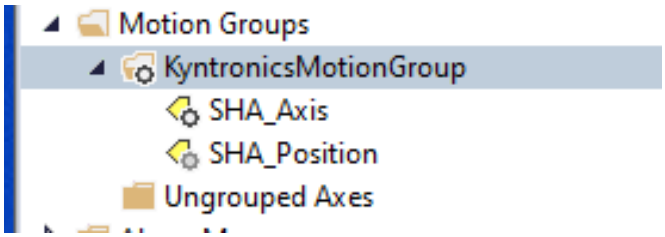

Categories:
General
Motor
Model
Analyzer
Motor Feedback
Scaling
Hookup Tests
Polarity
Autotune
Load
Compliance
Friction
Observer
Velocity Loop
Acceleration Loop
Torque/Current Loop
Planner
Homing
Actions
Exceptions
Cyclic Parameters
Parameter List
Status
Faults & Alarms
Tag
Action to Take Upon Exception Condition
Exception Condition
Action
Brake Slip
Disable
Bus Module Failure
Disable
Bus Overvoltage Factory Limit
Disable
Bus Power Blown Fuse
Disable
Bus Power Loss
Disable
Bus Power Sharing
Disable
Bus Regulator Overtemperature Factory Limit
Disable
Bus Undervoltage Factory Limit
Disable
Bus Undervoltage User Limit
Disable
Converter AC Power Loss
Disable
Enable Input Deactivated
Disable
Excessive Position Error
Disable
Excessive Velocity Error
Ignore
Feedback Data Loss Factory Limit
Disable
Feedback Data Loss User Limit
Disable
Feedback Device Failure
Disable
Feedback Incremental Count Error
Disable
Feedback Signal Loss Factory Limit
Disable
Feedback Signal Loss User Limit
Disable
Feedback Signal Noise Factory Limit
Disable
Hardware Overtravel Negative
Disable
Hardware Overtravel Positive
Disable

7. Setup a Motion Group

7.1 Motion Groups

In the Motion Groups, setup:

- One CIP AXIS for the Kinetix Drive itself (SHA_Axis)
- If using half axis feedback, you will have another feedback-only axis listed in the menu.
- It is recommended to use a VIRTUAL Axis to leverage the motion planner instructions (MAM, MAJ, MAPC, etc.) the SHA_Position axis:



8. Import the AOI, v1.4

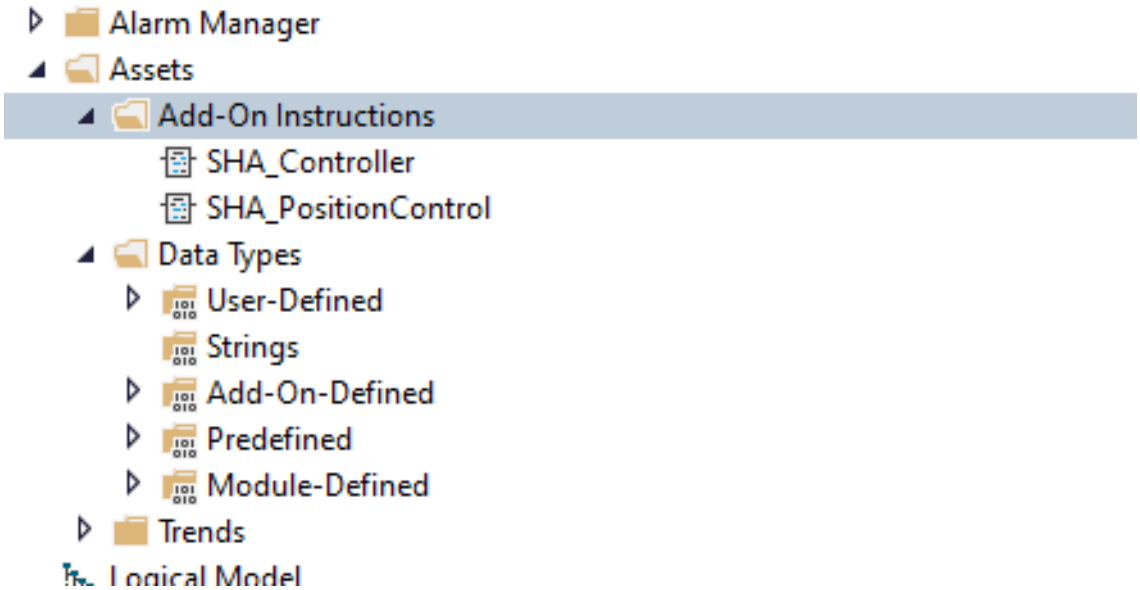
8.1 In the Project tree, Assets:

Right click to Import the AOI.

Select the SHA_PositionControl or the SHA_PositionControl_TL.

This will also import the dependencies of:

- SHA_Controller (AOI)
- SHA_Config (User Defined Type UDT)



8.2 Import the SHA_RelayId AOI

With v1.2 and higher, the SHA_RelayId AOI is also included, which can be used to perform position and force tunings of the SHA. The sample projects include 2 instances of the SHA_RelayId in the SHA_Routine ladder: one for Position Tuning and one for Force Tuning. These are designed to be used in conjunction with the SHA_Position control block. Once the tuning is complete, these blocks can be removed from the project, if desired.

The Force tuning SHA_RelayId can also be used for a “poor man’s” open-loop force control, although it is best to use the SHA_PositionControl for closed-loop force control in most applications.

See Section 10 for further details on the SHA_RelayId.

8.3 Add SHA_Config as a Public Tag

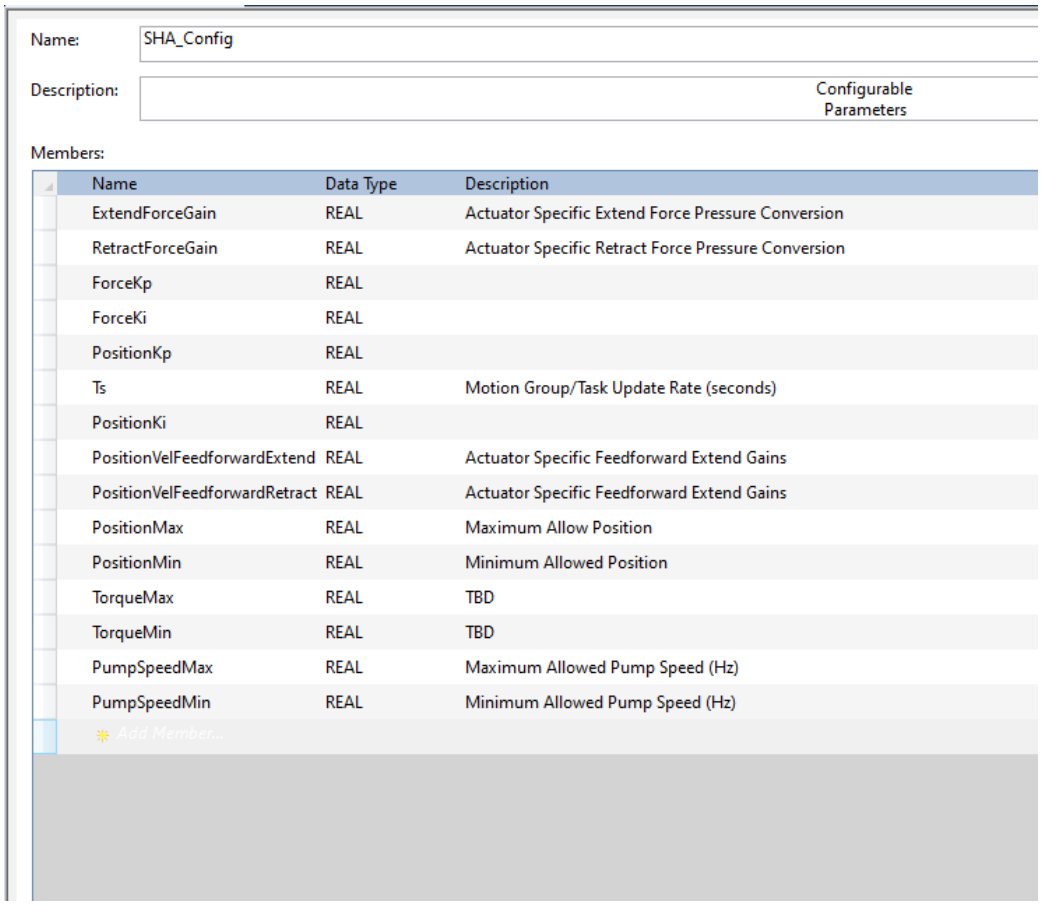
Create a MotionTask, such as SHA_MotionTask_Program. Create a public instance of SHA_Config there. The SHA program can be made here; for example, SHA_Routine.

