

Teaching materials

Student's guide – Mechatronics Motion Platform

MISCE project

Mechatronics for Improving and Standardizing Competences in Engineering



Competence: Mechatronics

Workgroup: Universidade do Minho



© 2025 MISCE Consortium. Licensed under CC Attribution-ShareAlike 4.0 International
(<https://creativecommons.org/licenses/by-sa/4.0/>)

This document corresponds to the Students guide – Hardware and Software of the conveyor and rotating disk platform, for the competence ‘Mechatronics’.

Version: 1.0

Date: October 5th, 2023

Visit <https://misceproject.eu/> for more information.

Index of contents

1	Motion - Axis Configuration 1 - Hardware	4
1.1	Motion Control	4
1.2	EtherCAT Network	4
1.3	CPU BUS	6
1.4	System Setup - Drive Configuration.....	7
1.5	Axis Configuration	12
1.6	Start Motion Control (MC) Test Run.....	14
1.7	MC Commands	17
2	Motion - Axis Configuration 2 - Hardware	19
2.1	Axis Configuration – Axis1 [Conveyor]	19
2.2	Unit Conversion Settings	19
2.3	Operation Settings.....	21
2.4	Home Settings.....	21
2.5	Unit Conversion Settings	22
2.6	Operation Settings.....	23
2.7	Home Settings.....	24
2.8	Position Count Settings	25
3	Motion: Start-up – Software.....	28
3.1	Start-up Procedure	28
3.2	_sAXIS_REF Structure	31
3.3	Motion Basics	31
3.4	Task.....	32
3.5	MC_Power.....	33
3.6	MC_Power – Exercise 1	34
3.7	MC_Home – Exercise 2.....	34
3.8	MC_MoveVelocity – Exercise 3	34
4	Motion: Exercise - Software.....	36
4.1	Exercise	36
4.2	Exercise 1.....	37
4.3	Exercise 2 - Complete the Logic.....	38

Index of figures

Figure 1.	Multiview Explorer window	4
Figure 2.	Scan network in online mode	4
Figure 3.	Compare and Merge with Actual Network Configuration	5
Figure 4.	Left side shows the project data and the right shows the live network configuration.....	5
Figure 5.	Project network configuration	6
Figure 6.	CPU/Expansion Racks.....	6

Figure 7. Compare and Merge with Actual Unit Configuration	7
Figure 8. Servo drive node and Online mode selection	7
Figure 9. Setup and Tuning selection	8
Figure 10. Setup and Tuning selection window	8
Figure 11. Quick Parameter Setup and I/O Monitor	8
Figure 12. Quick Parameter Setup and I/O Monitor	9
Figure 13. Asking for restart the drive	9
Figure 14. Selecting input configuration	10
Figure 15. Output Configuration	11
Figure 16. Configuring the first servo drive and switch Node 1 to Offline	11
Figure 17. Axis definitions	12
Figure 18. Axis name definitions	13
Figure 19. Final configurations	14
Figure 20. Start MC Test Run	15
Figure 21. Start MC Test Run	15
Figure 22. Test Run Interface steps	16
Figure 23. Example configuration	17
Figure 24. Jogging	17
Figure 25. Absolute Positioning	18
Figure 26. Relative Positioning	18
Figure 27. Homing	18
Figure 28. Axis Settings	19
Figure 28. Unit Conversion Settings	20
Figure 30. Homing configurations	21
Figure 31. Homing configurations	22
Figure 32. Operation Settings	23
Figure 33. Homing settings – Axis 2	24
Figure 33. Homing settings – Axis 2	25
Figure 35. Linear mode	25
Figure 36. Linear mode example	26
Figure 36. Rotary mode	26
Figure 36. Rotary mode example	26
Figure 39. Position Count Settings	27
Figure 40. Global variables in <i>Programming > Data</i>	28
Figure 40. Working with local variables	29
Figure 42. Global variables	29
Figure 43. Global Variables to Program0	30
Figure 44. Global variables accessible from all POUs	30
Figure 45. <i>_sAXIS_REF</i> Structure	31
Figure 46. <i>Section0_MotionBasics</i>	31
Figure 47. <i>Accessing Task Settings and creating assignments</i>	32
Figure 48. <i>Tasks and priorities</i>	32
Figure 49. <i>Program Assignment Settings tab</i>	33
Figure 50. <i>Axis1 (Conveyor)</i>	36
Figure 51. <i>Axis 2 (Disk)</i>	36
Figure 52. <i>Exercise</i>	37



1 Motion - Axis Configuration 1 - Hardware

1.1 Motion Control

To control one or more axes from the NJ controller, it is necessary to associate the project with the hardware information present in the EtherCAT network (machine network).

To do so, when starting the programming of the Motion Control module, we must set up the system so that it knows what it is controlling.

1.2 EtherCAT Network

To obtain the EtherCAT network automatically:

- In the **Multiview Explorer** window, under **Configurations and Setup > EtherCAT**, double-click.

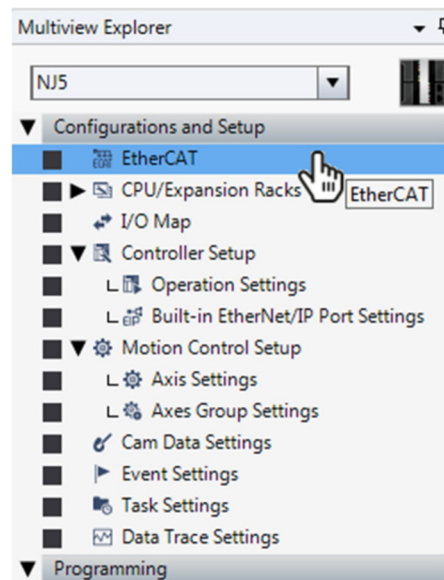


Figure 1. Multiview Explorer window

- The workspace will display the EtherCAT network diagram.
- Initially, only the NJ controller appears in the network. To scan the network, switch to **Online** mode.

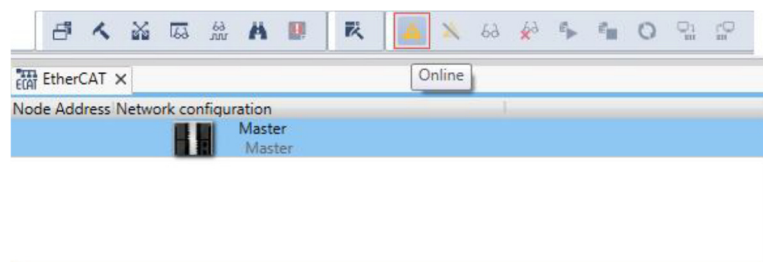


Figure 2. Scan network in online mode



- Right-click on the NJ controller and select **Compare and Merge with Actual Network Configuration**.

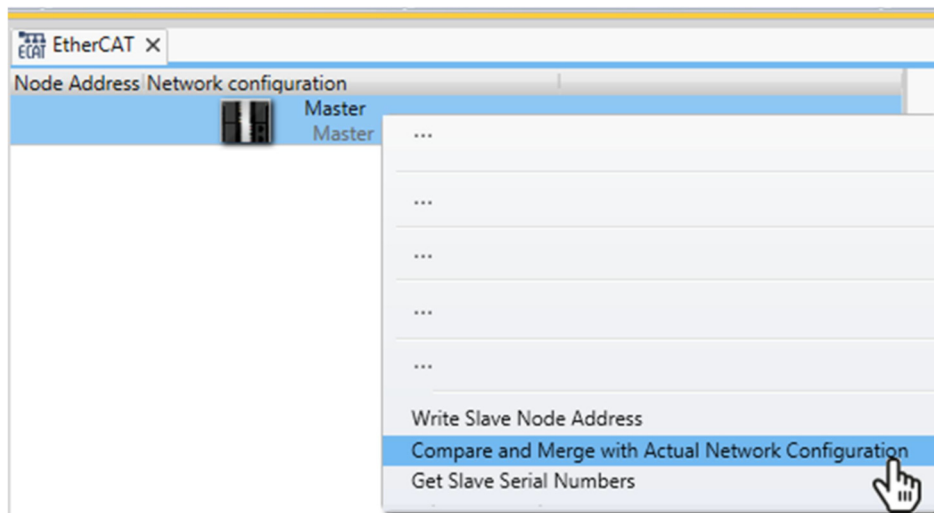


Figure 3. Compare and Merge with Actual Network Configuration

- The left side shows the project data, and the right shows the live network configuration.

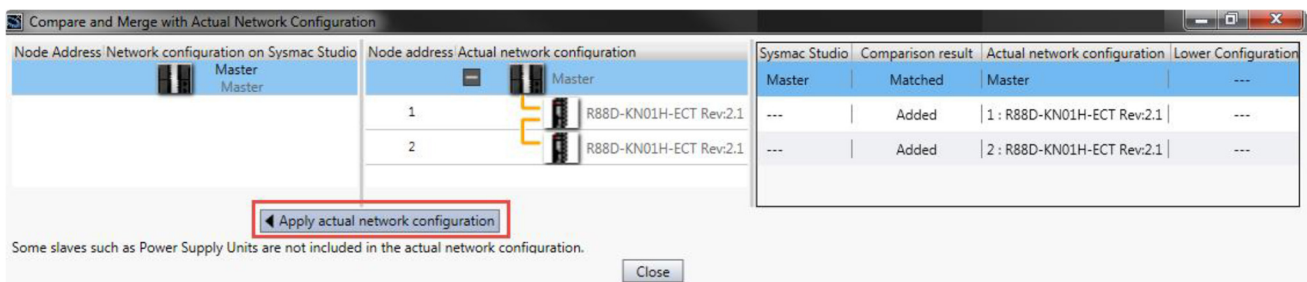
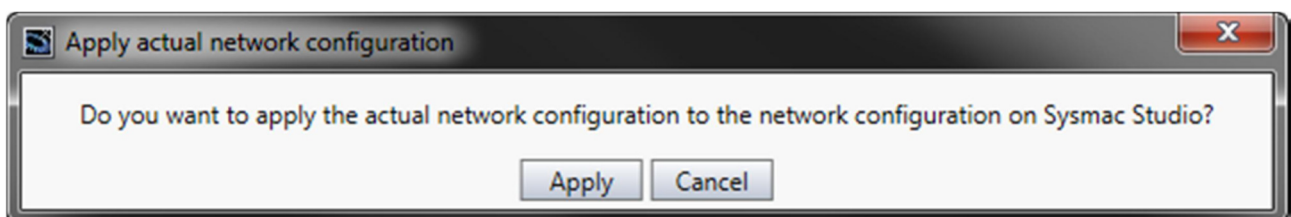


Figure 4. Left side shows the project data and the right shows the live network configuration

- Click **Apply actual network configuration** to incorporate the network into the project.



- Confirm when prompted and click **Close** to finalize.

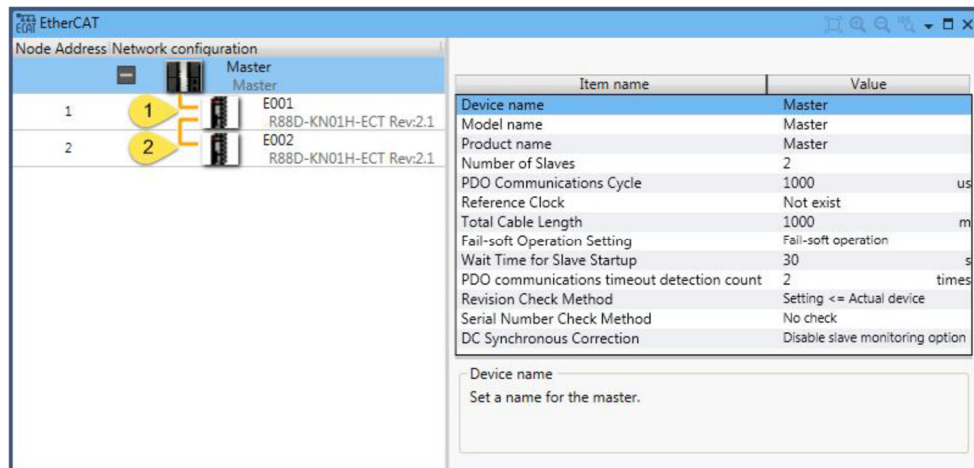


Figure 5. Project network configuration

1.3 CPU BUS

To configure the CPU/bus network, use the same approach.

- Go to **Configurations and Setup > CPU/Expansion Racks**.

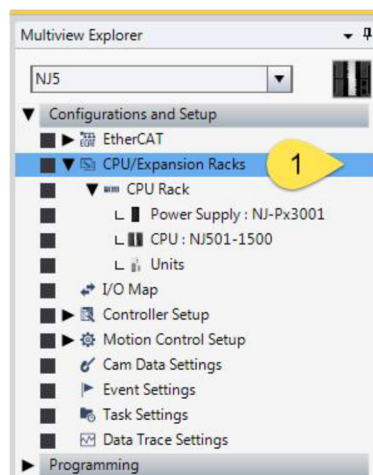


Figure 6. CPU/Expansion Racks

- If there are modules on the bus, right-click and select **Compare and Merge with Actual Unit Configuration** to apply it to the project.

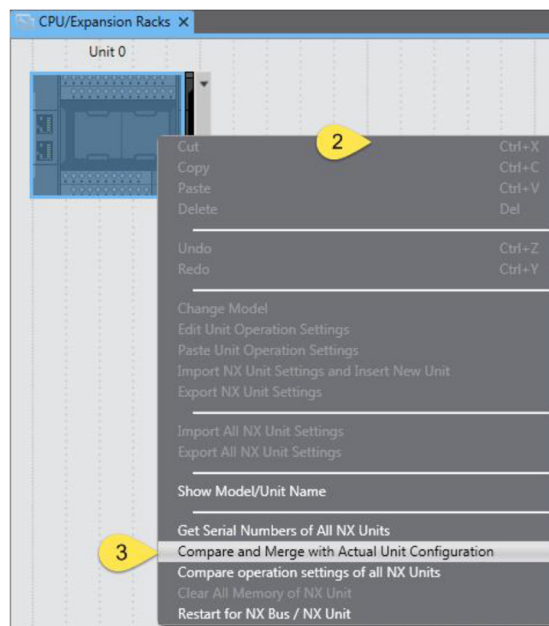


Figure 7. Compare and Merge with Actual Unit Configuration

1.4 System Setup - Drive Configuration

- Based on the updated project, click: **Project > Rebuild Controller > Synchronize > Transfer to Controller**.
- Right-click on the corresponding servo drive node and select **Online**.

