

Yaskawa Motors & Controller with MotionWorks

Objective

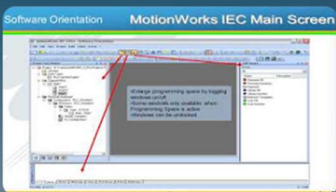
The objective of this lab is to provide an introduction to Yaskawa motor/controller hardware and its Motionworks IEC 3 software. The contents of this lab will include:

- How to create and execute basic block diagram programs
- How to create 1-axis motion
- How to link 2 axes motion

Background

When companies manufacture electronics and automation tools it is common for their products to have firmware already installed and documentation on how to use it. Engineers in industry spend time interpreting documentation and using firmware to design their processes.

We will be going through the set up and using the actual documentation as if you were to receive these motors and prepare them for use within your company. This lab and program blocks we are using are based on trainings used by Yaskawa to train employees and clients on their products seen below.



Training: IEC 61131-3 Basics with MotionWorks IEC

Yaskawa America
21 videos • 87,771 views • Last updated on Nov 3, 2022

[Play all](#) [Shuffle](#)

These videos comprise the official Yaskawa training course "IEC 61131-3 Basics with MotionWorks IEC"

Allow a total of 6-12 hours to view the videos and complete the programming exercises.

Full course information available on Yaskawa America, Inc. - Drives & Motion Division learning management system, "Learn with Yaskawa" at <https://training.yaskawa.com/learning-paths/iec61131-3-basics-with-motionworks-iecsg>.

Use Learn with Yaskawa for all your Yaskawa product training needs.
<https://training.yaskawa.com>









1	 <p>1.1 Software Orientation (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 24K views • 9 years ago 14:46</p>
2	 <p>MotionWorks IEC - Registration Yaskawa America • 9K views • 9 years ago 4:19</p>
3	 <p>2.1 Programming Quick Start Part 1 (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 26K views • 9 years ago 19:21</p>
4	 <p>2.2 Programming Quick Start Part 2 (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 12K views • 9 years ago 16:10</p>
5	 <p>2.3 Help Documentation (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 5.6K views • 9 years ago 11:33</p>
6	 <p>2.4 Using I/O Part 1 (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 8.6K views • 9 years ago 18:34</p>
7	 <p>2.5 Using I/O Part 2 (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 6.9K views • 9 years ago 22:58</p>
8	 <p>2.6 Monitoring (IEC 61131-3 Basics with MotionWorks IEC) Yaskawa America • 4.9K views • 9 years ago 23:37</p>

Figure 0: Yaskawa Training Module available on YouTube

Step 1: Move Motor with Code

- Open MotionWorks IEC and it should look like Figure 1. We are going to now create the blocks necessary for Axis motion.
- On the left side of the screen in the Project Tree Window, navigate to Project > Logical POUs > Main > Main and double click it as highlighted in Figure 1 to get to the correct workspace.