Kyntronics will provide the default values for the SHA_Config members, based on the specific actuator and application.

Note:

- The ExendForceGain and RetractForceGain values are set based on the geometry of the specific SHA, to properly calculate the Force from the pressure feedback, so these values should never be changed from the factory provided values (except if a load cell will be used to provide the force feedback; see the description of the Extend/Retract Pressure Feedbacks in Section 8.6 below).

These values allow the configuration to be tailored to the actuator geometry as well as allow the end user to adapt the tuning and applicable limits.

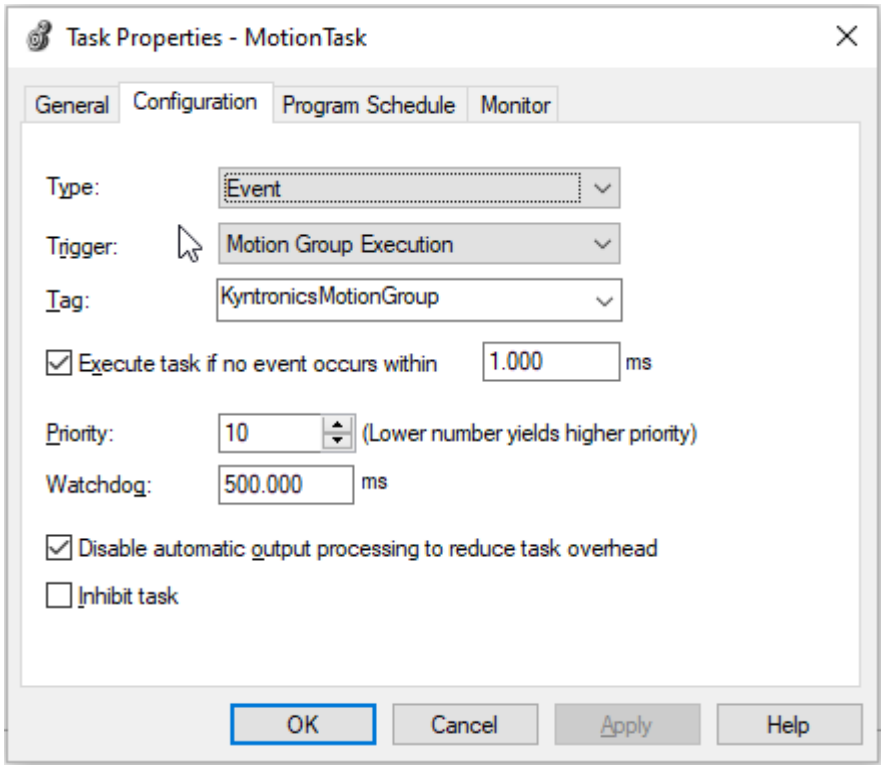


Name	Data Type	Description
ExtendForceGain	REAL	Actuator Specific Extend Force Pressure Conversion
RetractForceGain	REAL	Actuator Specific Retract Force Pressure Conversion
ForceKp	REAL	
ForceKi	REAL	
PositionKp	REAL	
Ts	REAL	Motion Group/Task Update Rate (seconds)
PositionKi	REAL	
PositionVelFeedforwardExtend	REAL	Actuator Specific Feedforward Extend Gains
PositionVelFeedforwardRetract	REAL	Actuator Specific Feedforward Extend Gains
PositionMax	REAL	Maximum Allow Position
PositionMin	REAL	Minimum Allowed Position
TorqueMax	REAL	TBD
TorqueMin	REAL	TBD
PumpSpeedMax	REAL	Maximum Allowed Pump Speed (Hz)
PumpSpeedMin	REAL	Minimum Allowed Pump Speed (Hz)
* Add Member...		

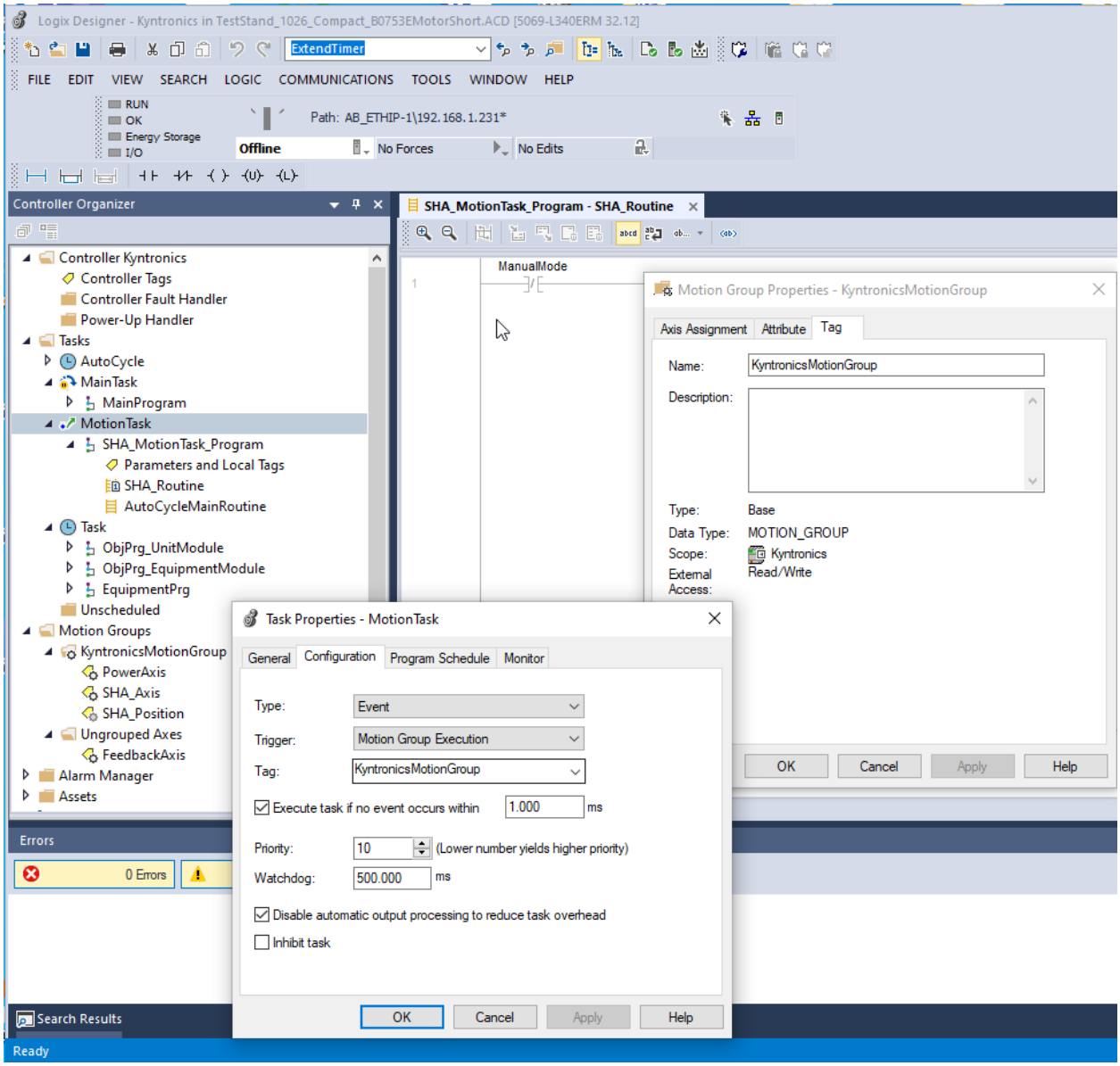
8.3 Set the Motion Task Execution Time

The AOI is intended to run in the motion group task context. This will allow the code to execute synchronously to the motion group coarse update period.

A task can be created using the Tasks -> Right Click -> New Task menu in the project tree. Enter your desired settings:



8.4 Coordinate the Motion Task and the Kyntronics Motion Group



8.5 The Motion Task runs the SHA AOI

Insert the SHA_PositionControl (which uses position control on the outer loop) or the SHA_PostionControl_TL (which uses Torque Limiting control on the outer loop) AOI onto a rung, in the motion group task (shown below). Generally, on faster, higher-end PLC systems, the Position Control version is preferred, while on slower systems, the TL version may do better. Both versions can be inserted for initial testing, if desired, so long as only one is ever enabled at a time. Use separate SHA_Config instances if both are present. After running ForceTrends to compare, the other AOI block can be removed.