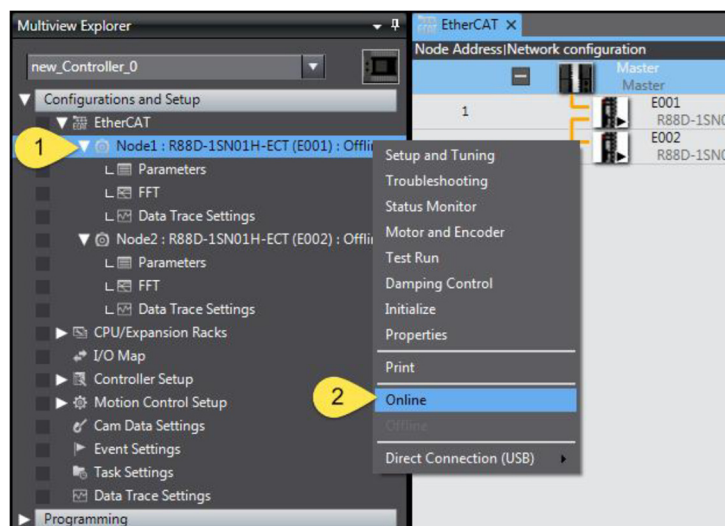


Figure 8. Servo drive node and Online mode selection

- Then, right-click **Node 1 > Parameters** and choose **Setup and Tuning**.

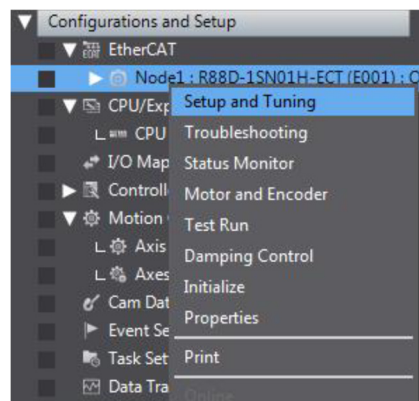


Figure 9. Setup and Tuning selection

- A configuration and tuning window opens.

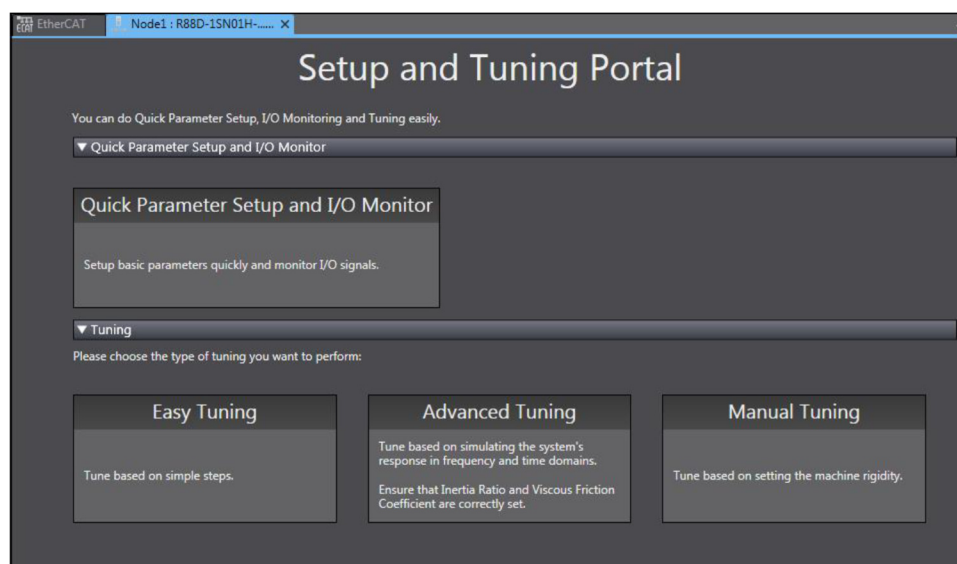


Figure 10. Setup and Tuning selection window

- Click **Quick Parameter Setup and I/O Monitor** to open the Sysmac Studio wizard.

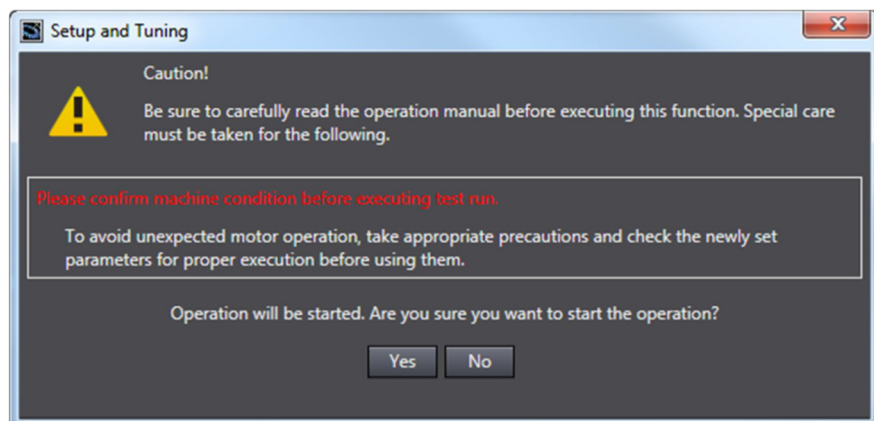


Figure 11. Quick Parameter Setup and I/O Monitor



- A warning appears indicating that the configuration might affect machine behavior.
- Confirm to proceed.

Step 1: Motor and Encoder Configuration

- The window will display motor parameters.

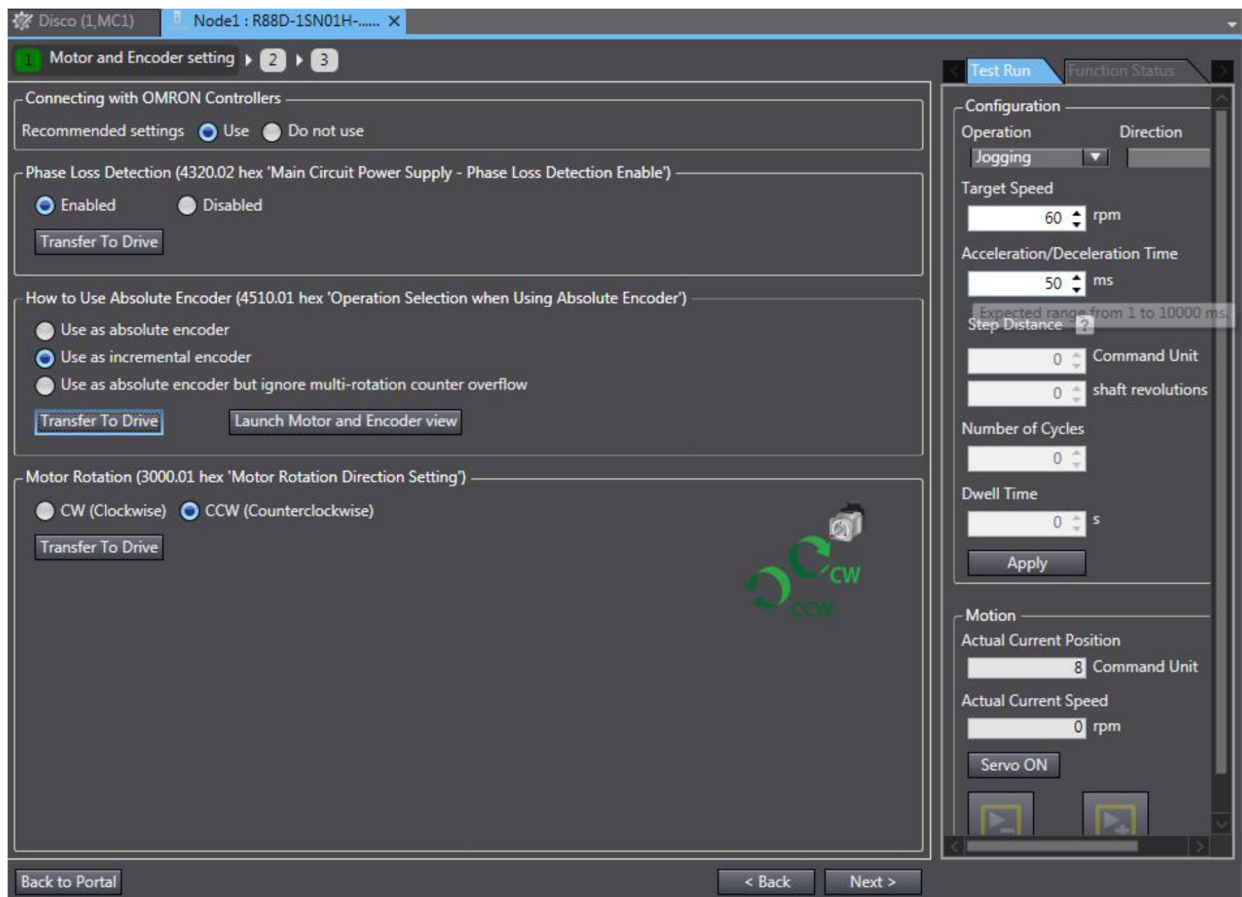


Figure 12. Quick Parameter Setup and I/O Monitor

- Confirm each setting and click **Transfer to Drive**.
- After transferring, Sysmac Studio will prompt to restart the drive. Click **Yes**.

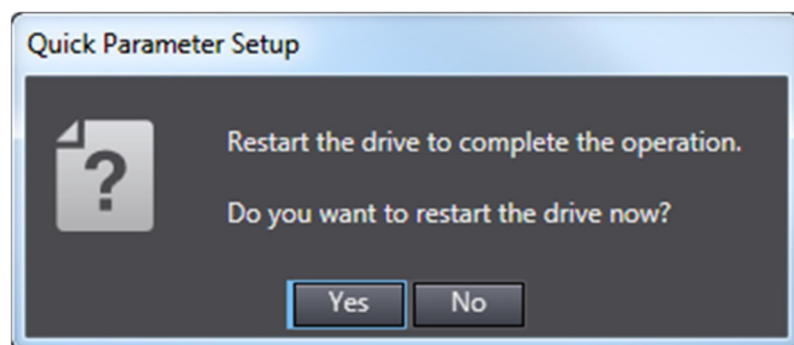


Figure 13. Asking for restart the drive



Step 2: Input Configuration

- Select input configuration and click **Transfer to Drive**.

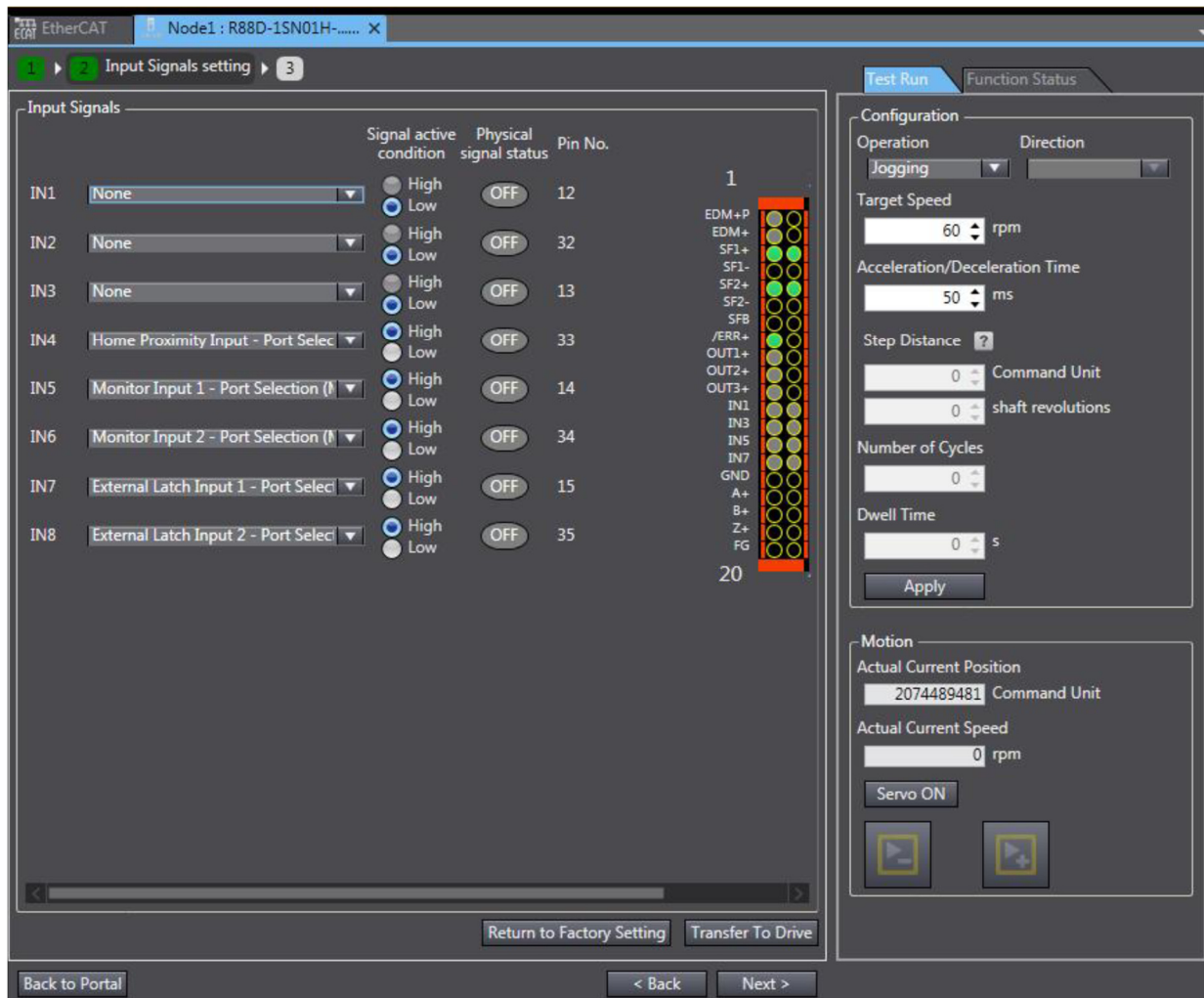


Figure 14. Selecting input configuration

- Then click **Next**.

Step 3: Output Configuration

- Select output configuration, click **Transfer to Drive**, then click **Finish**.