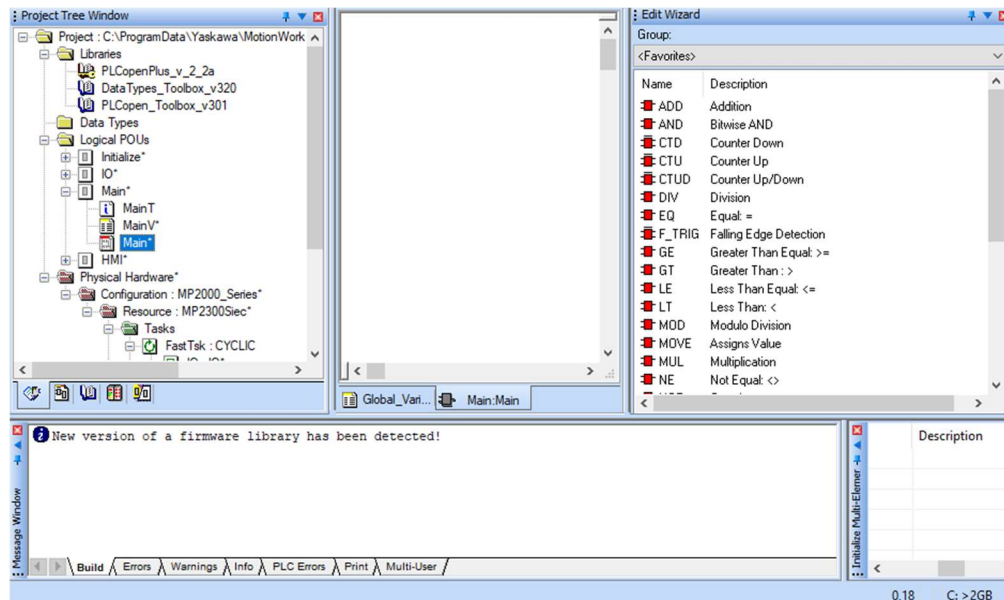


Figure 1: MotionWorks Workspace

- On the right side of the screen in the **Edit Wizard**, choose **PLCopenPlus** from the drop-down menu and click and drag **MC_Power** out onto the blank workspace in the middle of the window.

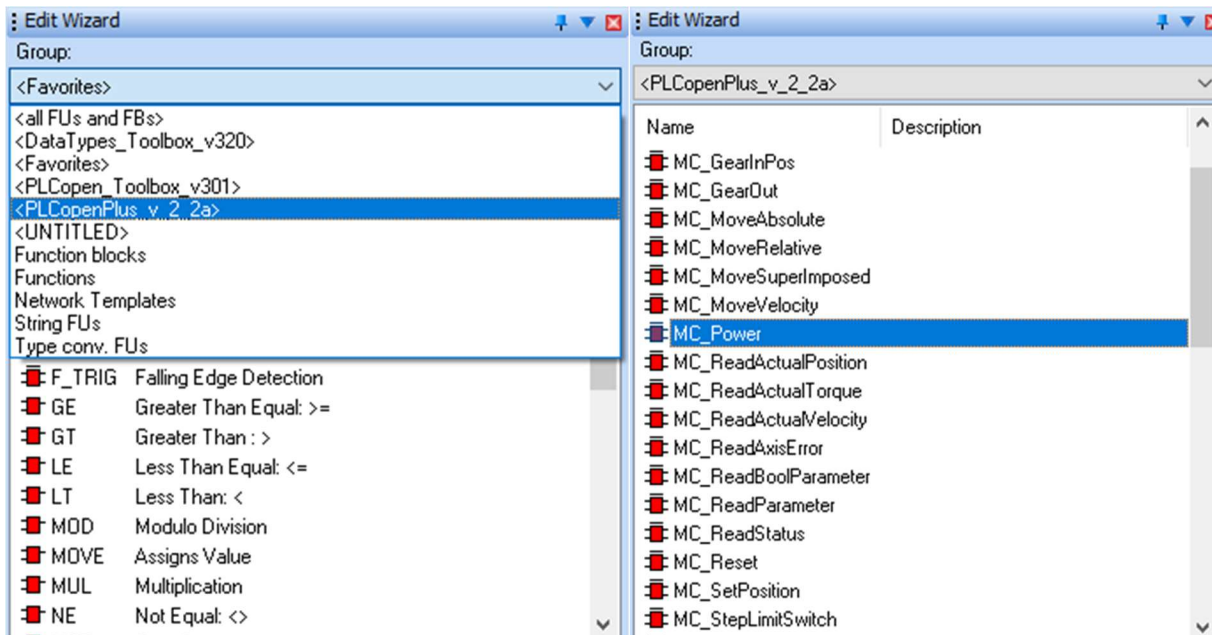


Figure 2: PLCopenPlus in Edit Wizard

- The window in Figure 3 should pop up but we can ignore it and just hit OK.