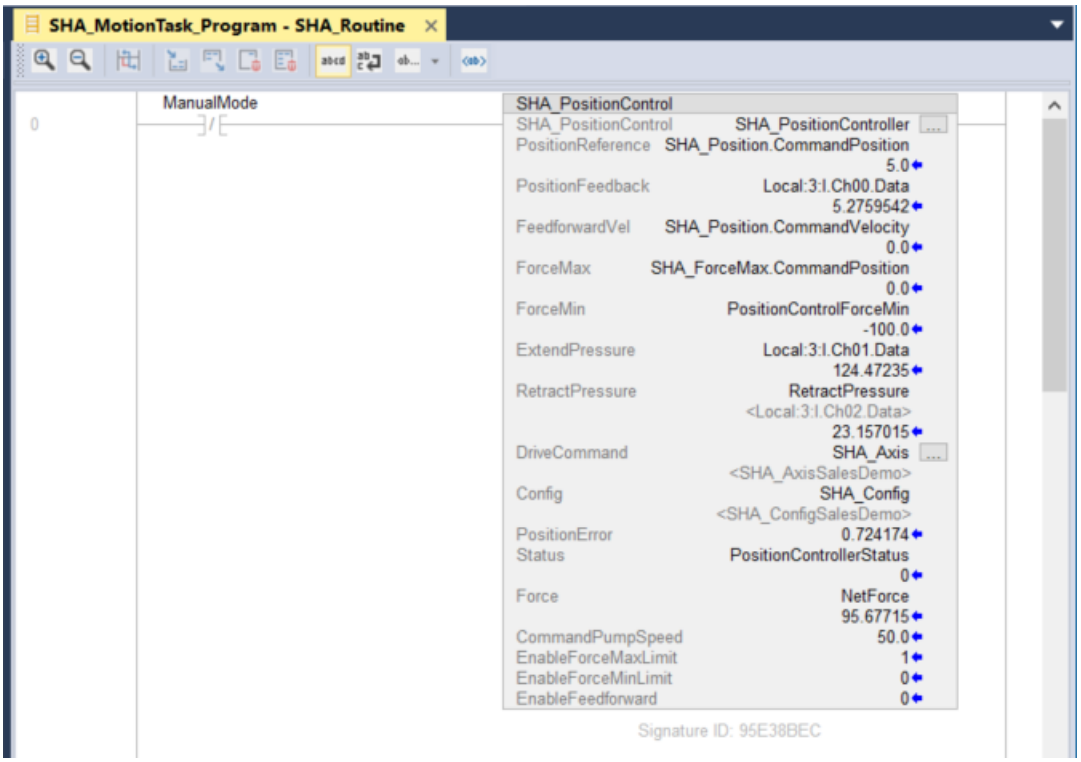
This AOI is executed unconditionally to allow for bump-less transfer from closed to open loop, and to continually update the Force output based on the Pressure feedback. The AOI is monitoring the actual position. The application developer is free to not use the commanded pump speed if manual mode is desired.

For example, this XIC allows for toggling between closed loop where the AOI determines the pump speed versus a Manual Mode where the velocity can be controlled open loop.

Closed loop control is enabled when the AOI output for pump speed is commanded to the CIP AXIS velocity trim via SSV or MOV instruction.

WARNING: When the AOI goes into closed loop control, if the PositionReference: SHA_Position.CommandPosition is different than the SHA PositionFeedback value, then motion can immediately occur, to move to the CommandPosition! It is best to use MRP to set the PositionReference tag to the current PositionFeedback value to prevent such unwanted motion.

The sample project uses a ManualMode tag to disable the SHA_Position block, so open loop movements of the SHA can be made, to jog and calibrate the position feedback during the initial setup. Or, there may be maintenance reasons to use the manual mode. But care needs to be taken when returning to closed loop control. Logic can be added to update the SHA_Position.CommandPosition to the PositionFeedback value just prior to setting ManualMode false, to help prevent unexpected motion.



8.6 Explanation of SHA AOI Members

Inputs:

- Backing AOI Tag Instance
- PositionReference (REAL, Position Units)
  - Recommend the use of a VIRTUAL AXIS and simply use <AXIS>.CommandPosition for this input.
  - This allows for the motion planner to generate MAM, MAJ, MAG, CAM, etc.
  - Alternative is to simply provide a value, but kinematic profiles are typically required.
- PositionFeedback (REAL, Position Units)
  - Scaled feedback from high-speed analog input, etc.
  - For analog input, set the notch filter to the highest value for fastest updates.
  - If scaling/offset needs to be done in ladder, then run that logic in this same task, and run it before the SHA_PositionControl is run.
  - Coordinate the positional resolution with the PositionDeadband value (0.001 typical).
- FeedforwardVel (REAL, Position Units/Second)
  - Recommend the use of a VIRTUAL AXIS and simply use its CommandVelocity for this input.
- Force Limits
  - Max (REAL, lbs)
  - Min (REAL, lbs)
  - Same recommendation: use a virtual axis for needed profiles if the Force limit needs to be changed/ranged in a controlled manner, otherwise a rate-limited or a fixed value variable can be used for simpler applications.
  - Force Limit control coordinates with the ForceLimitingDeadband value (typically 10 to 25).
  - Force Limit control also respects the SHA_Config.TorqueMax (for extend force) and SHA_Config.TorqueMin (for retract force). If these Torque Min/Max settings are set too low, they can prevent the specified ForceMax/ForceMin from being reached. Conversely, setting the TorqueMin or Max values just slightly higher than what is needed to reach the desired Force value helps minimize Force overshoot when entering the Force limit control zone.
- Extend/Retract Pressure Feedbacks
  - Provide Feedback (REAL, PSI)
  - In this demo, IF4FXOF2F analog inputs channels 0,1, and 2 were aliased to extend pressure, retract pressure, analog linear feedback (Position).
  - Use of Volts to Engineering Units is accomplished in the IF4X channel config.
  - If analog feedback is used, set the notch filter to the highest value for fastest updates; disable unused channels. If ladder scaling/offset is used, it must also run in the MotionTask and must be run prior to running the SHA_PositionControl block.
  - Load cell force feedback supported using Extend Pressure input in desired force units and SHA_Config tag ExendForceGain set to 1.0
  - Extend / Retract can be optionally set to 0 when not available.

- Config
  - SHA_CONFIG Data tag, public user defined data type.
  - The SHA_Config data values are provided by Kyntronics for out of box tuning, based on the SHA size, the configuration and application.
  - Use the Position and Force SHA_RelayId blocks, as needed, to help tune your system if the RPI is slower than 1 millisecond or if the position and pressure feedback are slower than 1 millisecond. See Section 10 for more information on the SHA_RelayId blocks.
- DriveCommand
  - The CIP AXIS Tag associated with the actuator
  - AOI uses diagnostic and feedback data available in tags each AOI scan.
  - User explicitly sets velocity command in ladder (See Motion Group Task)
- EnableForceMaxLimit, BOOL
  - Set True to enable the extend ForceMax value to be respected and applied.
- EnableForceMinLimit, BOOL
  - Set True to enable the retract ForceMin value to be respected and applied.
- EnableFeedforward, BOOL
  - Set True to enable the position velocity feedforward values to be respected and applied, which is useful when high speed positional moves are desired. Set False to disable, which can be useful when moving slowly into a Force Limit zone, to lessen any “bounce back” when the Force limit zone is small and/or the Force will rise very rapidly.

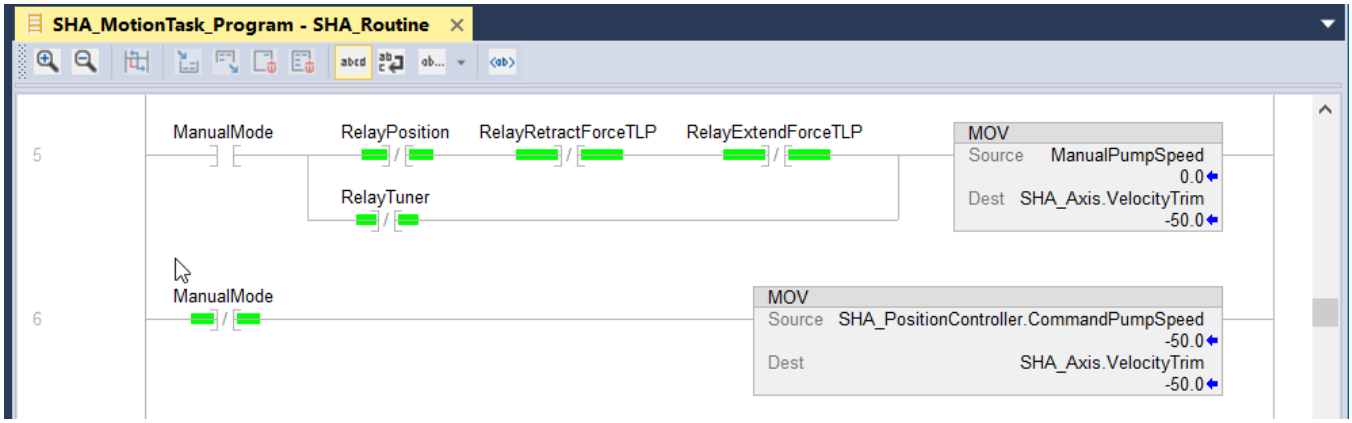
8.7 Explanations of AOI Outputs

- Status (DINT bit flags)
  - Error Codes
  - 0 for no error
- Force (REAL, lbs)
  - Current effective actuator force
  - Positive is extending push, negative is pulling retract.
- Command Pump Speed (REAL, Hz)
  - CIP AXIS is setup for 1 position unit per rev, so pump speed has units of Hz.
  - User can choose to use the AOI controller value or not.
  - Controller block will track current CIP axis speed and torque such that the user can choose when to close the loop.
  - Positive Speed extends, negative speed retracts.