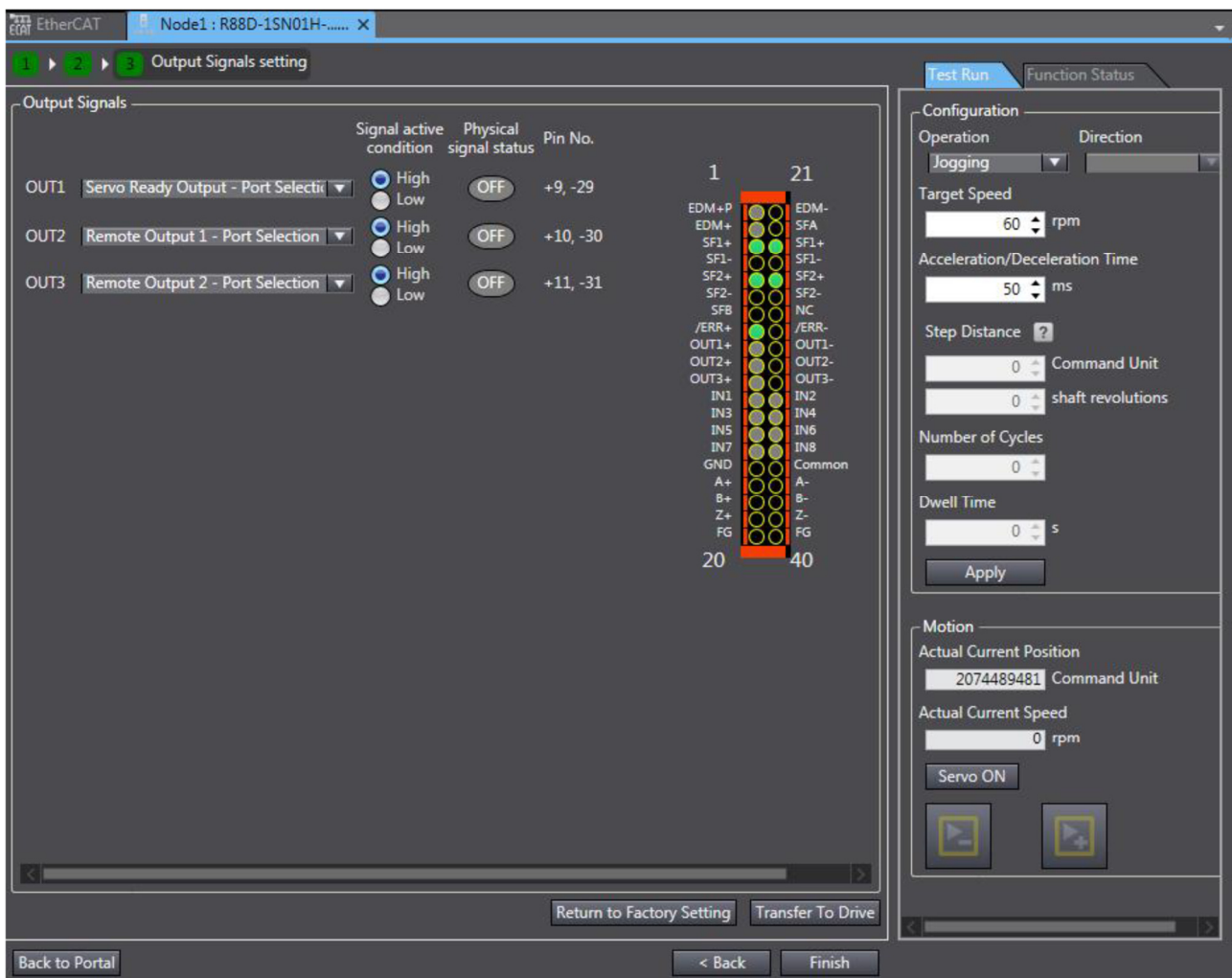


Figure 15. Output Configuration

- After configuring the first servo drive, switch Node 1 to **Offline**.

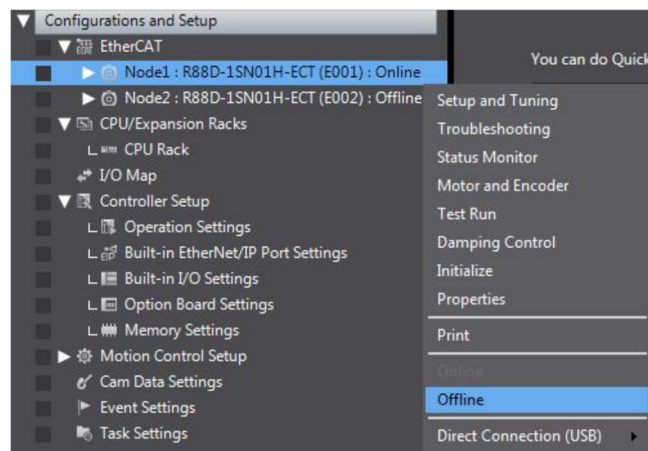


Figure 16. Configuring the first servo drive and switch Node 1 to Offline

- Repeat the process for the second node.

1.5 Axis Configuration

After validating the EtherCAT network and configuring the drives, the axes can be defined.

- Go to **Configurations and Setup > Motion Control Setup > Axis Settings**.
- Right-click and select **Add > Axis Settings** twice (for two axes).

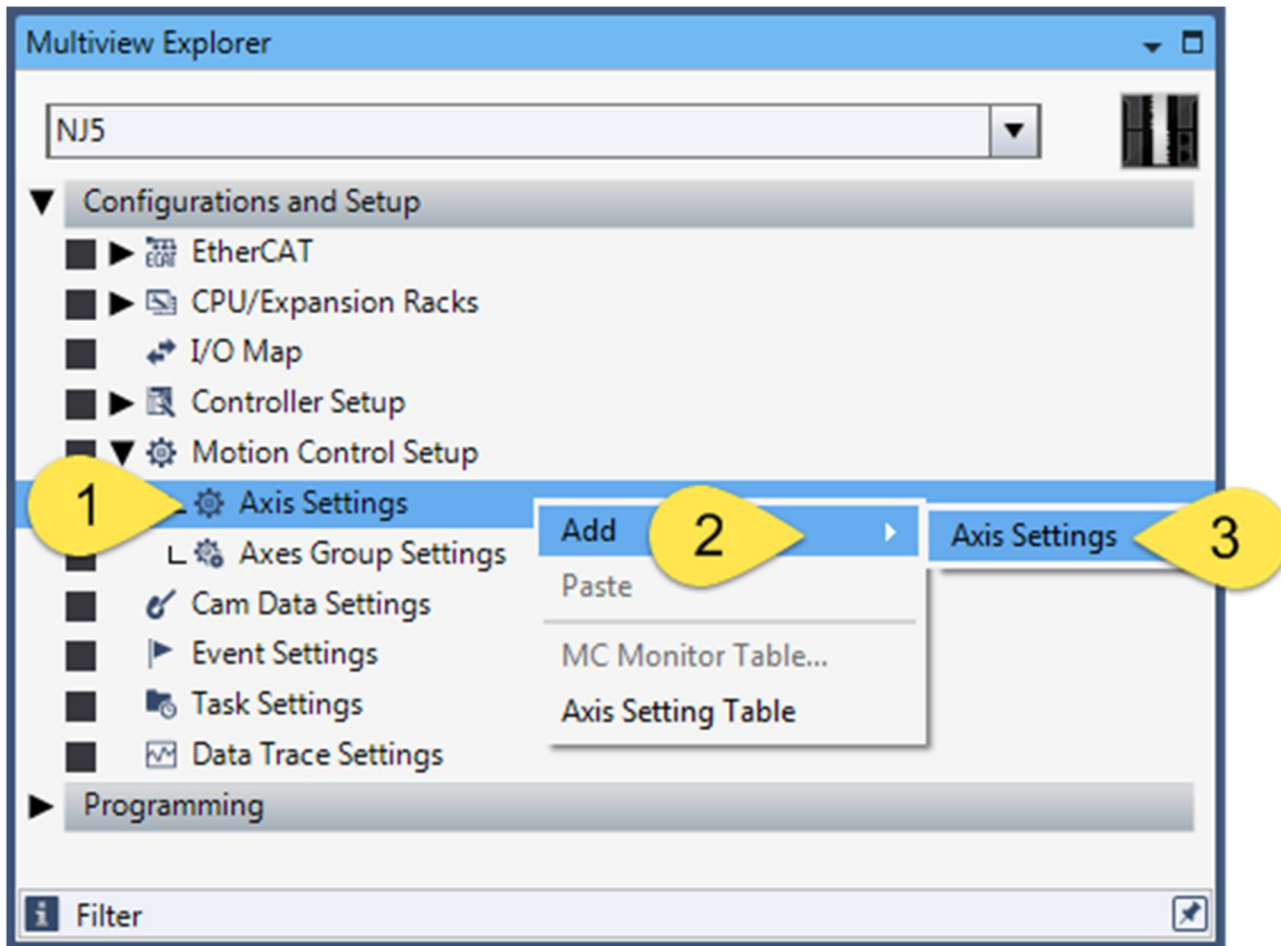


Figure 17. Axis definitions

- Example names:
 - MC_Axis000 (0) → renamed to Axis1 → Conveyor
 - MC_Axis001 (1) → renamed to Axis2 → Disk

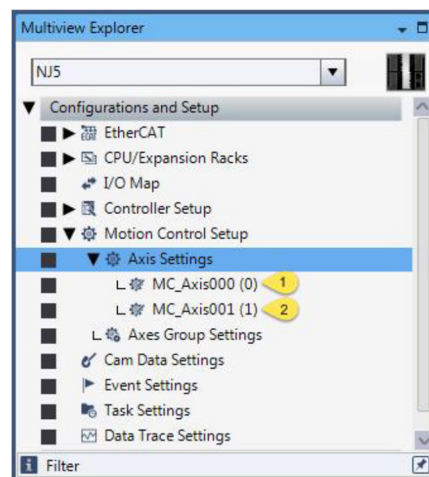
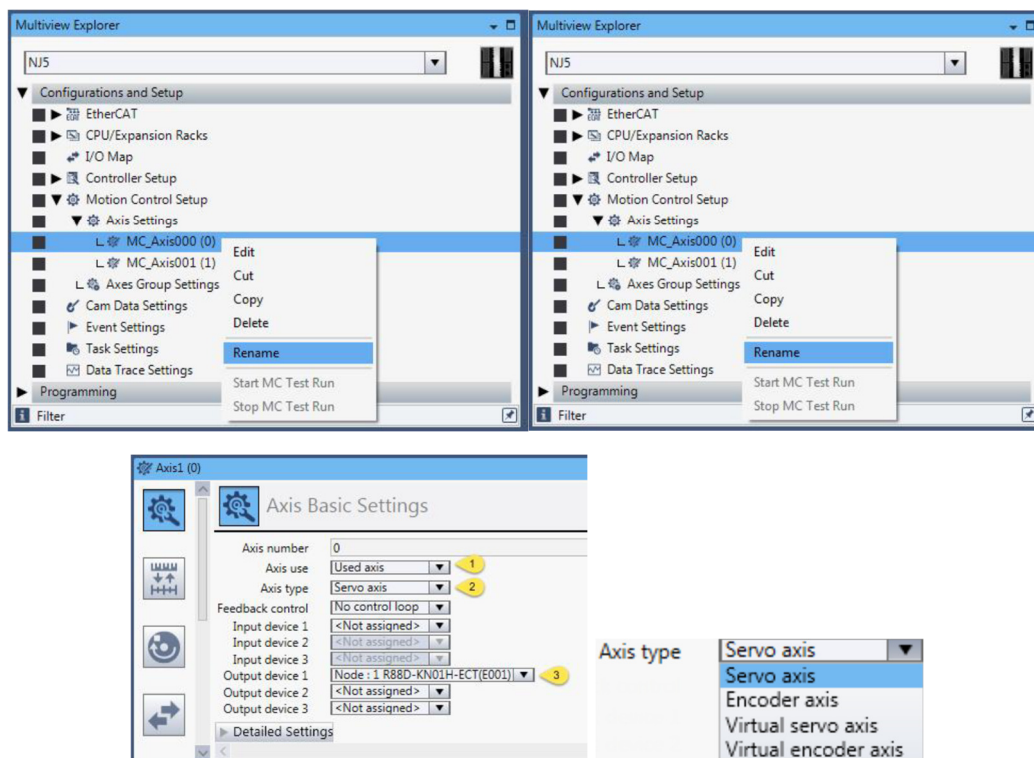


Figure 18. Axis name definitions

- Right-click on each axis to rename them appropriately.
- Double-click **Axis1** and assign **Node 1** as the output device:
 - Axis Use: Used Axis
 - Axis Type: Servo Axis
 - Output Device: Node 1 R88D-KN01H-ECT (E001)
- Repeat for **Axis2** using **Node 2**.
- Then execute:
 - **Project > Rebuild Controller > Online > Synchronize > Transfer to Controller**



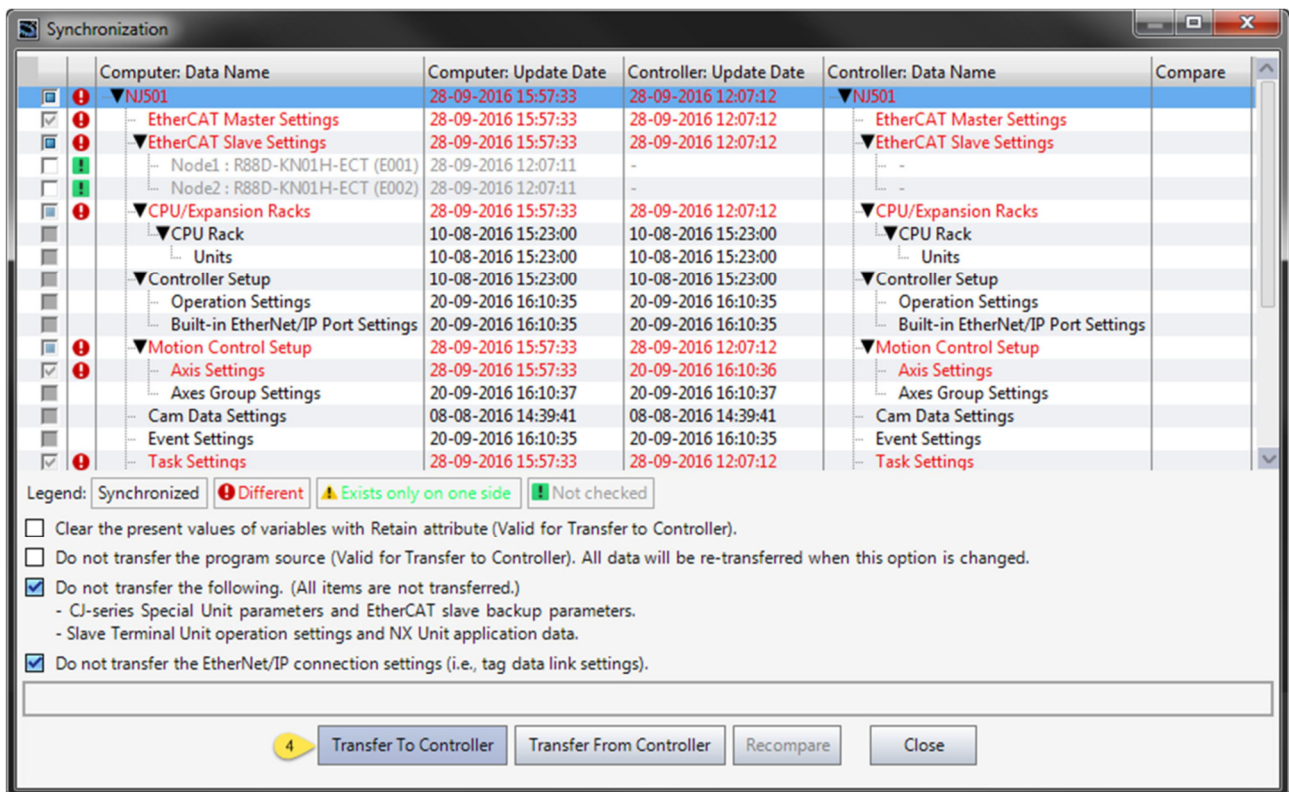
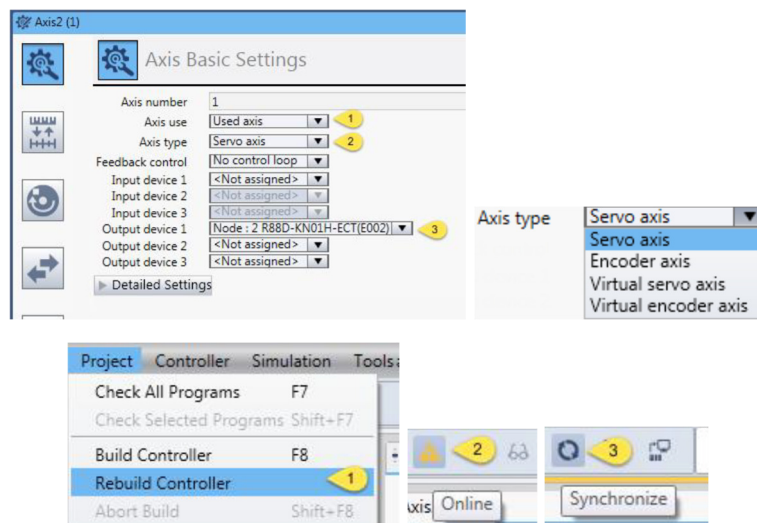