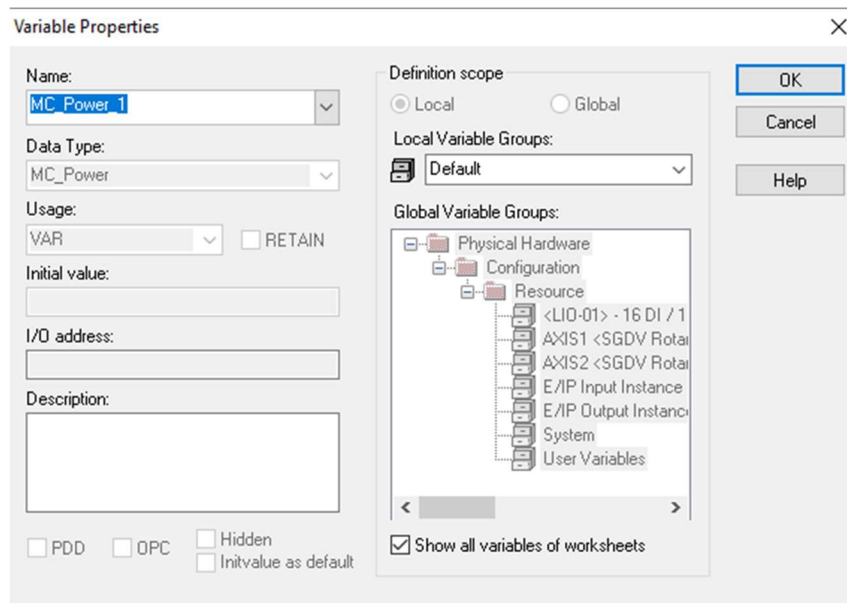


Figure 3: Variable Properties

- Double-click 'Axis' on the MC_Power block and configure its variable properties so that it matches with Figure 4 and repeat for 'Enable' in Figure 5
- The final product should look like Figure 6