8.8 Rungs to apply SHA CommandPumpSpeed (or to jog manually)

The rungs shown below are typically required, and should be placed below/after the SHA they work with. These rungs are needed for each actuator/SHA_PositionControl block.

The rung 3 shown below show applies a ManualPumpSpeed tag to the SHA_Axis VelocityTrim member for a manual jog (positive values extend, in Hz; negative values retract, in Hz). It will also serve to set the VelocityTrim to 0 (and so stop the manual mode SHA) when the SHA_RelayId blocks are stopped (as they typically leave the VelocityTrim non-zero when they are disabled).
Rung 4 is necessary to apply the SHA_PositionControl's CommandPumpSpeed to the axis when not in ManualMode, to allow for the closed loop control of the SHA.



9. Virtual Command Axis

9.1 Virtual Command Axis Initialization for Position Control

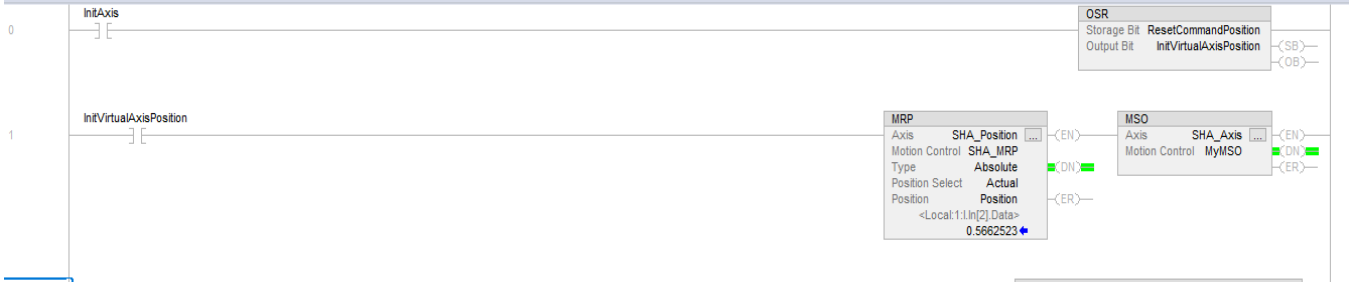
When using a virtual axis to command motion instructions to generate motion trajectories, you must initialize the virtual axis to the initial position of the actuator prior to closing the loop.

The Dynamics tab of the virtual Axis Properties can be used to set the maximum velocity, acceleration and jerk. Typically the maximum velocity is set to the (higher) Retract Speed, as that is the actuator's fastest speed. But remember to observe the quoted Extend Speed whenever extend motion is being commanded!

See the example below for how this can be done.

Hint: A virtual axis' command and actual positions will both be updated when using this MRP instruction.

Also, use a simple MSO in the CIP axis to enable active velocity control to the Kinetix drive.



9.2 Commanding Motion

Use motion instructions targeting the SHA Virtual Axis to generate the commanded motion profile: MAM, MAJ, MAG, MAPC, etc. Use of S-Curve motion is recommended (do not use Trapezoidal motion).

Keep in mind the virtual axis kinematic limits are also provided by Kyntronics such that you can use “percent of” or absolute kinematic limits on each instruction.

10. SHA_RelayId Tuning Blocks

With v1.2 and higher of the Kyntronics AOI, the SHA_RelayId block is included, which can be used for Position and Force tuning, and for open-loop Force control, if needed.

When a ControlLogix PLC and fast-update position and pressure feedback are used, with a 1 millisecond RPI as recommended, the SHA_PositionControl block is recommended over the SHA_PositionControl_TL block. Note that the RelayId tuning method will be different depending on which SHA control block is used.

However, on other slower systems, the SHA_Config Position and Force feedback tuning values will need to be softer, so the SHA_RelayId blocks can be added to the project to setup and test the tuning parameters. Once the tuning is finished, the SHA_RelayId blocks can be removed from the project, if desired.

"Hot" Positional Tuning
For faster force reaction, it is better to have hotter positional tuning, as the SHA_PositionControl AOI has an inner torque control loop and an outer position control loop.
With large motors, there may be too much vibration with more aggressive position tuning values. In these cases, the SHA_Axis Properties, Autotune tab's Loop Response can be set to Medium or Low to help. Also note the SHA_PositionController tag structures has both PositionDeadband and ForceDeadband settings which can be used to further reduce "hot" tuning.
Observe the feedback position, to see how large the +- variance is sitting idle, and that should give a gauge of the setting to use in the PositionDeadband. Typically this will be 0.0005" (0.013mm) as a starting point. The ForceDeadband is based on the smallest change of the analog input versus the max force of the actuator. The ForceDeadband is typically ~5 to 25 lbf, depending on the size/max force.

10.1 Traditional PI Position Tuning

Tradition PI Position Tuning can also be used. Make sure the EnableForceMaxLimit, EnableForceMinLimit and EnableFeedforwards are off/disabled, prior to starting position tuning.

With traditional PI tuning, the I gain is turned off or set very low (0.1), and then the P-gain is raised (for example, double the P value at each try), and small extend and retract moves are commanded around the middle of the actuator stroke, to minimize running into either hard stop. When the P-gain is too high, then instability will occur, and the P value is then reduced into the stabile range.

An SHA_Position Trend is included in the example project, which is helpful to observe the result. You may want to use a slower velocity for the initial moves: one quarter to one half the rated extend speed, until the P gain value is established.

If a move leaves the actuator far off the commanded position, then the I-gain can be raised temporarily to move the actuator to the commanded position, before running the next move.

Once the P-gain is finished, the I-gain is often set to around 10% of the P-gain value. However, depending on how the actuator will be used, the I-gain may need to be raised, and sometimes raised close to or above the P-gain, for processes that are force oriented, to help reduce “bouncing” when entering the force impact zone.

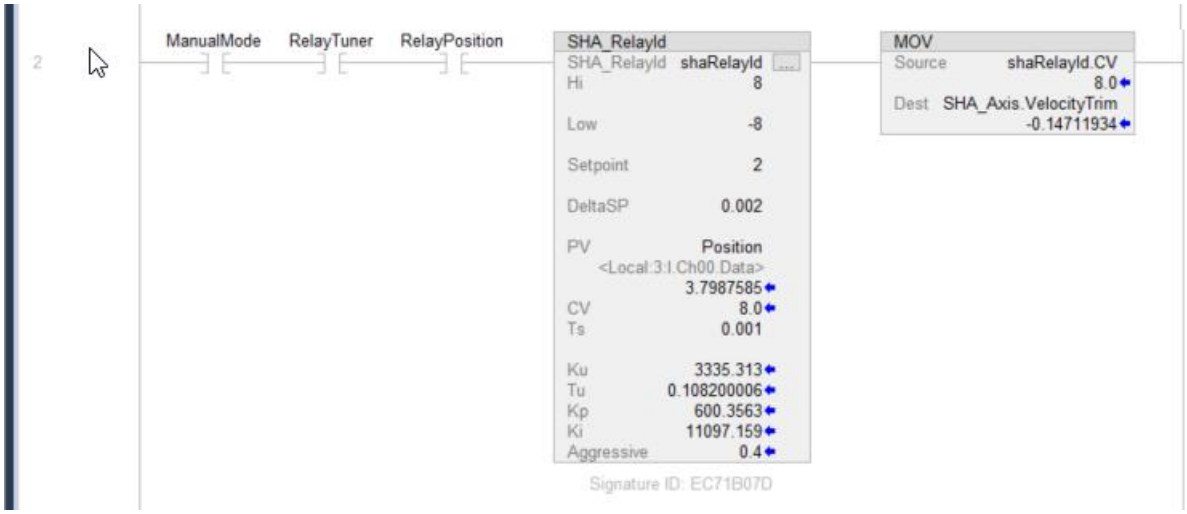
10.2 Using SHA_RelayId For Position PID Tuning

The sample project includes ladder logic with an instance of the SHA_RelayId setup for Position tuning, shown below. The SHA_PositionControl (or TL) block and related ladder logic for it are also expected to be in place. However, the SHA_PositionControl block must be disabled when the SHA_RelayId block is active. Also, if SHA_RelayId blocks for both Position and Force are present, only one of these RelayId blocks can be active at a time.