Figure 19. Final configurations

1.6 Start Motion Control (MC) Test Run

- Right-click on **Axis1 (0)** > **Start MC Test Run**

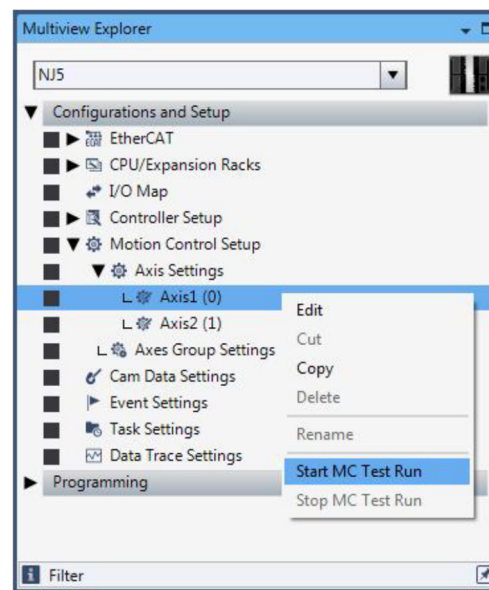


Figure 20. Start MC Test Run

- A warning will appear because the servomotor will rotate. Take precautions and confirm.

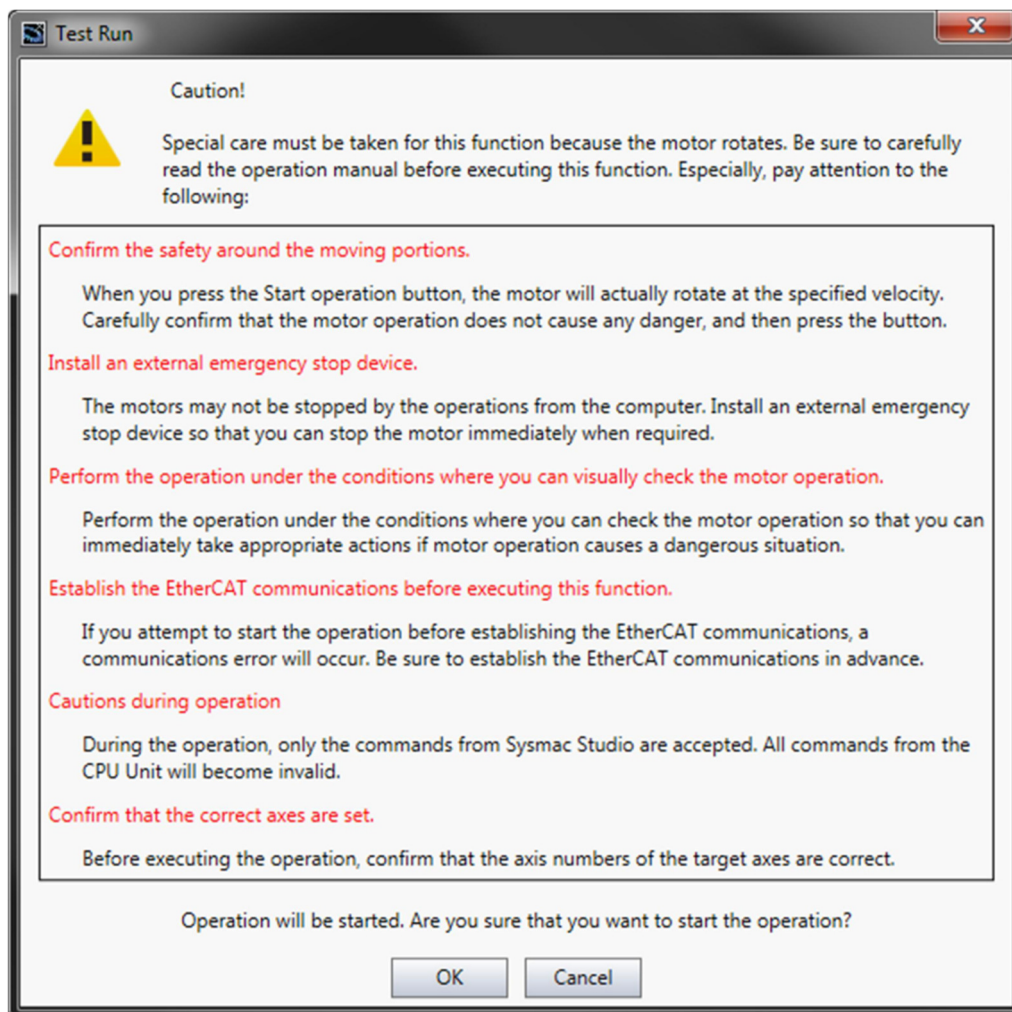


Figure 21. Start MC Test Run



Test Run Interface includes:

1. Axis selection
2. Axis state and error list
3. Axis ON/OFF
4. MC tabs: Jogging, Absolute Positioning, Relative Positioning, Homing
5. Speed, distance, acceleration, etc.
6. Execute MC commands

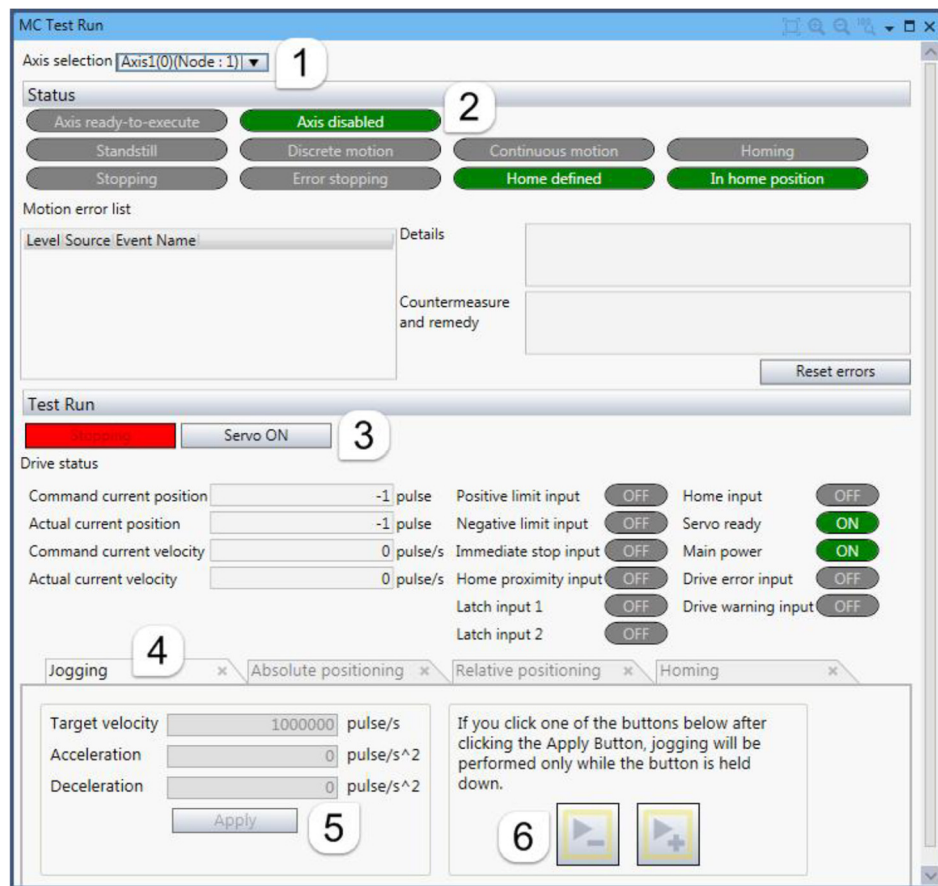
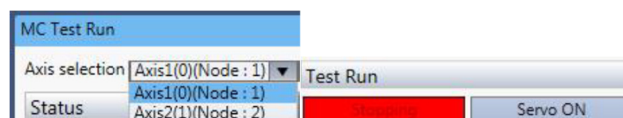


Figure 22. Test Run Interface steps

Example:

- Set the axis to ON. The button will switch to OFF.
- Under **Jogging**, click **Apply**.
- Click **MC Jogging +** → motor rotates counterclockwise (CCW), position increases.
- Click **MC Jogging -** → motor rotates clockwise (CW), position decreases.





The screenshot displays the software interface for the Mechatronics Motion Platform. At the top, there is a 'Test Run' section with a red 'Stopping' button and a blue 'Servo OFF' button. Below this is the 'Jogging' section, which includes input fields for 'Target velocity' (1000000 pulse/s), 'Acceleration' (0 pulse/s^2), and 'Deceleration' (0 pulse/s^2), along with an 'Apply' button. To the right of the 'Jogging' section is a text box explaining that jogging will be performed only while the button is held down, accompanied by two directional buttons (left and right). Below the 'Jogging' section is the 'Drive status' section, which displays four rows of data: 'Command current position' (11482601 pulse), 'Actual current position' (11464089 pulse), 'Command current velocity' (1000000 pulse/s), and 'Actual current velocity' (933000 pulse/s). To the right of the 'Drive status' section is another text box explaining that jogging will be performed only while the button is held down, accompanied by two directional buttons (left and right).

Figure 23. Example configuration

1.7 MC Commands

- **Jogging:** moves the motor in + or – direction at the defined speed.

The screenshot shows the 'Jogging' section of the software interface. It includes input fields for 'Target velocity' (1000000 pulse/s), 'Acceleration' (0 pulse/s^2), and 'Deceleration' (0 pulse/s^2), along with an 'Apply' button. To the right of the 'Jogging' section is a text box explaining that jogging will be performed only while the button is held down, accompanied by two directional buttons (left and right).

Figure 24. Jogging

- **Absolute Positioning:** moves motor to defined position from origin.



Absolute positioning

Target position pulse

Target velocity pulse/s

Acceleration pulse/s²

Deceleration pulse/s²

Jerk pulse/s³

Apply

If you click the button below after clicking the Apply Button, test run will start.

Direction

Figure 25. Absolute Positioning

- **Relative Positioning:** moves motor a defined distance from current position.

Relative positioning

Distance pulse

Target velocity pulse/s

Acceleration pulse/s²

Deceleration pulse/s²

Jerk pulse/s³

Apply

If you click the button below after clicking the Apply Button, test run will start.

Figure 26. Relative Positioning

- **Homing:** defines origin, may use limits, proximity, or home sensors.

Homing

Homing is performed according to axis parameter settings on the Homing page.

Show Homing Settings

Apply Homing Settings

If you click the button below after clicking the Apply Button, test run will start.

Figure 27. Homing



2 Motion - Axis Configuration 2 - Hardware

2.1 Axis Configuration – Axis1 [Conveyor]

After assigning the hardware (servo drive) to the axis via **Axis Basic Settings** and validating correct operation via **MC Test Run**, we continue configuration via **Motion Controller Setup > Axis Settings**.

Button	Name	Description
	Axis Basic Settings	Set whether to use the axis, the axis type, the axis number, an input or output device, and the channel.
	Unit Conversion Settings	Set the gear ratio of the electronic gear. Set the pulses per motor rotation and the travel distance.
	Operation Settings	Set the velocity, acceleration rate, deceleration rate, torque warning values, and other monitor parameters.
	Other Operation Settings	Set the I/O for the Servo Drive.
	Limit Settings	Set software limits and following error limits.
	Homing Settings	Set the homing operation.
	Position Count Settings	Set Count Mode of the Controller.
	Servo Drive Settings	Set the Servo Drive parameters.

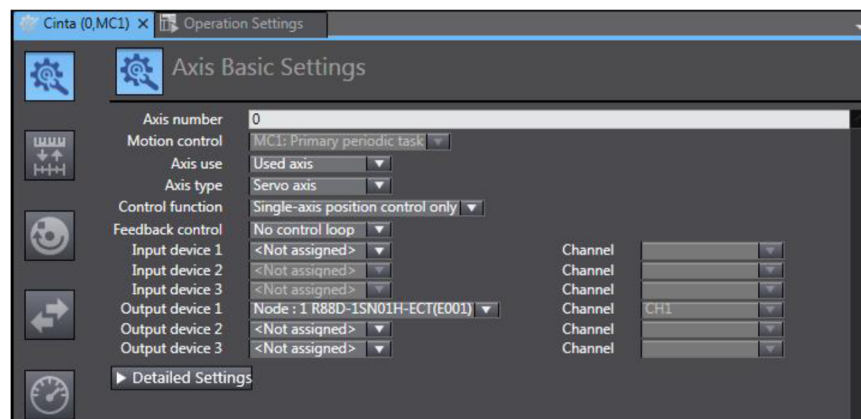


Figure 28. Axis Settings

2.2 Unit Conversion Settings

Unit Conversion Settings

- Configure units for **Axis1 (Conveyor)** in **Unit Conversion Settings**.
- Based on mechanical data:
 - Each motor rotation moves the conveyor **110 mm**
 - Encoder: 23-bit → **8,388,608 pulses/rev**

Settings:



1. Working unit: mm
2. Pulses per rotation: 8,388,608
3. Distance per rotation: 110 mm

→ After these values are set, the **Operation Settings** icon shows an error until configured.