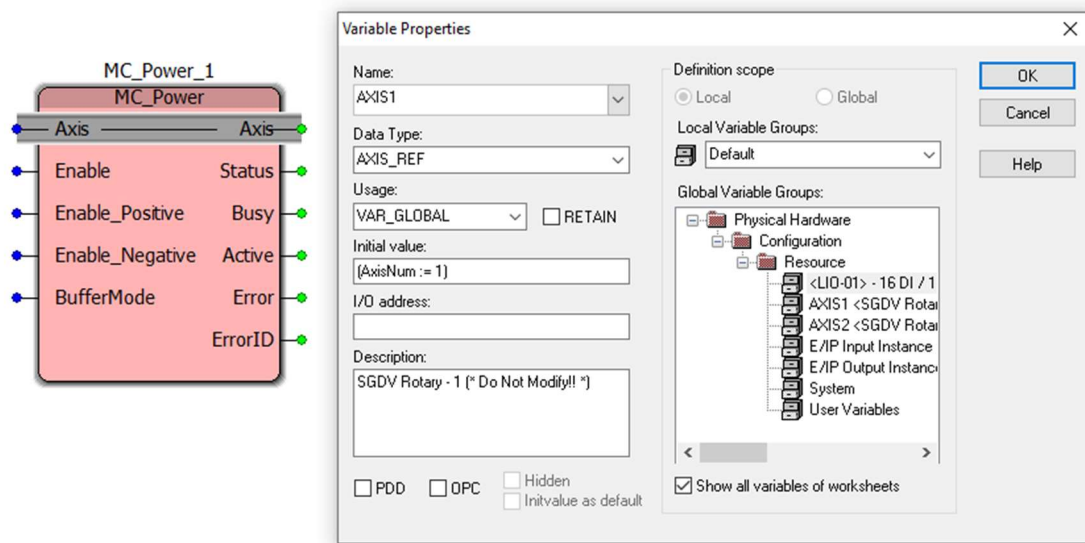


Figure 4: MC_Power Axis Properties

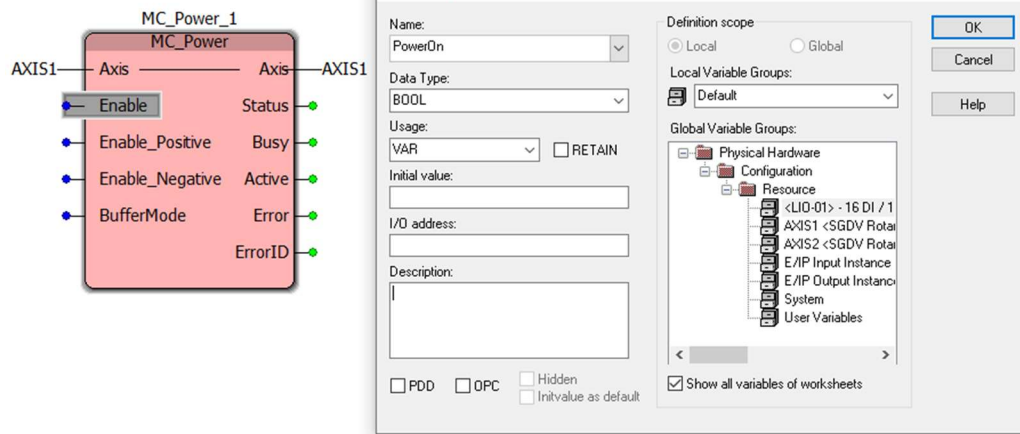


Figure 5: MC_Power Enable Properties

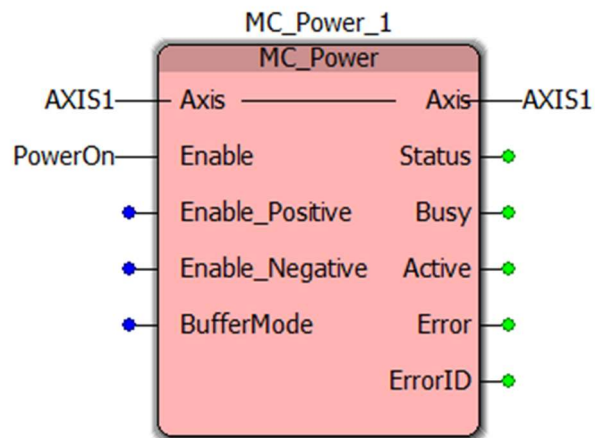


Figure 6: MC_Power Final Product

- Next create an MC_MoveRelative block and configure the variables to match with Figures 7-11
- The final product should look like Figure 12