The intent of the Position Tuning block is to move the SHA back and forth between two positions, and measure the observed response, to provide position Ki and Kp values.

The AOI should be disabled while the SHA_RelayId blocks are being run!

Notes:
Running the RelayId block can cause the entire SHA to shake and move; make sure it is securely mounted or clamped prior to enabling the block!



10.2.1 SHA_RelayId Position Tuning Parameters

In the position tuning example above, typical input values are shown. The Kp and Ki output values will need to be entered into the SHA_Config.PositionKp and PositionKi tags for this SHA, and then tested using moves with the SHA_PositionControl block.

- Hi – Represents the highest extend command pump speed to use, when moving to the Setpoint position + the DeltaSP (SetPoint) position.
- Low – Represents the maximum retract command pump speed to use, when moving from the Setpoint + DeltaSp position, back to the Setpoint – DeltaSP position. This Lo value must be negative, and is typically the same magnitude as the Hi value.
- Setpoint – Represents the starting position to use. From this starting position, the actuator will be jogged back and forth by the DeltaSP offset. In the example, this is a 2” extend position, as inches are being used. If in metric, this would instead be a value in mm.
- DeltaSP – Represents the position offset value to extend and retract to, from the setpoint. In the example, this is 0.002 inches. Depending on the position feedback resolution, a slightly higher value may be needed.
- PV – Represents the Process Variable, so in a Position use case, this is the scaled position feedback value.
- CV – Represents the Control Variable. In the Position case, this is the value to move into the axis.VelocityTrim to provide the motion.
- Ts – Represents the RPI time being used for the MotionTask, in seconds. The default value of 0.001 represents 1 millisecond.
- Ku – Represents the amplitude of the position ripple observed when the RelayId block runs. The value should stay consistent, if the feedback is being provided at a constant rate.
- Tu – Represents the observed time-period of the position oscillations. The value should be fairly steady, if the feedback is being provided at a constant rate.
- Kp – Represents the positional Kp value calculated by the RelayId block, for the given Aggressive value. This value is the average of the last 10 oscillations.
- Ki – Represents the positional Ki value calculated by the RelayId block, for the given Aggressive value, and is the average from the last 10 oscillations.
- Aggressive – Represents a percentage of the “hottest” tuning values to use when calculating the Ki and Kp. So a value of 0.2 represents PID values that are 20% of the “hottest” values (the values that would be used if Aggressive = 1.0). Depending on the slowness of the RPI being used and the speed of the feedback data, an Aggressive setting of 0.2 is a good starting point. Put the Kp and Ki values into the Position members of the SHA_Config, and then try some moves using the SHA_PositionControl to observe the results. For faster systems, an Aggressive of 0.4 may give better results. For faster systems using the SHA_PositionControl block (rather than the TL version), an Aggressive of 0.5 is a good starting point.

Notes:

1. The Position Ki and Kp values from the RelayId block will typically have Ki greater than Kp; this kind of ratio helps minimize the positional “bounce back” when the actuator moves in to the

ForceLimit zone. It is also recommended to move into the ForceLimit zone slowly, to minimize the effects of the SHA's momentum when reaching the ForceLimit, and to limit the Force overshoot in this case. As a consequence of these PID ratios, the high-speed positional moves may have some rabbiting. If the positional Ki value is instead kept less than the Kp, say 1/3 to 1/10 of the Kp, then the high-speed moves will be smoother, but there can be more bounce-back when moving into the Force zone.

2. For systems using the SHA_PositionControl_TL block, generally keep the Ki larger than the Kp value.
3. For systems using the SHA_PositionControl block, ignore the Ki value from the position RelayId block, and instead use a Ki value that is 1X to 3X the Position Kp value.
4. Using de-tuned values that give a smoother point-to-point motion may be too soft for optimal force control.

The Ki, Kp and output values are held and remain displayed after the block is disabled. When the block is re-enabled, it will take 10 cycles before the new data starts to display, because of the averaging.

When the position RelayId block is disabled, make sure there is logic to set the SHA Axis.VelocityTrim back to 0.0 to stop the axis motion. See the sample ladder in section 8.8.

10.3 SHA_RelayId For Extend Force Tuning

The sample project includes ladder logic with instances of the SHA_RelayId setup for Extend and Retract Force tuning. The Extend Force tuning will be covered first, and an example is shown below in section 10.3.1. Tuning Force using the RelayId block is recommended when the SHA_PositionControl_TL block is being used. If the _TL version is NOT being used, then either use this section but with the Aggressive set to 0.05 or else use the Force Tuning instructions in Section 10.3 below.
The SHA_PositionControl_TL block and related ladder logic for it are also required to be in place. However, the SHA_PositionControl_TL block must be disabled when the SHA_RelayId Force block is active. Also, if SHA_RelayId blocks for both Position and Force are present, only one of these RelayId blocks can be active at a time.

The intent of the Force Tuning block is to command the SHA back and forth between two Force values, and measure the observed response, to provide Force Ki and Kp values.

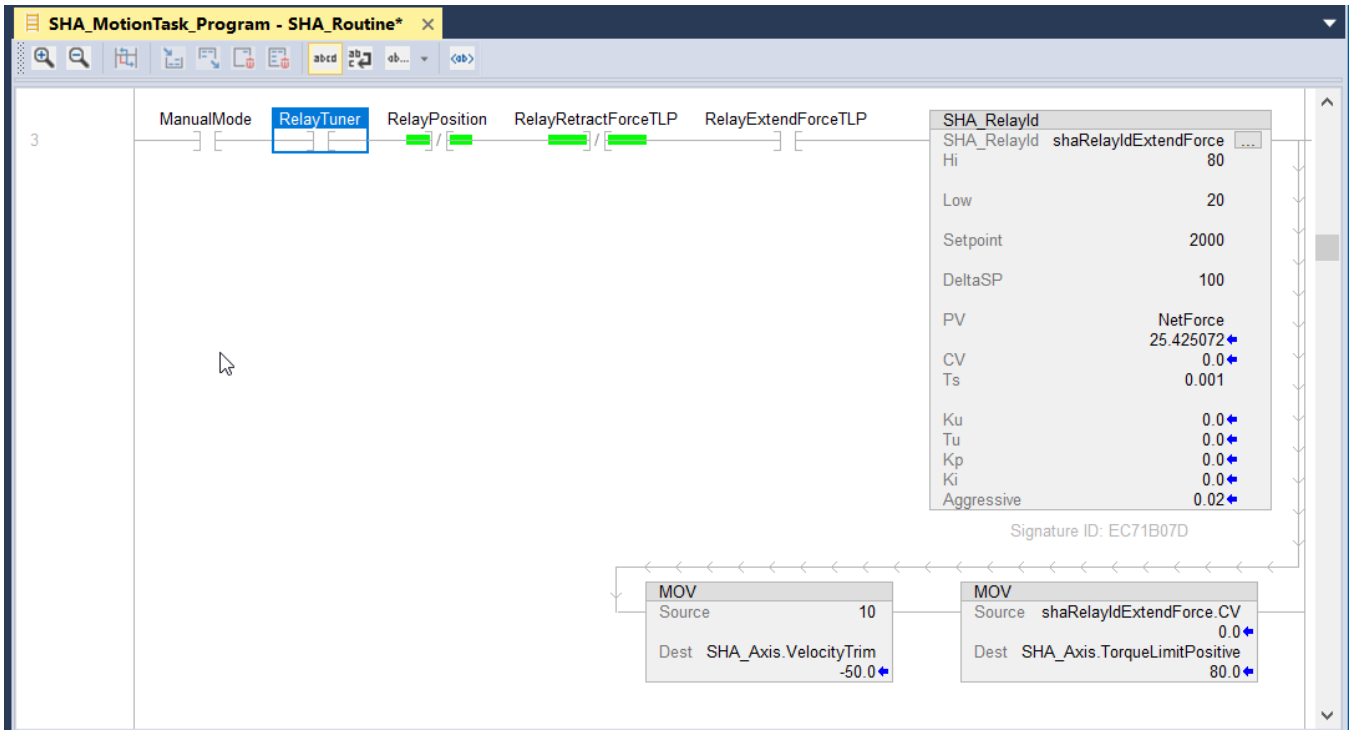
Notes:

1. Running the RelayId block can cause the entire SHA to shake and move; make sure it is securely mounted or clamped prior to enabling the block!
2. Prior to running the Force block, the pressure feedback MUST be properly calibrated. Pressure feedback values should always be positive, for both Extend and Retract pressure sensors. The

Force output of the SHA_PositionControl_TL will typically be negative when a Retract Force zone is active.