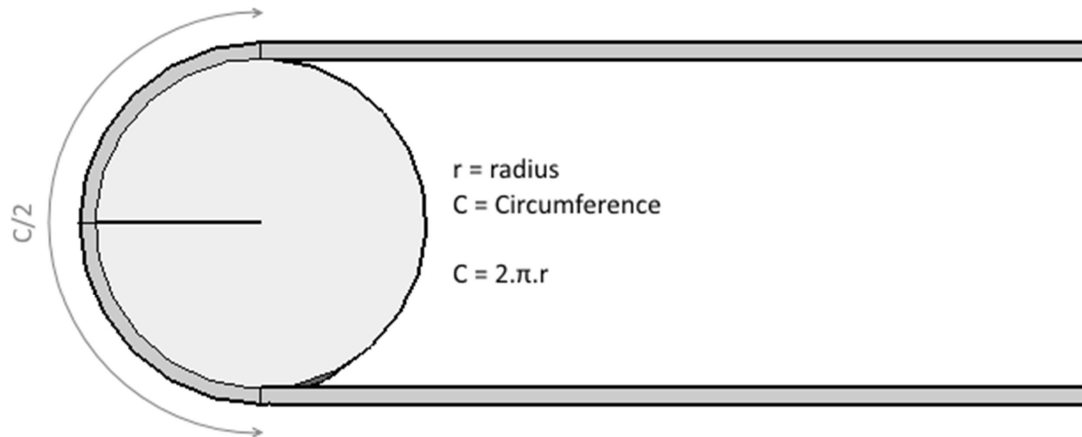


Figure 29. Unit Conversion Settings

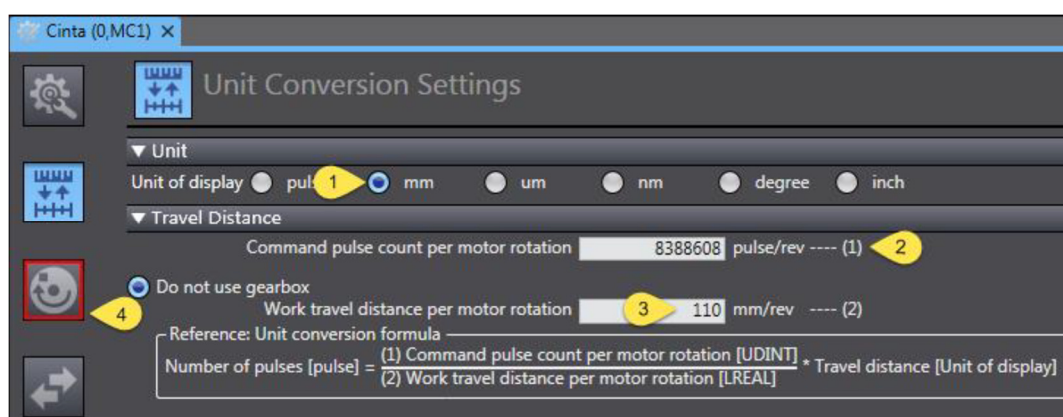
For the existing 1S servo drive / servo motor, we have:

- 23-bit incremental encoder;
- 3,000 rpm.

Standard Servo Motors

Servo motors 1,000 – 2,000 – 3,000 r/min
 100 to 3,000 Kw

Specifications			Servo Drive
Voltage	Encoder	Capacity	1S
230 V	Absolute encoder (23 bit)	50 W	R88D-1SN01H-ECT



(1) Definition of the working unit, in this case **mm**

(2) Pulses per motor revolution

The motor generates $2^{23} = 8,388,608$ pulses per revolution, based on its 23-bit incremental encoder.



(3) Distance traveled per rotation: 110 mm

(4) After defining these values, the **Operation Settings** icon will indicate an error.

In the next step, we must configure the **Operation Settings** for axis 1.

2.3 Operation Settings

Velocity calculations:

- Max Velocity:
 $RPM / 60 = RPS$
 $3000 / 60 = 50 RPS$
 $50 RPS \times 110 \text{ mm} = \mathbf{5,500 \text{ mm/s}}$
- Max Jog Velocity:
 $10 RPS \times 110 \text{ mm} = \mathbf{1,100 \text{ mm/s}}$

2.4 Home Settings

- Since Axis1 controls a conveyor, homing is configured so that when executed, the current position is reset to **zero**.

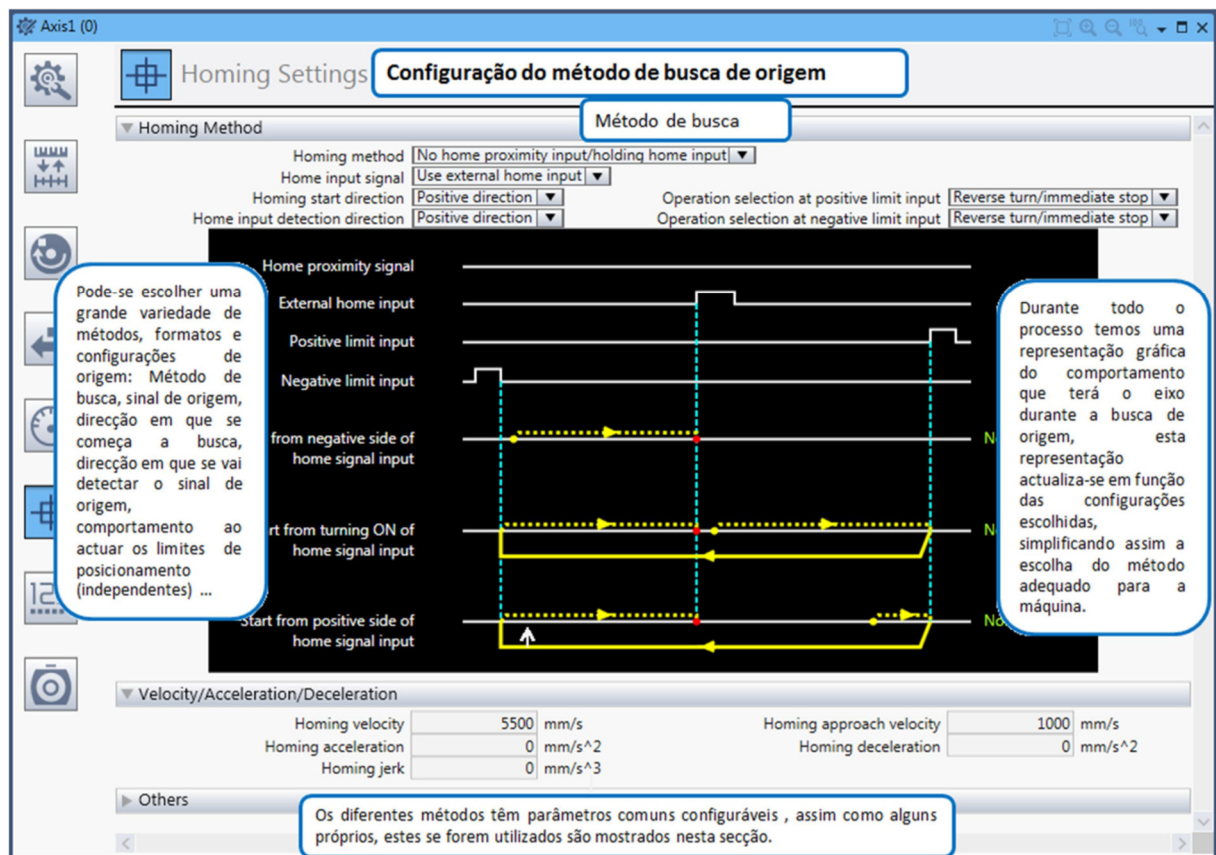


Figure 30. Homing configurations

Since this axis controls a conveyor, the homing method is configured as follows. This means that when the homing instruction is executed, the current axis position is set to zero.

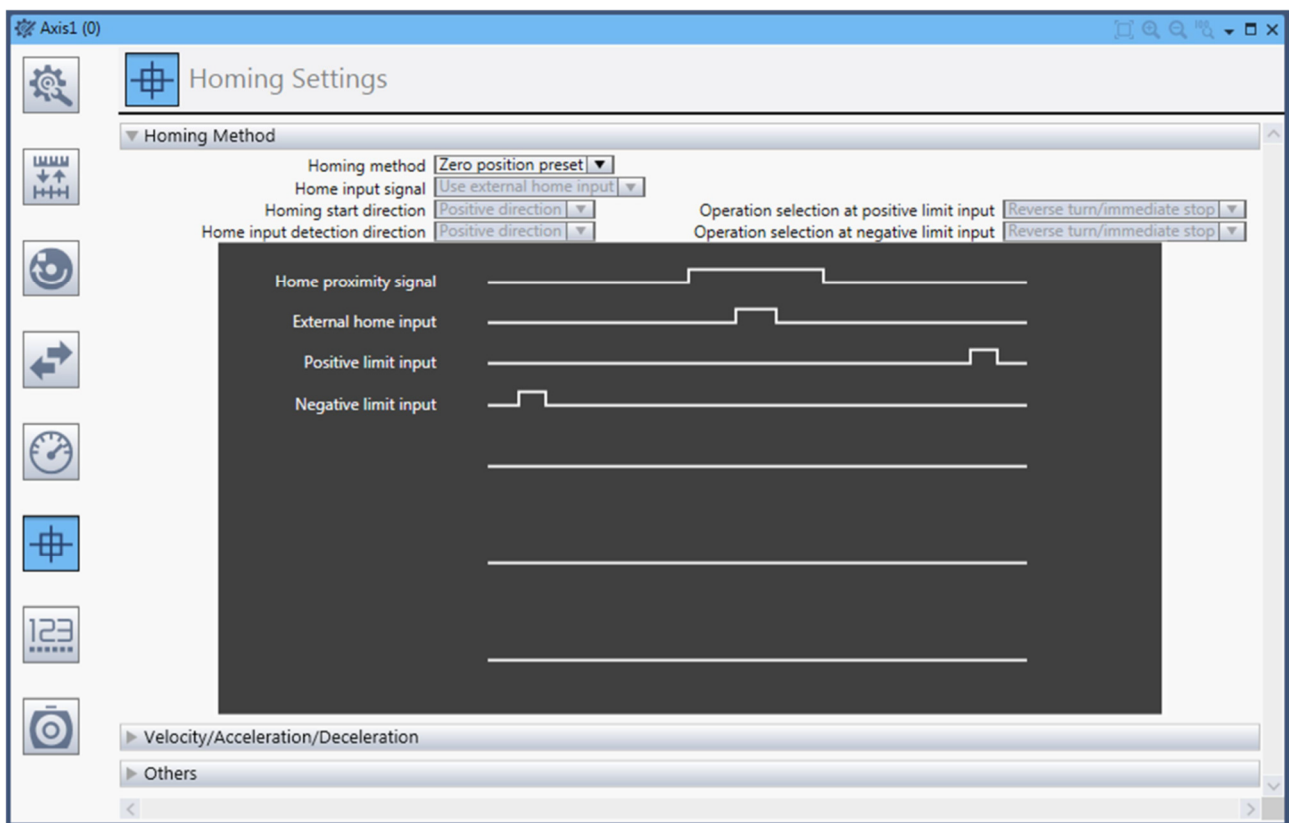
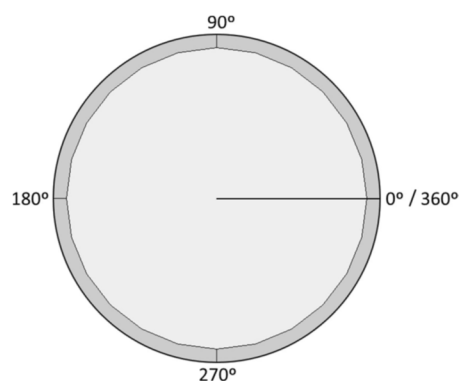


Figure 31. Homing configurations

2.5 Unit Conversion Settings

We must configure the units in which we want to work for **axis 2 (disk)**. These are defined in **Unit Conversion Settings**.

Based on the mechanical element driven by the servo motor of axis 2, we have the following:



Working in degrees [°], we verify that for each motor revolution, the disk moves 360°.

For the servo drive / servo motor used in the **Sysmac didactic bench**, we have:

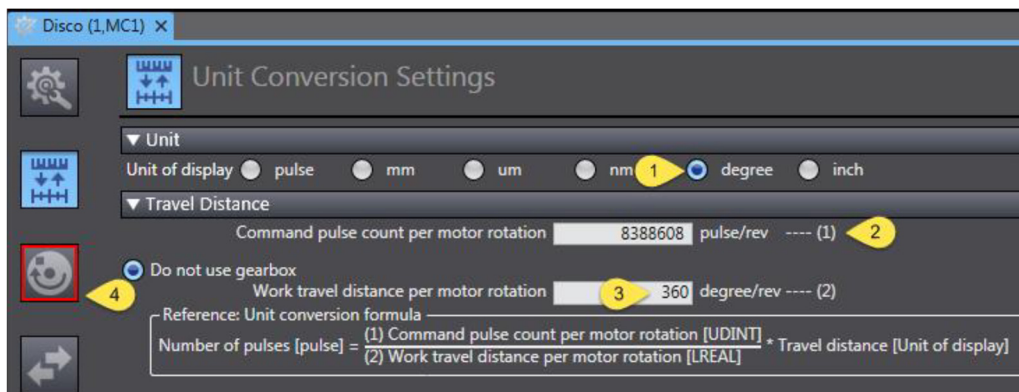
- 23-bit incremental encoder
- 3,000 rpm



Standard Servo Motors

Servo motors 1,000 – 2,000 – 3,000 r/min
100 to 3,000 Kw

Specifications			Servo Drive
Voltage	Encoder	Capacity	1S
230 V	Absolute encoder (23 bit)	50 W	R88D-1SN01H-ECT



(1) Definition of the working unit, in this case degrees [°]

(2) Pulses per motor revolution:

$$2^{23} \text{ bits} = 8,388,608 \text{ pulses/revolution}$$

(3) Distance traveled per revolution: 360°

(4) After setting these values, the **Operation Settings** icon displays an error. In the next step, we must configure the **Operation Settings** for axis 2.