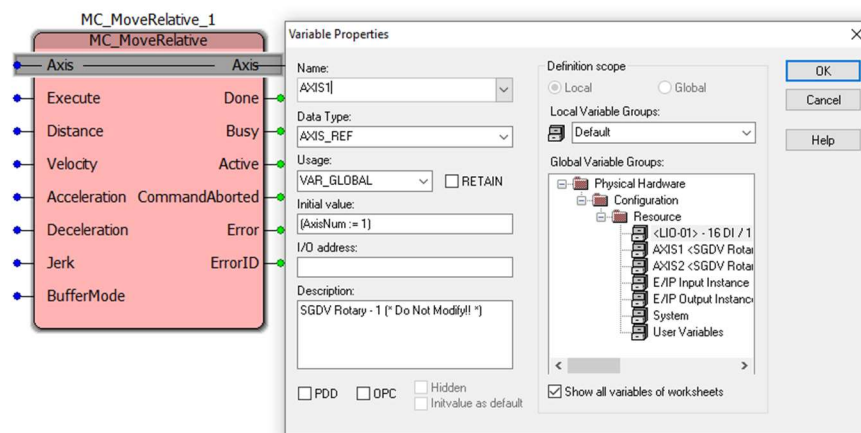


Figure 7: MC_MoveRelative Axis Properties

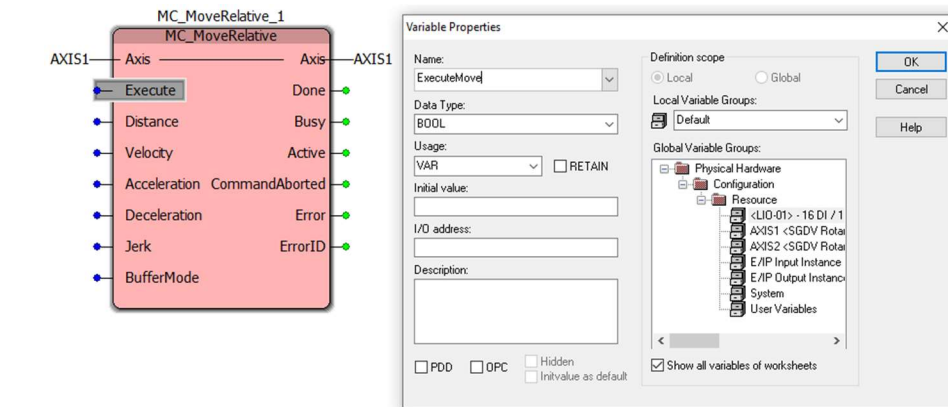


Figure 8: MC_MoveRelative Execute Properties

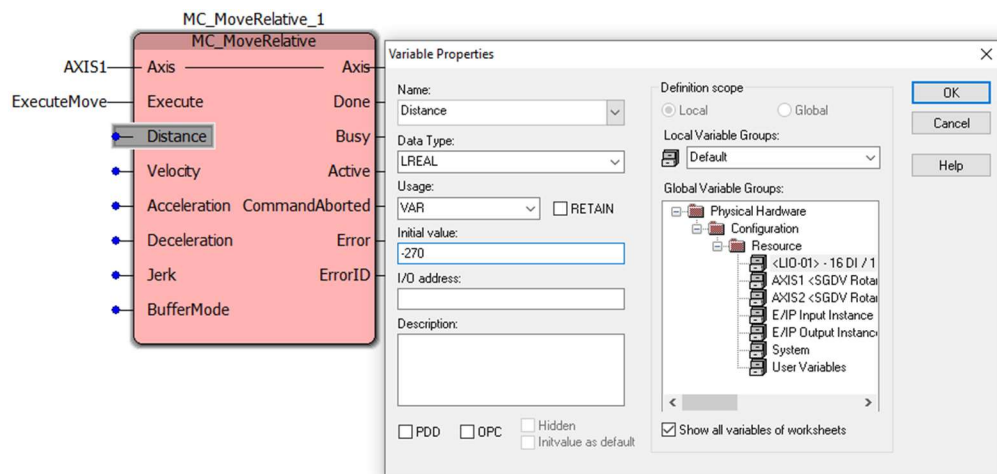


Figure 9: MC_MoveRelative Distance Properties

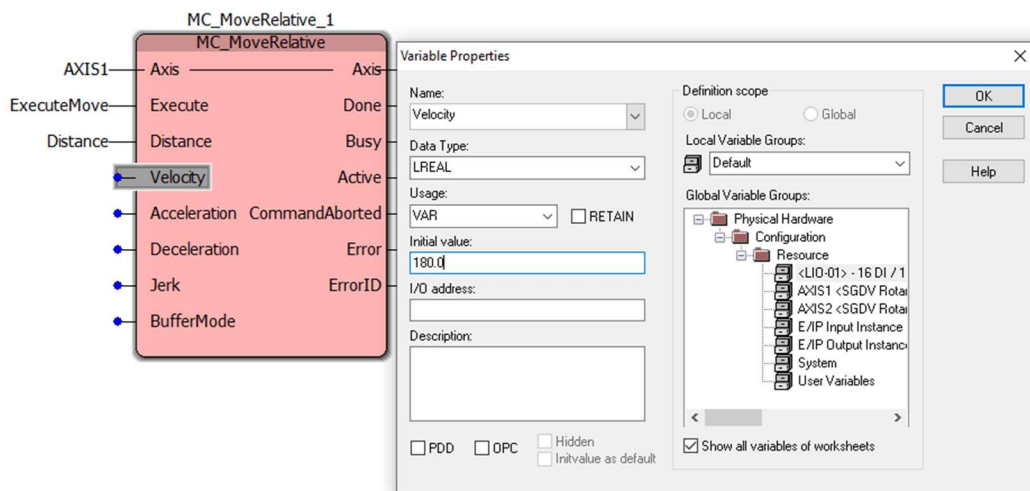


Figure 10: MC_MoveRelative Velocity Properties

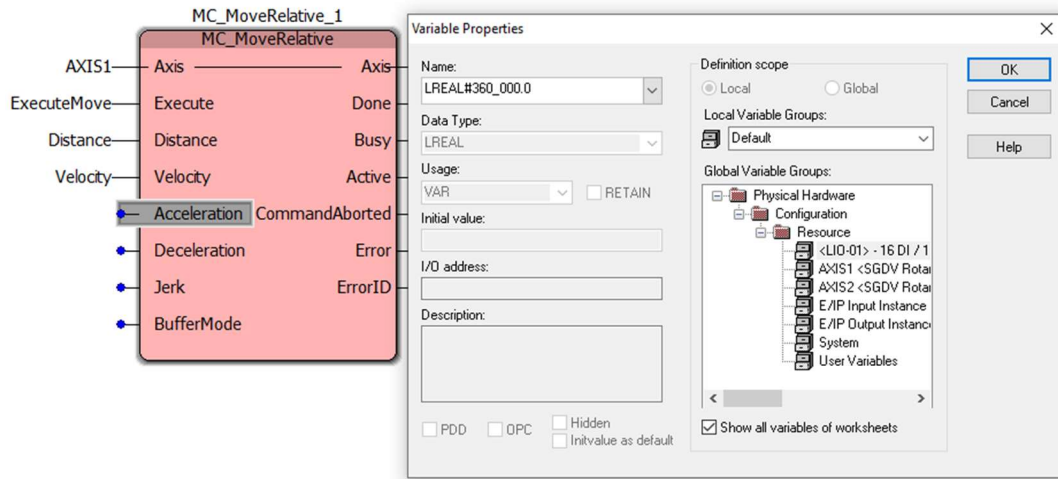


Figure 11: MC_MoveRelative Acceleration/Deceleration Properties

- Once you are done configuring all the variables and the block matches Figure 12, click 'Download Changes' which is also shown in Figure 12.