3. The Extend Force calibration can be performed using the SHA's internal "end stop", the physical extend position limit (or retract limit, if calibrating for a Retract Force Limit). In this case, make sure the fully extend position can be reached, without damaging any other part of the fixture.
4. If a fixture will be used to provide the Force Limit, make sure to use a value for the Force Setpoint (+ DeltaSP) that is within the limits of the fixture.

10.3.1 Extend Force SHA_RelayId Parameters



If your SHA has a lower rating than a 2000 lbf, then use a Force Setpoint that is below the limit of your SHA!
The Kp and Ki output values will need to be entered into the SHA_Config's ForceKp and ForceKi tags for this SHA, and then tested using force limited moves with the SHA_PositionControl_TL block.

- Hi – Represents the highest TorqueLimitPositive value to use, when moving to the extend Setpoint force + the DeltaSP force offset value.
- Low – Represents the lowest TorqueLimitPositive value to use, when changing to the Setpoint - DeltaSp force value.
- Setpoint – Represents the starting force to use. From this starting Force, the force will be dithered by the DeltaSP offset force. In the example, the actuator will extend (because of the MOV of 5 to the SHA_Axis.VelocityTrim) until a Force of 2000 lbs is reached.

- DeltaSP – Represents the Force offset value to add to and subtract from the setpoint value, to measure the response. In the example above, this is 100 lbs.
- PV – Represents the Process Variable, so in a Force use case, this is the tag associated with the Force output of the SHA_PositionControl_TL block.
- CV – Represents the Control Variable. In the Force case, this will be the Torque % value applied to the Axis.TorqueLimitPositive value (or the Axis.TorqueLimitNegative value if tuning a retract force). NOTE: if the Hi and Lo specified are not sufficient to reach the desired range of forces, the Axis.TorqueLimitPositive value may "stick" at one extreme or the other. In this case, either adjust the Setpoint force (provided it is within the SHA limits), or stop and adjust the Hi or Low value to allow the specified Force dithering.
- Ts – Represents the RPI time being used for the MotionTask, in seconds. The default value of 0.001 represents 1 millisecond. Adjust to match the SHA_Config.Ts value and RPI being used.
- Ku – Represents the amplitude of the force ripple observed when the RelayId block runs. The value should stay consistent, if the feedback is being provided at a constant rate.
- Tu – Represents the observed time-period of the force oscillations. The value should be fairly steady, if the feedback is being provided at a constant rate.
- Kp – Represents the force Kp value calculated by the RelayId block, for the given Aggressive value. This value is the average of the last 10 oscillations.
- Ki – Represents the force Ki value calculated by the RelayId block, for the given Aggressive value, and is the average from the last 10 oscillations.
- Aggressive – Represents a percentage of the "hottest" tuning values to use when calculating the Ki and Kp. So a value of 0.02 represents PID values that are 2% of the "hot" values (the values that would be used if Aggressive = 1.0). Depending on the slowness of the RPI being used and the speed of the feedback data, a force Aggressive setting of 0.02 is a good starting point. Because the RelayId block can tune both Position and Force, the Aggressive value should be kept between 0.02 and 0.04 for the TL Force tuning case. Put the Kp and Ki values into the Force members of the SHA_Config, and then try some force limited moves using the SHA_PositionControl to observe the results. With most systems, an Aggressive of 0.02 is sufficient for Force tuning.
  If tuning a non-TL SHA_PositionControl block, use a starting Aggressive value of 0.05, and then also limit the Force Ki, by using a Ki value that is 1x to 6x the reported Force Kp value. Also review the tuning recommendations in Sectio 10.3, to confirm the Force Kp and Ki values for a the non-TL SHA block.

Notes:

The Force Ki and Kp values from the RelayId block will typically be very small; often both are less than 1. It is also recommended to move into the ForceLimit zone slowly, to minimize the effects of the SHA's momentum when reaching the ForceLimit, and to limit the Force overshoot in this case. As a consequence of these PID ratios, the high-speed positional moves may have some rabbitting. If the positional Ki value is instead kept less than the Kp, say 1/3 to 1/10 of the Kp, then the high-speed moves will be smoother, but there can be more bounce-back when moving into the Force zone.

The Ki, Kp and output values are held and displayed when the block is disabled. When the block is re-enabled, it will take 10 cycles before the new data starts to display, because of the averaging done in the RelayId block.

When the Force RelayId block is disabled, make sure there is logic to set the SHA Axis.VelocityTrim back to 0.0 to stop the axis motion. See the sample ladder in section 8.8.

10.4 Extend/Retract Force Tuning when using the SHA_PositionControl block

For the non-TL SHA block, an alternate method to Force tuning is as follows.

In this method, the SHA_PositionControl block will be used to push the actuator against the desired force endstop (extend or retract end stop), using the SHA_PositionControl block itself to observe the results.

Notes:

- Read through this entire section prior to starting the actual tuning, to understand the overall methods.
- Make sure the Position Tuning is already complete and the Position Kp and Ki values are in the SHA_Config.
- Make sure the desired Force Limit is enabled in the SHA_PositionControl block: EnableForceMaxLimit or EnableForceMinLimit. Set the EnableFeedforwards to false.
- Make sure the ForceMax (for Extend Force tuning) or the ForceMin (for Retract Force tuning) is set to a reasonable value: either the desired force limit for your actuator, or a value around 50% to 80% of it.
- Start with the ForceKi value in the SHA_Config set to 0.0.

The Force tuning method is then to set the Force Kp value, and then raise it until the system goes unstable. Note: on large SHAs, making the system unstable may cause a serious level of shaking, so make sure the system is clamped, strapped, or otherwise securely held in place, and be ready to quickly back down the Force Kp value! Also in this case, raise the Kp in smaller steps, so likely there will be some indication that the system is starting to get unstable, to avoid going into or much past the unstable value. Start with a Force Kp value of around 0.001

With the starting Kp value in place, and the ForceLimit enabled, and a proper ForceMax/ForceMin setting in place, and with the position tuning values in place, use the MAM on the virtual position axis to move the SHA near to the desired positon: nearly extended for the Extend Force tuning, or nearly fully Retracted for Retract Force tuning. From there drop the MAM speed lower: 0.04 inch/sec or so, and then command a position a quarter to half an inch past, which will put the SHA into the Force Limit zone.

Now, start to raise the Force Kp value, looking for instability.

Once the unstable Force Kp value is approached, then back down to a Force Kp that is 0.2 to 0.5X that value.

The Force Ki value will typically be in the range of 1X to 6X the Force Kp value. (In some small SHAs, the Ki may be as high as 8X the Force Kp value.)
Change the ForceMax for Extend Force (or ForceMin for Retract Force) to a value 10% lower, and see the response, and then set it back up to the original value. The ForceTrend can also be used to view the response.
Once the Force tuning is complete, use the MAM to return the SHA to a position that is off the respective end stop, back in range, out of the force limit zone.