2.6 Operation Settings

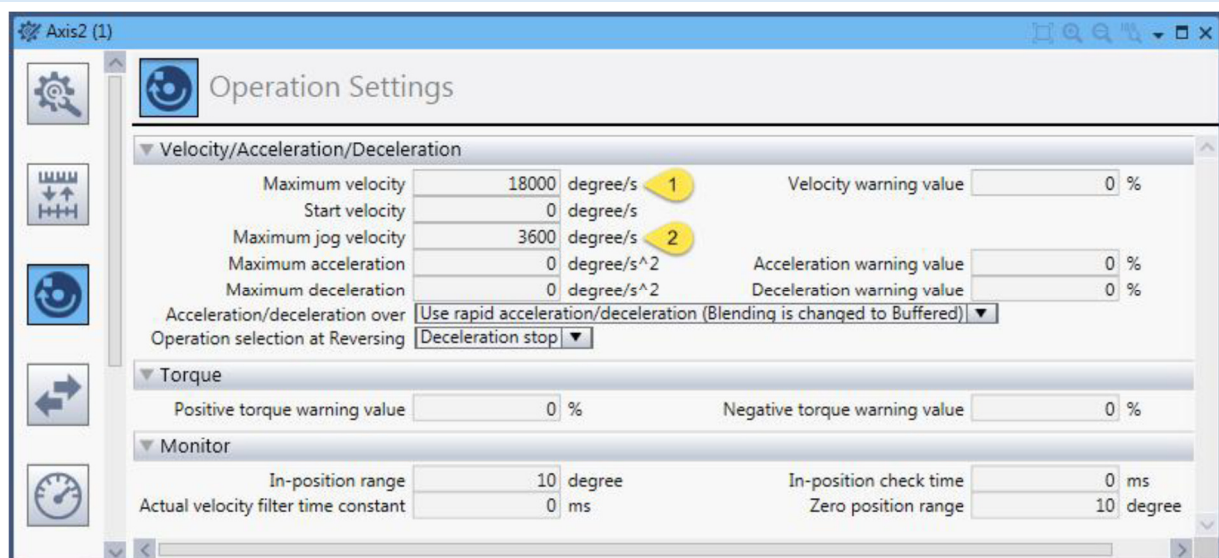


Figure 32. Operation Settings

$$velMax [Hz] = velMax [rpm]/60 [s];$$

$$velMax [Hz] = 3\,000 [rpm]/60 [s] = 50 [Hz];$$

(3) Maximum Speed

18,000 °/s

$$velMax [^{\circ}/s] = velMax [Hz] \times \Delta\theta [^{\circ}]$$

$$velMax [^{\circ}/s] = 50 [Hz] \times 360 [^{\circ}] = 18,000 [^{\circ}/s]$$

(4) Maximum Jog Speed

3,600 °/s

$$velMaxJog [^{\circ}/s] = 10 [Hz] \times \Delta\theta [^{\circ}]$$

$$velMaxJog [^{\circ}/s] = 10 [Hz] \times 360 [^{\circ}] = 3,600 [^{\circ}/s]$$

2.7 Home Settings

In the homing method configuration, we define the behaviour of the axis during the homing operation. Since this axis controls a disk, the homing method is configured as follows.

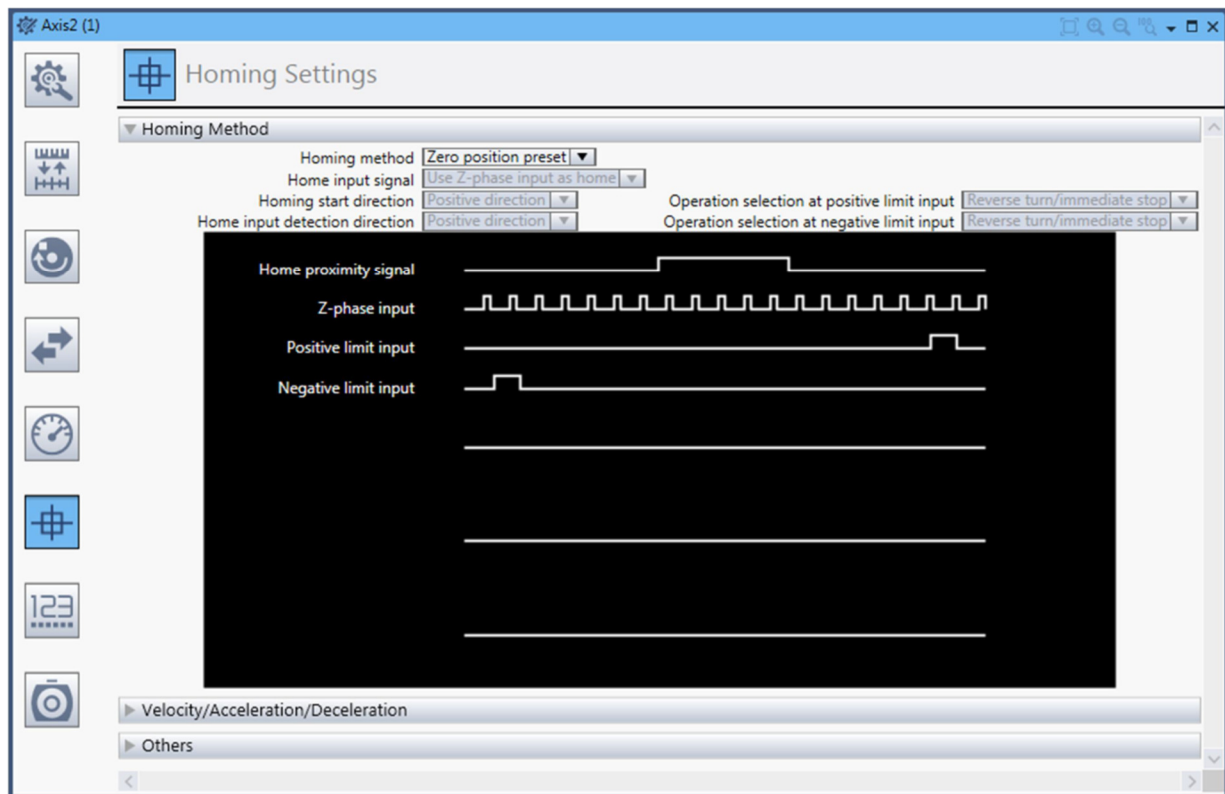


Figure 33. Homing settings – Axis 2

This means that when the homing instruction is executed, the current position of the axis is set to zero.

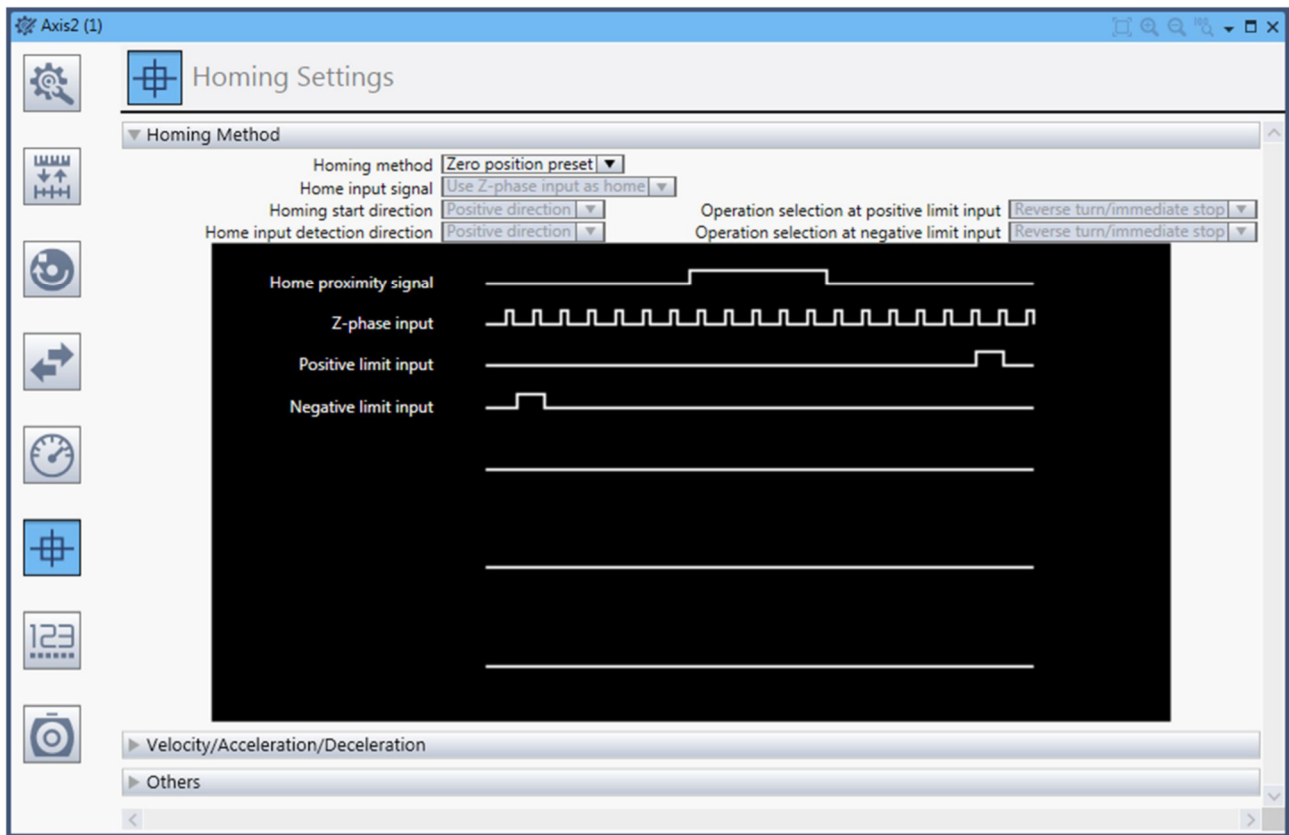


Figure 34. Homing settings – Axis 2

2.8 Position Count Settings

The count mode defines how the axis position is tracked. The position count mode must be set for the command positions of each axis. There are two count modes:

Linear Mode (*Default mode*)

The count mode is set to linear when an axis always moves within predefined limits that will never be exceeded.

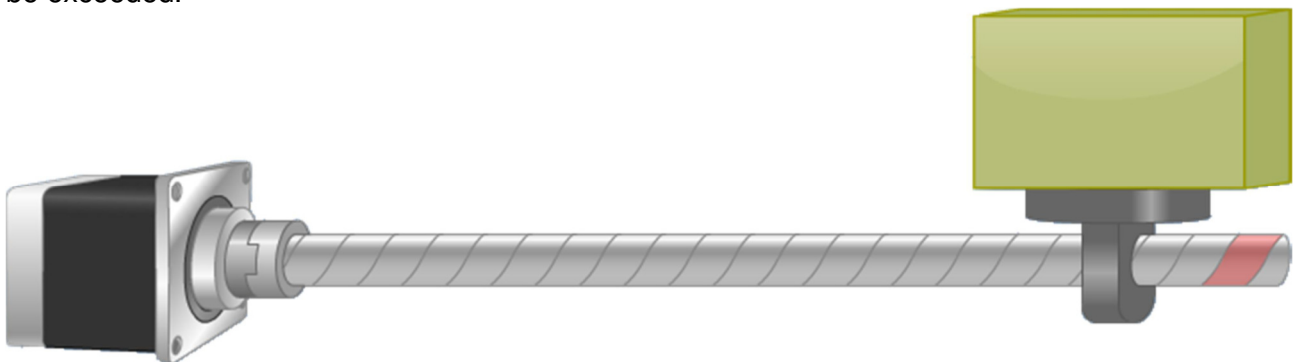


Figure 35. Linear mode

123 Position Count Settings

Count mode ☒ Linear mode ☐ Rotary mode

Modulo maximum position setting value 2147483647 degree

Modulo minimum position setting value -2147483648 degree

Encoder type Incremental encoder

Figure 36. Linear mode example

Rotary Mode

The count mode is set to rotary when an axis performs continuous or infinite motion that exceeds the maximum or minimum value of the Position Module.



Figure 37. Rotary mode

123 Position Count Settings

Count mode ☐ Linear mode ☒ Rotary mode

Modulo maximum position setting value 360 degree

Modulo minimum position setting value 0 degree

Encoder type Incremental encoder

Figure 38. Rotary mode example

For **Axis2 [Disk]**, this means that we will have a new position reference every **90°** of rotation.

1st Rotation					2nd Rotação				
0°	90°	180°	270°	360°	450°	540°	630°	720°	Linear Count Mode
0°	90°	180°	270°	360° / 0°	90°	180°	270°	360° / 0°	Rotary Count Mode

Accordingly, we apply the following configuration to **Axis2 [Disk]**:

Position Count Settings

1. **Rotary Mode**
2. **Maximum position:** 360°
3. **Minimum position:** 0°



Then execute the following steps:

Project > Rebuild Controller > Online > Synchronize > Transfer to Controller

Run a test using **MC Start Test Run** and verify the differences in the position count mode.

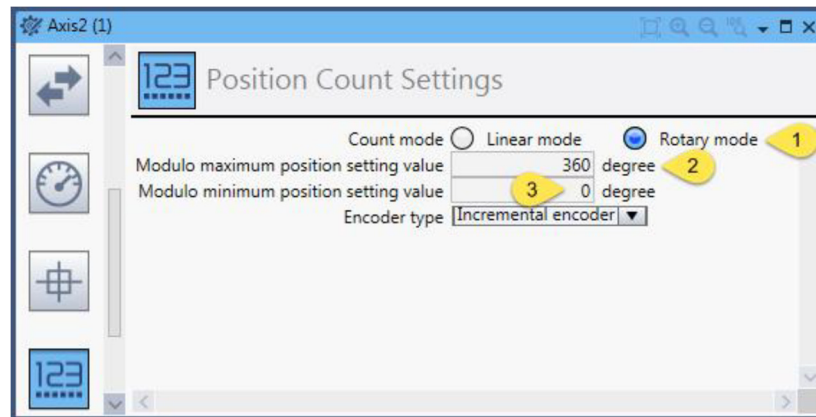


Figure 39. Position Count Settings

3 Motion: Start-up – Software

After assigning the hardware (servo drive) to the axis via **Axis Basic Settings** and validating correct operation via **MC Test Run**, we continue configuration via **Motion Controller Setup > Axis Settings**.

3.1 Start-up Procedure

To move any axis, only **3 commands** are required: MC_Power, MC_Home, and MC_Move, all linked to one of the previously created and configured axes.

Global Variables view in Programming > Data

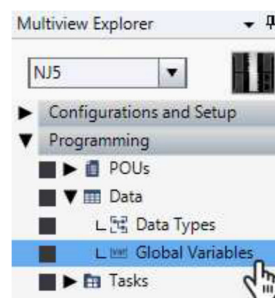


Figure 40. Global variables in *Programming > Data*

When we access global variables via **Programming > Data > Global Variables**, we see that two global variables of type `_sAXIS_REF` were automatically created, one for each axis.

Global Variables							
Name	Data Type	Initial Value	AT	Retain Constant	Network Publish	Comment	
Axis1	_sAXIS_REF		MC://_MC_AX[0]	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Do not publish	
Axis2	_sAXIS_REF		MC://_MC_AX[1]	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Do not publish	

So far, we have been working with local variables. To view them:

1. Go to **View**
2. Open the **Variable Manager**

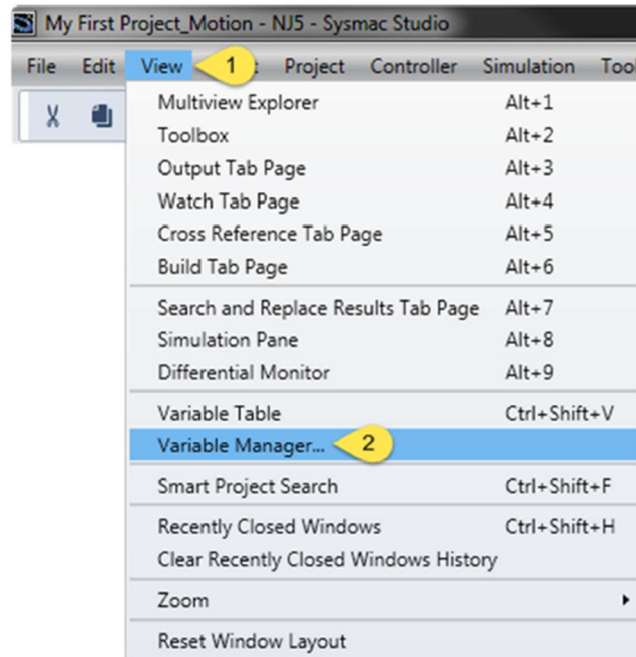


Figure 41. Working with local variables

Switching between Global and Program0 Variables

To convert a local variable to global, right-click the variable and select **Move Variable to Global**.

Scope of Global Variables:

They are accessible from any POU (Program, FB, or FUN). To avoid unintended access, a variable accessed by a POU should be declared as **external**.

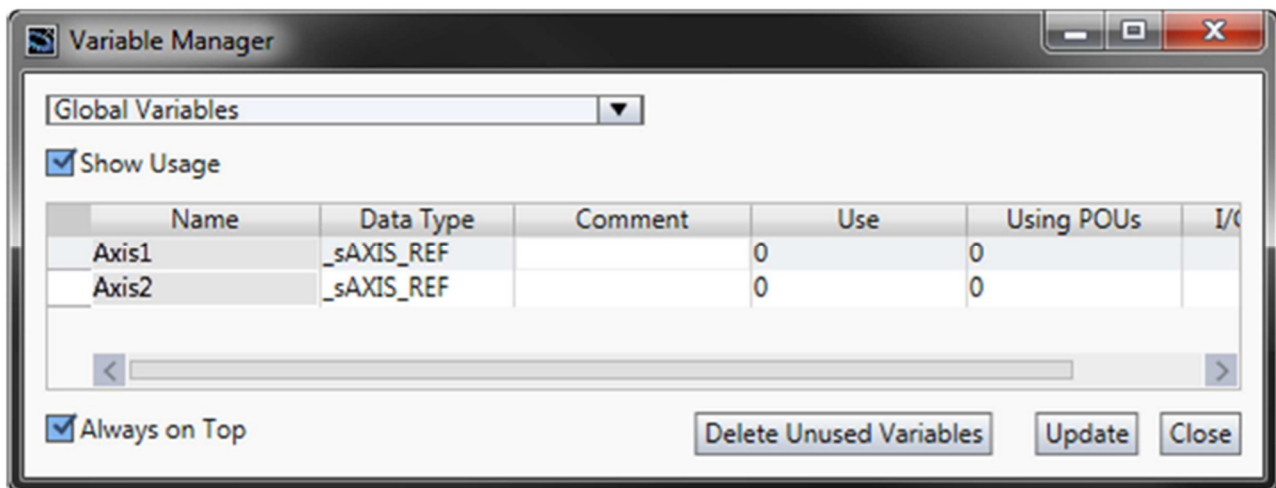


Figure 42. Global variables

To view the **local variables** associated with **Program0**:

1. Switch from **Global Variables** to **Program0**
2. If you want to change a variable from local to global, right-click the variable and select **Move Variable to Global**

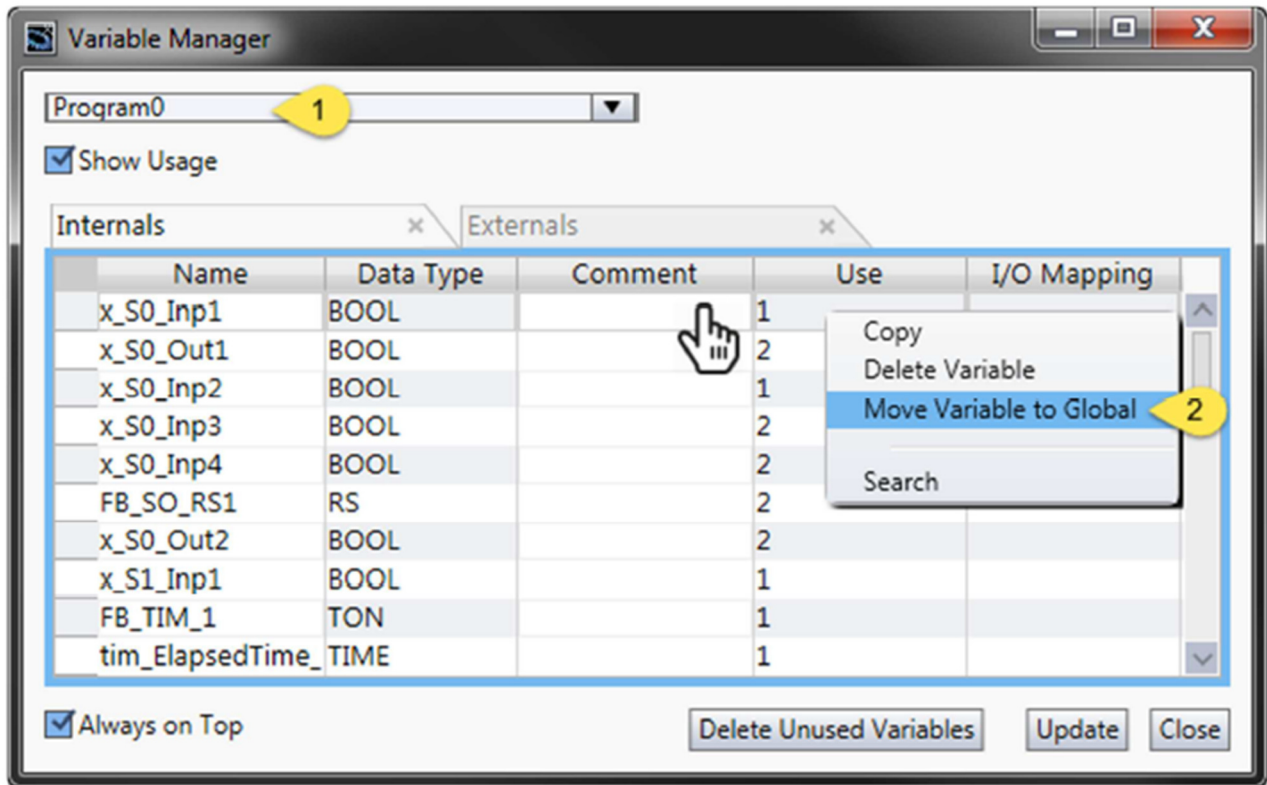


Figure 43. Global Variables to Program0

Scope of Global Variables

Global variables are accessible from all POU (Programs, Function Blocks, and Functions). To prevent unnecessary access and avoid potential issues, access to a POU should be defined as an **external variable**.

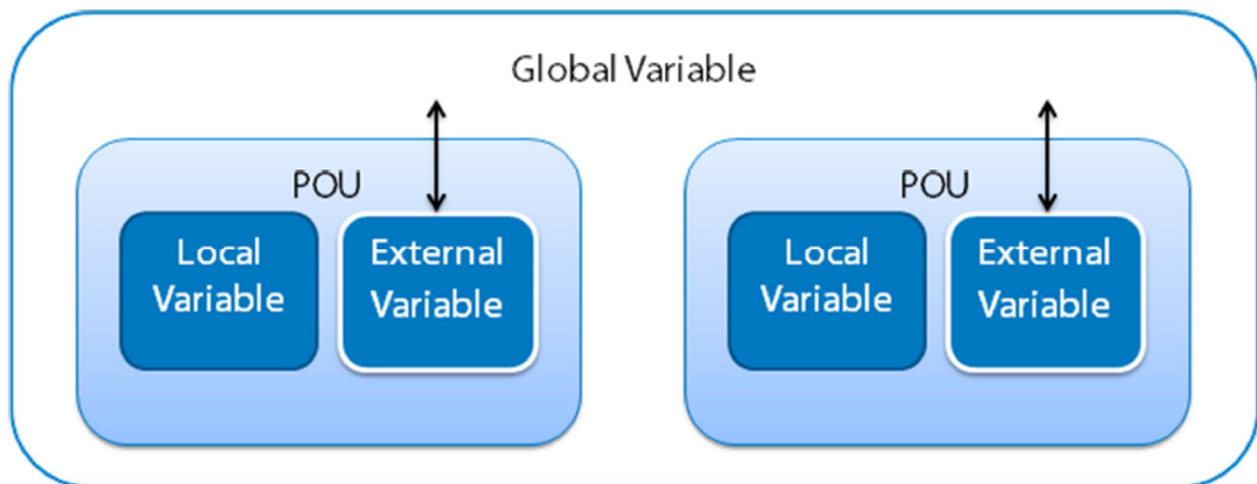


Figure 44. Global variables accessible from all POU



3.2 _sAXIS_REF Structure

Each created axis is of type `_sAXIS_REF`. Using the **Watch Window**, we can inspect the structure of these axes.

Name	Data type	AT
▼ Axis1	<code>_sAXIS_REF</code>	MC://_MC_AX[0]
► Cfg	<code>_sAXIS_REF_CFG</code>	
► Scale	<code>_sAXIS_REF_SCALE</code>	
► Status	<code>_sAXIS_REF_STA</code>	
► Details	<code>_sAXIS_REF_DET</code>	
► Dir	<code>_sAXIS_REF_DIR</code>	
► DrvStatus	<code>_sAXIS_REF_STA_DRV</code>	
► Cmd	<code>_sAXIS_REF_CMD_DATA</code>	
► Act	<code>_sAXIS_REF_ACT_DATA</code>	
► MFaultLvl	<code>_sMC_REF_EVENT</code>	
► Obsr	<code>_sMC_REF_EVENT</code>	

AXIS_REF	
Setup and Command	
Monitoring and Status	
Alarms and Warnings	

▼ Cfg	<code>_sAXIS_REF_CFG</code>
AxNo	UINT
AxEnable	<code>_eMC_AXIS_USE</code>
AxType	<code>_eMC_AXIS_TYPE</code>
NodeAddress	UINT
ExecID	UINT

▼ Scale	<code>_sAXIS_REF_SCALE</code>
Num	UDINT
Den	LREAL
Units	<code>_eMC_UNITS</code>
CountMode	<code>_eMC_COUNT_MODE</code>
MaxPos	LREAL
MinPos	LREAL

▼ Status	<code>_sAXIS_REF_STA</code>
Ready	BOOL
Disabled	BOOL
Standstill	BOOL
Discrete	BOOL
Continuous	BOOL
Synchronized	BOOL
Homing	BOOL
Stopping	BOOL
ErrorStop	BOOL
Coordinated	BOOL

▼ Details	<code>_sAXIS_REF_DET</code>
Idle	BOOL
InPosWaiting	BOOL
Homed	BOOL
InHome	BOOL
Vellimit	BOOL

▼ Dir	<code>_sAXIS_REF_DIR</code>
Posi	BOOL
Nega	BOOL

▼ DrvStatus	<code>_sAXIS_REF_STA_DRV</code>
ServoOn	BOOL
Ready	BOOL
MainPower	BOOL
P_OT	BOOL
N_OT	BOOL
HomeSw	BOOL
Home	BOOL
ImdStop	BOOL
Latch1	BOOL
Latch2	BOOL
DrvAlarm	BOOL
DrvWarning	BOOL
ILA	BOOL
CSP	BOOL
CSV	BOOL
CST	BOOL

▼ Cmd	<code>_sAXIS_REF_CMD_DATA</code>
Pos	LREAL
Vel	LREAL
AccDec	LREAL
Jerk	LREAL
Trq	LREAL

▼ Act	<code>_sAXIS_REF_ACT_DATA</code>
Pos	LREAL
Vel	LREAL
Trq	LREAL
TimeStamp	ULINT

▼ MFaultLvl	<code>_sMC_REF_EVENT</code>
Active	BOOL
Code	WORD

▼ Obsr	<code>_sMC_REF_EVENT</code>
Active	BOOL
Code	WORD

Figure 45. `_sAXIS_REF` Structure

3.3 Motion Basics

In **Programs**, add a new program in **Ladder**, naming it `Program_Motion`. Call the initial section: `Section0_MotionBasics`.

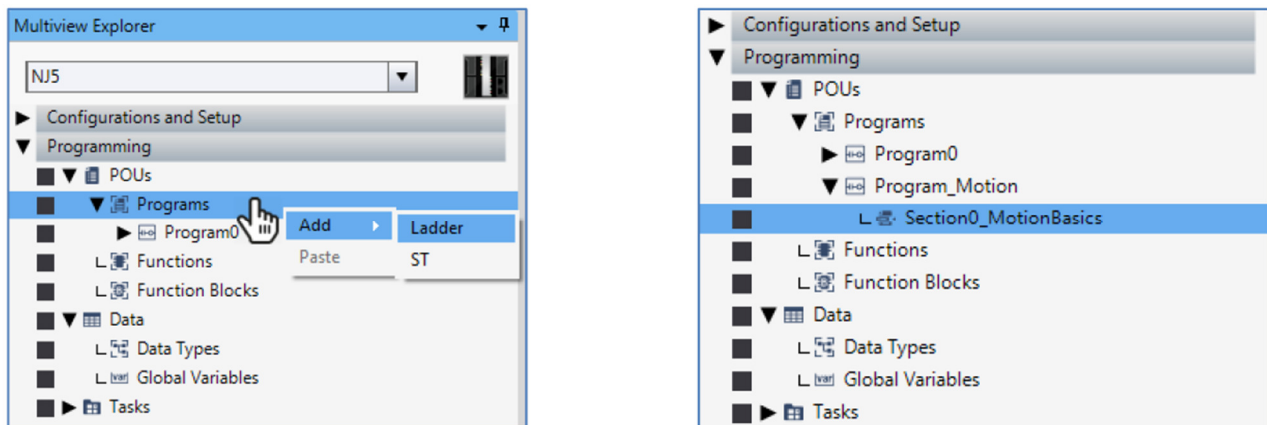


Figure 46. `Section0_MotionBasics`

3.4 Task

New programs must be assigned to a **Task**, otherwise the controller won't execute them.