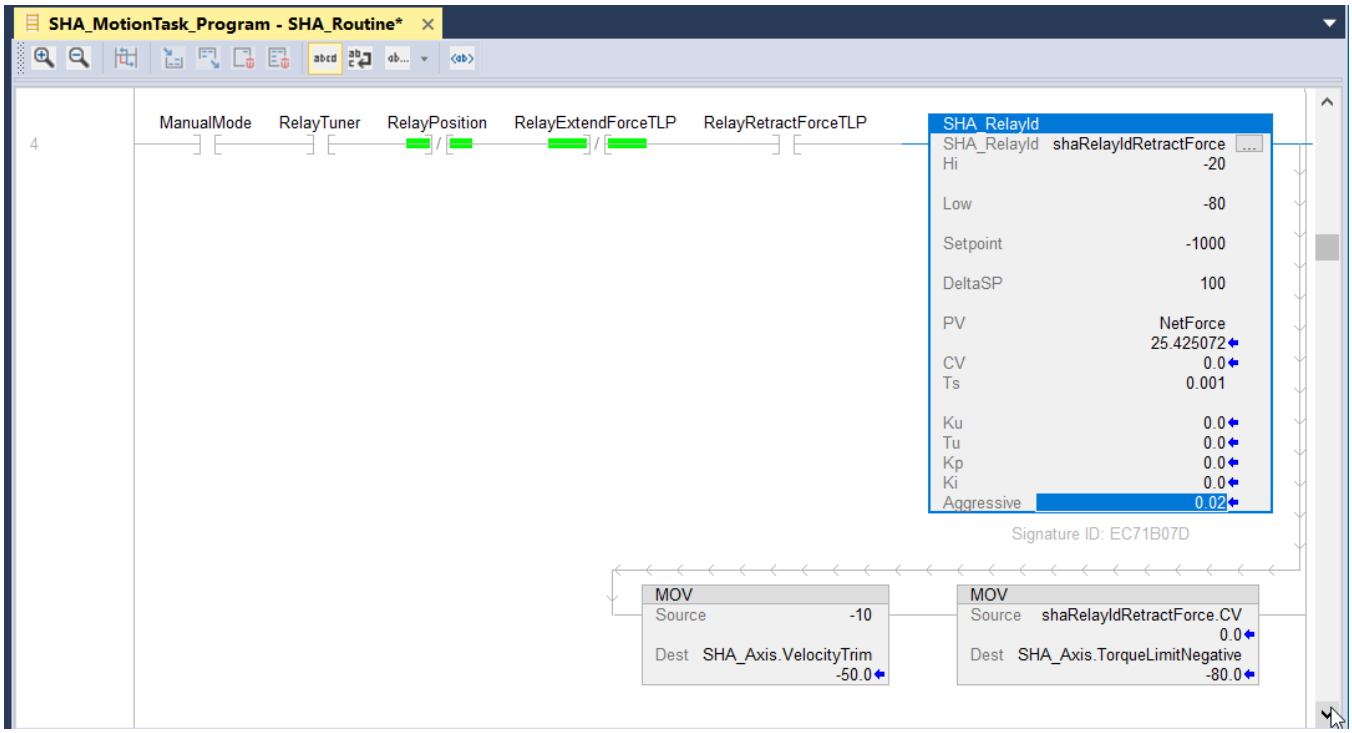
10.5 Retract Force SHA_RelayId Tuning

The Retract Force tuning example is shown below in section 10.3.1. The SHA_PositionControl block and related ladder logic for it are also required to be in place. However, the SHA_PositionControl block must be disabled when the SHA_RelayId Force block is active. Also, if SHA_RelayId blocks for both Position and Force are present, only one of these RelayId blocks can be active at a time.

The intent of the Force Tuning block is to command the SHA back and forth between two Force values, and measure the observed response, to provide Force Ki and Kp values.
Notes:

1. Running the RelayId block can cause the entire SHA to shake and move; make sure it is securely mounted or clamped prior to enabling the block!
2. Prior to running the Force block, the pressure feedback MUST be properly calibrated. Pressure feedback values should always be positive, for both Extend and Retract pressure sensors. The Force output of the SHA_PositionControl will typically be negative when a Retract Force zone is active, and the Force will report a positive value in the Extend Force zone.
3. The Retract Force calibration can be performed using the SHA's internal "end stop", the physical retract position limit. In this case, make sure the fully retracted position can be reached, without damaging any other part of the fixture.
4. If a fixture will be used to provide the Force Limit, make sure to use a value for the Force Setpoint (+ DeltaSP) that is within the limits of the fixture.

10.5.1 Retract Force SHA_RelayId Parameters



If your SHA has a Retract Force rating lower than a 1000 lbf, then use a Force Setpoint that is below the limit of your SHA! For a Retract Force RelayId tuning block, the (force) Setpoint should be entered as a negative value, as retract Force from the SHA_PositionControl block will be negative. Likewise, in the Retract Force case, the VelocityTrim will be negative (so the SHA retracts), and the RelayId.CV (Control Variable) needs to be MOV'd to the axis.TorqueLimitNegative.

The Kp and Ki output values will need to be entered into the SHA_Config ForceKp and ForceKi tags for this SHA, and then tested using force limited moves with the SHA_PositionControl block.

- Hi – Represents the most positive value, so in the Retract Force case, this is the negative Torque value that is closest to 0 (-20 in the example above).
- Low – Represents the lowest, most negative value. For a Retract Force tune, this is the -80 value in the example above.
- Setpoint – Represents the starting force to use. From this starting Force, the force will be dithered by the DeltaSP offset force. In the example, the actuator will retract (because of the MOV of -10 to the SHA_Axis.VelocityTrim) until a -1000 lbf is reached.
- DeltaSP – Represents the Force offset value to add to and subtract from the setpoint value, to measure the response. In the example, this is 100 lbs.
- PV – Represents the Process Variable, so in a Force use case, this is the tag associated with the Force output of the SHA_PositionControl block.

- CV – Represents the Control Variable. In the Force case, this will be the Torque % value applied to the Axis.TorqueLimitNegative value for tuning a retract force. NOTE: if the Hi and Lo specified are not sufficient to reach the desired range of forces, the Axis.TorqueLimitNegative value may “stick” at one extreme or the other. In this case, either adjust the Setpoint force (provided it is within the SHA limits); or stop and adjust the Hi or Low value to allow the specified Force dithering.
- Ts – Represents the RPI time being used for the MotionTask, in seconds. The default value of 0.001 represents 1 millisecond. Adjust to match the SHA_Config.Ts value and RPI being used.
- Ku – Represents the amplitude of the force ripple observed when the RelayId block runs. The value should stay consistent, if the feedback is being provided at a constant rate.
- Tu – Represents the observed time-period of the force oscillations. The value should be fairly steady, if the feedback is being provided at a constant rate.
- Kp – Represents the force Kp value calculated by the RelayId block, for the given Aggressive value. This value is the average of the last 10 oscillations.
- Ki – Represents the force Ki value calculated by the RelayId block, for the given Aggressive value, and is the average from the last 10 oscillations.
- Aggressive – Represents a percentage of the “hottest” tuning values to use when calculating the Ki and Kp. So a value of 0.02 represents PID values that are 2% of the “hot” values (the values that would be used if Aggressive = 1.0). Depending on the slowness of the RPI being used and the speed of the feedback data, a force Aggressive setting of 0.02 is a good starting point. Because the RelayId block can tune both Position and Force, the Aggressive value should be kept between 0.02 and 0.04 for the Force tuning case. Put the Kp and Ki values into the Force members of the SHA_Config, and then try some force limited moves using the SHA_PositionControl to observe the results. With most systems, an Aggressive of 0.02 is sufficient for Force tuning.

Notes:

The Force Ki and Kp values from the RelayId block will typically be very small; often both are less than 1. It is also recommended to move into the ForceLimit zone slowly, to minimize the effects of the SHA’s momentum when reaching the ForceLimit, and to limit the Force overshoot in this case. As a consequence of these PID ratios, the high-speed positional moves may have some rabbitting. If the positional Ki value is instead kept less than the Kp, say 1/3 to 1/10 of the Kp, then the high-speed moves will be smoother, but there can be more bounce-back when moving into the Force zone.

The Ki, Kp and output values are held and displayed when the block is disabled. When the block is re-enabled, it will take 10 cycles before the new data starts to display, because of the averaging done in the RelayId block.

When the Force RelayId block is disabled, make sure there is logic to set the SHA Axis.VelocityTrim back to 0.0 to stop the axis motion. See the sample ladder in section 8.8.

11. SHA and AOI Programming Tasks

11.1 Commanding Position and Force

The AOI will use the PI tuning values when attempting to reach the specified command position, and/or any Force Min or Max Limit. If the AOI reaches the commanded position, then it will stop there and actively hold that position. Likewise, if a Force limit is reached, then the AOI will adjust the position as needed to maintain the Force limit (and provided the commanded position is not reached).

For positional applications, the Force feedback (if present) should still be monitored, and an upper force limit should typically be enforced (for example, the quoted Maximum Force of the actuator). Or, if a pressure sensor is present, then the appropriate Force Min/Max value can be set and the EnableForceMax (or Min)Limit can be enabled, too, so that during positional moves, the Force Limit of the actuator is respected. Defensive logic can be included to monitor the force, and if the force stays 10% or more higher than the expected max/min force for more than 1 second, then an error has occurred, and the force should be released.

For Force applications, the target force is applied by using “position with force limiting” logic. The Force Max (or Min) value is set to the desired target force, and the EnableForceMax (or Min) Limit is then enabled. Now, a position slightly beyond the expected force zone is commanded, so that the desired force is then applied as the actuator tries to reach that position. The reaction of the Force PI gains is larger/faster as the positional error grows. A position target ~0.1 to 0.25” or 4 to 6 mm beyond the expected force zone is a typical value. Smaller positional errors will take longer to ramp up the force. Using a very large positional error will then require more time to retract, as it will take longer for the virtual axis command position to “catch up” to the actual position to then start moving the actuator.

11.2 In Position and In Force Tolerance and Time Windows

It is up to the control logic to determine when the AOI has reached the commanded position or force. This is usually done with a Position or Force Tolerance, which is a distance or force range, applied as a +- value around the setpoint. Once the value is within the tolerance, then typically an in position or force Window is used, which is a time value that the position or force must remain inside the tolerance range to consider the Step to be complete.

11.3 Notes on EnableForceLimits

If the EnableForceMaxLimit is disabled, then the actuator will use up to its maximum force when attempting to reach the Command/Reference position. It's a good idea to have logic which always monitors for a maximum force value, to stop the process with an error if it is reached.

The EnableFeedforwards allows for VelocityFeedforwards to be used when high-speed approach or retract positional moves will be run. When the actuator goes into the Force zone, then often the Feedforward is disabled to prevent the Force control from being overly reactive (if the feedforwards are left enabled, then if there is a high impact force, the actuator is more likely to back away from the force zone as a result).

12. SHA and Force Limit/Control Application Notes

This section covers details and considerations when using the SHA for Position Control with Force Limits.

12.1 Impact Force and Force Overshoot

When a force limit or force limit zone is being used, it is often desirable to limit the initial impact force, to minimize the force overshoot when the force increases. The actuator has momentum from the motor, the pump, the fluid and the rod. To minimize the impact force/force overshoot, the SHA velocity should be kept slow, within reason, based on the process, when entering the force zone. In addition, the ForceMax value could be set lower for initial contact, and then be raised (via a virtual axis, as shown in the Kyntronics SalesDemo Studio project) in a controlled manner to the desired force setpoint. It is also possible to set and adjust the TorquePositiveLimit (for an Extend force) or the TorqueNegativeLimit (for a Retract force) on the SHA Axis, to control the maximum force and the force ramp-up. Likewise, a virtual axis or rated-limited variable could be used to increase the respective Torque limit after contact.

To help determine the Torque required to reach the desired Force value, set the related TorquePositive or NegativeLimit to a low value, say 8%. Then run the SHA sequence and view the reported Force. Then slowly increase the Torque limit until the desired ForceMax (or ForceMin for Retract force limit) can be reached. Let's say the observed Torque required to reach a 4000 lbf target is 48%. The SHA_Config.TorqueMax (for extend force) could then be set to 50% (to allow a little overhead), which will help limit the impact force. In this case, the TorqueMax could also be set to a lower value, and then raised in a controlled manner when the Force Limit zone is reached.

In practice, the goal is often to minimize the cycle time. Therefore, it may be simplest to set the TorqueMax value to 50% in this example, and then setup a ramp on the ForceMax that is triggered when contact is detected (i.e.– the initial ForceMax value, or some percentage of, it is reached). Finally, adjust the velocity into the force zone, to see how fast is practical for the application. Note that the EnableFeedforward can be set False just prior to starting the move into the Force zone, as this helps limit the "bounce back" of position effect when/if the force ramps up quickly.

12.2 Extend Position with Force Limit

For SHAs using the Extend Position with Force limit case, the SHA_PositionControl ForceMax value contains a positive number with the Force limit. The SHA_PositionControl EnableForceMaxLimit needs to be set True, typically when entering the ForceLimitZone; or it can be set True all the time. The SHA_Config.TorqueMax, a positive value, is used for the Axis.TorqueLimitPositive value when building up extend Force. The SHA_PositionControl EnableFeedforward can be set False, if desired, to help limit any positional "bounce back" effect when/if the ForceMax value is exceeded.

12.3 Retract Position with Force Limit

For SHAs using the Retract Position with Force limit case, the SHA_PositionControl ForceMin value contains a negative number with the Force limit. The SHA_PositionControl EnableForceMinLimit needs to be set True, typically when entering the ForceLimitZone; or it can be set True all the time. The SHA_Config.TorqueMin, a negative value, is used for the Axis.TorqueLimitNegative value when building up retract Force. The SHA_PositionControl EnableFeedforward can be set False, if desired, to help limit any positional "bounce back" effect when/if the ForceMin value is exceeded.

12.4 Position Control and Force Limit

Typically the Force Limit zone has a finite maximum position, so the SHA_PositionControl block's PositionReference is driven to a position slightly (0.1 to .2") beyond the zone, so that the Force continues to be applied for as long as needed.

Note that when the Force portion of the cycle is complete, the SHA_PositionControl's PositionReference will need to be reversed, until it is clear of the Force zone. That commanded position "overshoot" (of 0.1 to 0.2") will need to be unwound, before the actuator will actually start to move the other way.

While it is possible to command a position well beyond the Force zone (say 1"); it would then take that much longer to unwind that extra inch; or, the Virtual axis' CommandPosition would need to be MAM'd to the SHA's actual position to then start the move out of the Force Zone immediately, without any unwind.

12.5 High Speed/High Force and Control Valves

On units with combination High Speed/High Force, control valves will be present to switch between these modes. The control valves typically take 20 to 50 milli-seconds to respond. Make sure to adjust the max speeds of the MAM motion blocks to match the mode.

12.6 Using Lock Valves to Hold Force or Position

If the position/force needs to be held for a "long" time, then it is possible to use the Lock Valve to hold the setpoint. Once the Lock Valve has been engaged (subject to the 20 to 50 milliseconds typical time), the AOI could be disabled, or the CommandPumpSpeed could be withheld from the VelocityTrim of the axis. The process then would need to monitor the force/position feedback, with

a percentage deadband window. When the force/position reaches the deadband, then the command position should be set to the actual position, and the AOI enabled, and then the move (back) to the target position should be started, and let it run for ~ 290 milliseconds (to build up pressure/force so that when the Lock Valve is released, there is not a sudden drop in position/force), and then open the Lock valve, to bring the process back to the original setpoint.

13. SHA Error Conditions and Detection

13.1 Relief Valve Trip, Fault Detection

- Unless otherwise noted, the SHA typically has a relief valve which limits the (internal) pressure to ~3000 psi. If this limit is reached, the relief valve will open, causing the pressure to drop. When tripped, the motor will continue to run the pump, but the fluid is recirculating through the relief valve path. In this situation, the pump will continue to generate heat. Therefore, logic should be included to detect this condition and issue a fault to stop the process.
- For example, in position control, if the CommandPumpSpeed is non-zero but there is no movement (from the linear transducer feedback) for ~2 to 3 seconds, then a fault condition should be set and kept active for 10 seconds. With a Force control application, if no movement occurs for ~2X the process time (e.g.—if the process requires a 5 second force control to press a part, then the fault condition should trip should be set for 10 seconds).

13.1.1 Relief Valve Details

- The Kyntronics actuators have relief valves that are set to protect the actuator and machine. If the actuator runs into a hard-stop, the drive's current will be at maximum, the motor will spin but there will be no movement.
- Extended running into the relief valves can cause excessive heat and potential damage to the actuator.

13.1.2 Hardstops and Position Monitoring

Another way to trip the relief valve is to move the actuator into the retract or extend internal hard stop. In some cases, the press may intentionally command a position past an endstop in order to apply the desired force.

However, most actuators have absolute position feedback (or incremental feedback that can be homed). Take advantage of this, and add logic to monitor the actual position, and Stop and/or fault if the actual position ever encroaches to with a couple of millimeters of either hardstop position.

Note that this does require gentle motion into both endstops during commissioning to determine the minimum and maximum absolute positions.

Finally, it is possible for the absolute position sensor to be accidentally bumped hard enough to physically move/shift it, which could then allow one endstop to be reached without tripping this

detection logic. The actuator can be marked during installation, to later allow for a quick visual check for this situation.

13.2 Over-temperature Sensor, Thermal Trip

- The process should stop within 2 seconds if the pump thermostat (thermal switch) gives a thermal trip. There is a temperature hysteresis, and if the sensor trips, the over-temperature cause needs to be determined. The actuator must be kept off for a minimum of 5 minutes to allow the pump to cool.
- The sensor is wired to the same connector that supplies power to the (optional) fans on the pump housing cover.
- The thermal sensor has limited trip cycles.

13.3 Over Force Limit

Always monitor the Force output, and if it ever holds 10% (or more) above the quoted force of the unit for more than a second continuously, then error and stop the process. Need to allow for impact overshoot, which is typically less than 1 second.

13.4 Trends: Capture the Process Baseline, Compare for troubleshooting

When the process cycle logic is complete and running, capture a Trend of several cycles, logging the command and actual positions, the pressure and force, motor torque or current, and the CommandPumpSpeed at a minimum. Make sure to save a CSV version of the data.

Then, if a problem ever arises, capture a fresh Trend to compare with the baseline. This can be an invaluable troubleshooting tool.

14. Further Information

For more examples and information, visit Rockwell Automation Integration (kyntronics.com)

For assistance, email support@kyntronics.com

Chris Knaack (Sr. Controls Engineer, AOI Developer)

- cknaack@kyntronics.com
- Office: 855-596-8765

Scott Schmocker (Sr. Engineer, Software and Support)

- sschmocker@kyntronics.com
- Office: 855-596-8765
- Cell: 216.212.4734

Carl Richter (VP, GM)

- crichter@kyntronics.com
- Office: 855-596-8765
- Cell: 440-221-4269