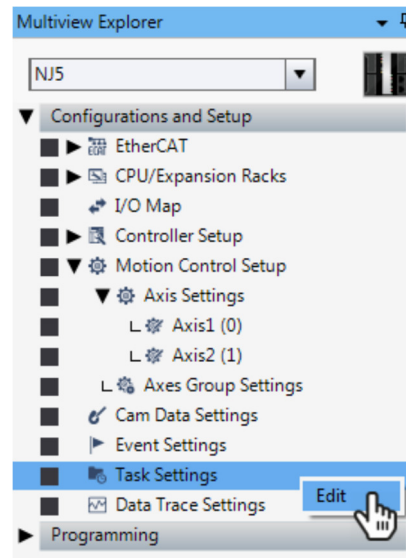


Figure 47. Accessing Task Settings and creating assignments

The **Task Settings** window opens. Here, you can create **three types of tasks**:

- **Primary task**
 - Priority level: **4**
- **Additional periodic tasks** can be created and assigned a lower priority
 - Priority levels: **16, 17, and 18**
- **Event tasks** can be configured to execute when a specific condition is met
 - Priority levels: **8 and 48**

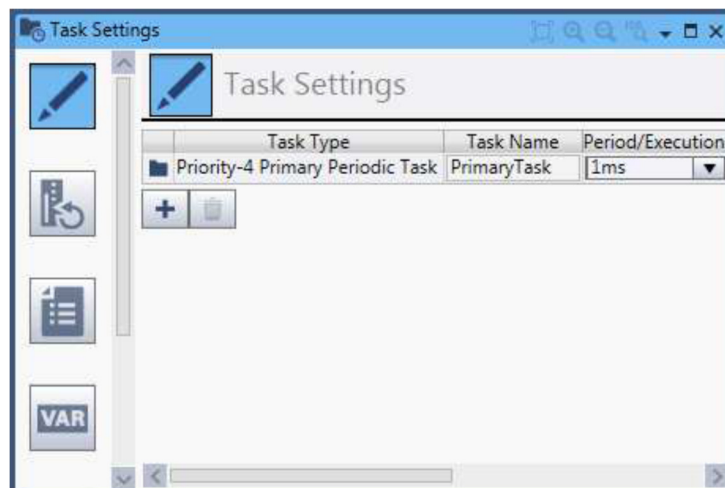


Figure 48. Tasks and priorities

For this example, we will use the existing **primary task**.



1. Select the **Program Assignment Settings** tab
2. Click + to add a program to the primary task
3. Click on the **drop-down list**
4. Select the previously created program, Program_Motion

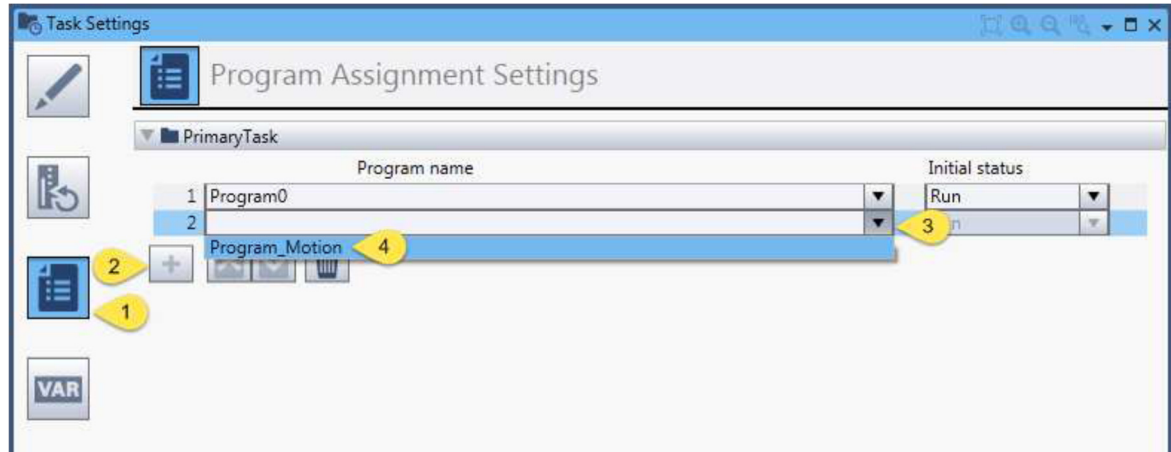
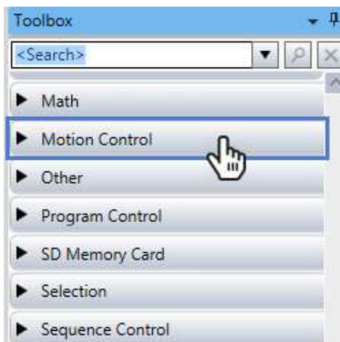


Figure 49. Program Assignment Settings tab

3.5 MC_Power

The first step in using an axis is to **power it on**.



To do this, use the MC_Power instruction, which can be found in the **Toolbox**, under the **Motion Control** tab.

Whenever the prefix MC_ is used in the system, it refers to **Motion Control**.

Objective: Power on the created axes using separate digital inputs.

Name	Data Type
x_MS0_Axis1_PowerOn	BOOL
FB_MC_Power_Axis1	MC_Power
x_MS0_Axis2_PowerOn	BOOL
FB_MC_Power_Axis2	MC_Power



3.6 MC_Power – Exercise 1

Create a program in Sysmac Studio to test the functions FB_MC_POWER to Axis 1 and axis 2

Once the programming is complete, perform the following steps:

Project > Check All Programs > Rebuild Controller > Online > Synchronize > Transfer To Controller

After synchronizing the project with the controller, validate and verify the code operation.

3.7 MC_Home – Exercise 2

Depending on the homing method defined for each axis, the MC_Home instruction will execute the homing operation.

In this case, when the homing instruction is executed, the current position is set to zero.

Objective: Apply the homing command to each of the axes

Naming convention:

Name	Data Type
FB_MC_Power_Axis2	MC_Power
x_MS0_Axis1_Homing	BOOL
FB_MC_Home_Axis1	MC_Home
x_MS0_Axis2_Homing	BOOL
FB_MC_Home_Axis2	MC_Home

Create a program in Sysmac Studio to test the functions FB_MC_HOME to Axis 1 and axis 2

Once programming is complete, perform:

Project > Check All Programs > Rebuild Controller > Online > Synchronize > Transfer To Controller

After synchronizing the project with the controller, validate and test the code.

Executing MC_Power for each axis and then triggering the inputs that student have created.

Note: student can verify via the **Watch Window** that the homing instruction is executed.

Also, verify the position values of **Axis1 (Conveyor)** and **Axis2 (Disk)** and test if different from zero, are then reset to **zero**.

3.8 MC_MoveVelocity – Exercise 3

After executing MC_Power and MC_Home (if necessary), the next step is to use MC_Move to put the axis in motion.

Objective: Apply the MC_MoveVelocity instruction to each of the axes



Naming convention:

Name	Data Type
x_MS0_Axis1_MoveVel	BOOL
FB_MC_MoveVel_Axis1	MC_MoveVelocity
Ir_Axis1_MoveVel	LREAL
Ir_Axis1_Acc	LREAL
Ir_Axis1_Dcc	LREAL
FB_MC_Stop_Axis1	MC_Stop
x_MS0_Axis2_MoveVel	BOOL
FB_MC_MoveVel_Axis2	MC_MoveVelocity
Ir_Axis2_MoveVel	LREAL
Ir_Axis2_Acc	LREAL
Ir_Axis2_Dcc	LREAL
FB_MC_Stop_Axis2	MC_Stop

Objective: Analyze the motion logic applied to each axis using the MC_MoveVelocity and MC_Stop function blocks.

1. Apply the FB MC_MoveVelocity Code to Axis1 (Conveyor)
2. Sets the motion profile with velocity, acceleration, and deceleration values.
3. Starts the conveyor when the system is powered and a start command is given.
4. Stops the axis smoothly using the MC_Stop block.

Task 1: Explain the role of each input and output in the MC_MoveVelocity block for Axis1.

Task 2: What would happen if the x_MS0_Axis1_MoveVel command were suddenly interrupted? Justify your answer.

5. Apply the FB MC_MoveVelocity Code Applied to Axis2 (Disk)
6. Initializes the motion profile for the rotating disk.
7. Starts the motion of Axis2 based on the command input.
8. Applies deceleration and stops the disk in a controlled manner.

Task 3: What are the main differences between the way Axis1 and Axis2 are controlled?

Task 4: In what real-world situations would this type of motion control be useful for a labeling disk?

Once the programming is complete, perform the following steps:

Project > Check All Programs > Rebuild Controller > Online > Synchronize > Transfer To Controller

After synchronizing the project with the controller, validate and verify the code operation.

4 Motion: Exercise - Software

The goal is to create a program that controls a **conveyor belt (Axis1 - Conveyor)** and a **label applicator disk (Axis2 - Disk)**.

4.1 Exercise

To move any axis, only **3 commands** are required: MC_Power, MC_Home, and MC_Move, all linked to one of the previously created and configured axes.



Figure 50. Axis1 (Conveyor)

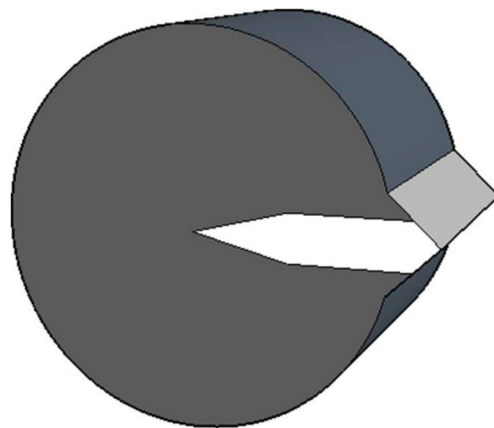


Figure 51. Axis 2 (Disk)

Components are transported along the conveyor, which has **two sensors** installed along its path.

If a component **triggers sensor 1** and, after **5 seconds**, does **not trigger sensor 2**, this means the component is **stuck at the entry zone**.

In this case, the system must **stop**.

The conveyor runs at a speed defined in the program code, and when the **detector** located at the **labeling station** is triggered, the **conveyor must stop** until the label is placed by the **disk**.

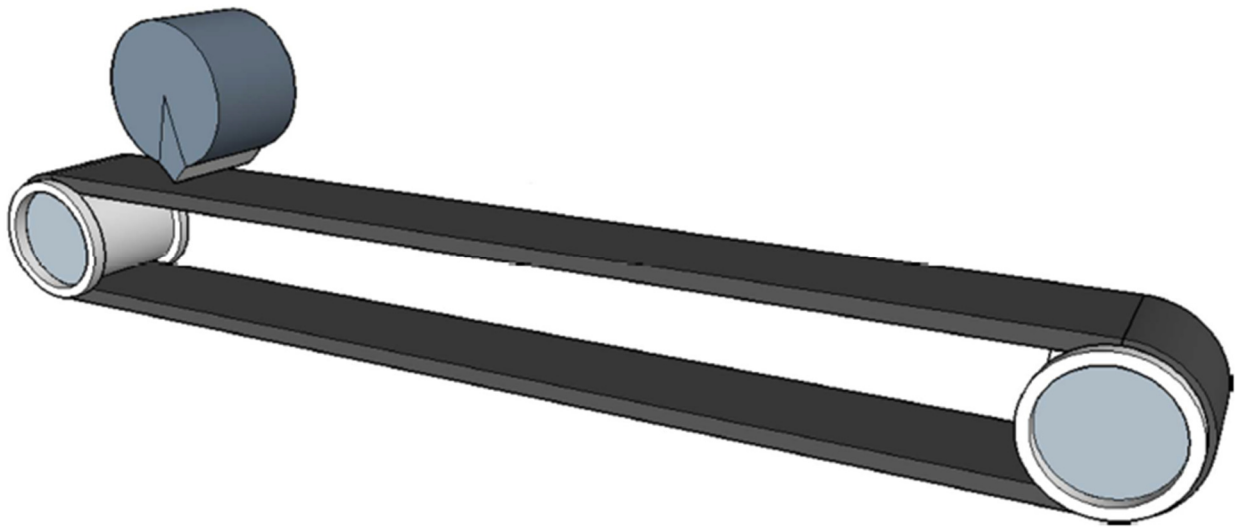
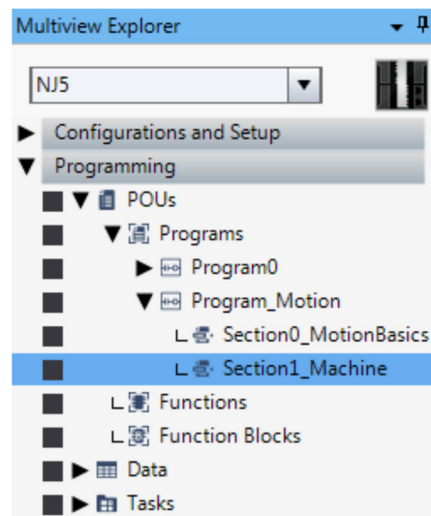


Figure 52. Final exercise

4.2 Exercise 1

In **Programming** > **Programs** > **Program_Motion**, create a **new section** named: **Section1_Machine**



Exercise – Motion Sequence Programming

The following logic rungs are part of a labeling system that uses two axes:

- **Axis1:** Conveyor
- **Axis2:** Labeling Disk

Student goal is to analyze, understand, and complete the motion sequence logic described below:



Rung Descriptions

- **Rung 0**
Activates the servos based on the digital input x_MS1_RUN.
- **Rungs 1 / 2 / 3**
The variables X_Box1_Detection and X_Box2_Detection simulate the passage of components past **Sensor 1** and **Sensor 2**.
If Sensor 1 is triggered but Sensor 2 is **not** triggered within **5 seconds**, the system must stop by **resetting the run command**.
- **Rung 4**
After Axis2 (Disk) is powered on, its **homing procedure** is executed.
- **Rungs 5 / 6 / 7**
Once Axis2 has been homed, the **conveyor (Axis1)** starts moving until the **labeling position detector** is triggered (x_Box1Mark).
At this point, the conveyor must stop to allow labeling.

Note: You will need to add **3 extra conditions** to Rungs 6 and 7 (see Exercise 2).

- **Rungs 8 / 9 / 10 / 11 / 12**
When the conveyor stops because a component is detected at the labeling position, the **disk moves from its resting position to the labeling position**, waits **3 seconds**, and then returns to its home position.

•

4.3 Exercise 2 - Complete the Logic

Student must **complete Rungs 6 and 7** to ensure that the conveyor **resumes movement after the label is applied**.

- Add **point 1** to **Rung 6**



- Add **points 2 and 3** to **Rung 7**



These additions must ensure that the system **automatically restarts** conveyor motion **after the labeling process finishes**.

Once completed, test the logic in simulation and validate the expected behavior of the conveyor and labeling disk.