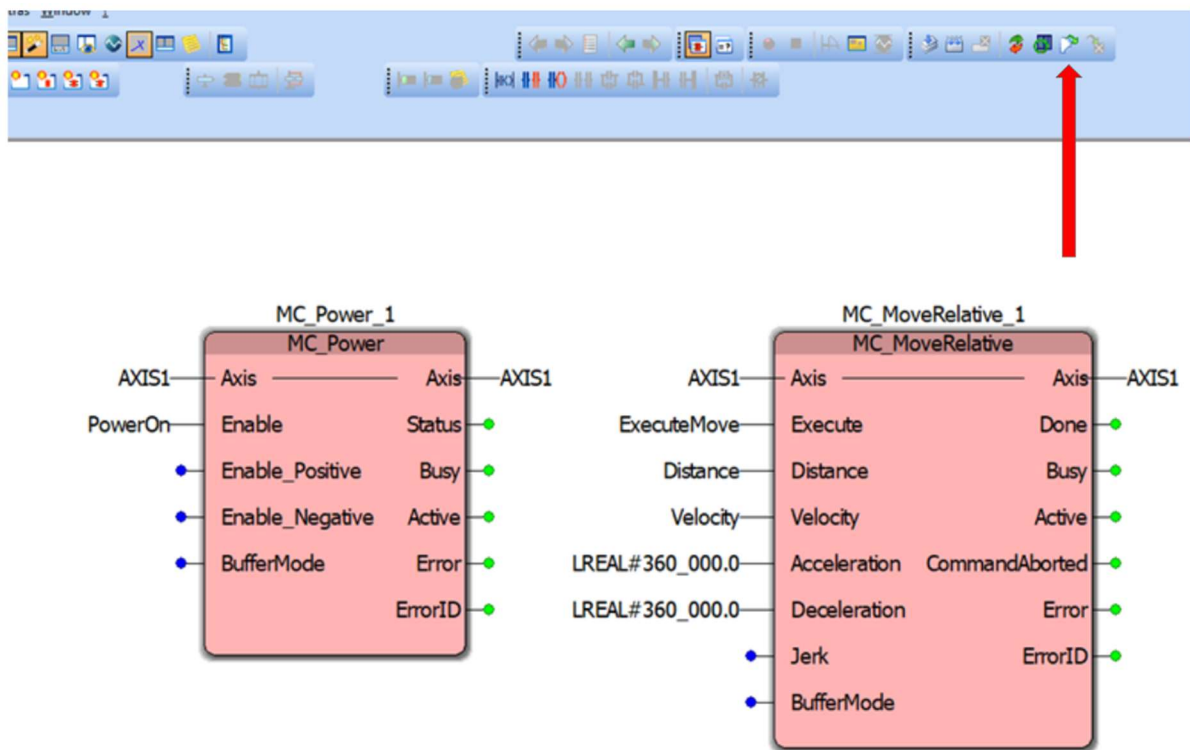


Figure 12: MC_Power + MC_MoveRelative Final Product and Arrow to 'Download Changes'

- After a few seconds the window on the left of Figure 13 should appear, hit Download and the window on the right of Figure 13 should appear. Hit Download again.
- After Downloading, the window on the left of Figure 14 should appear; hit warm and the window should change to look like the window on the right of Figure 14 and the program is ready to run.

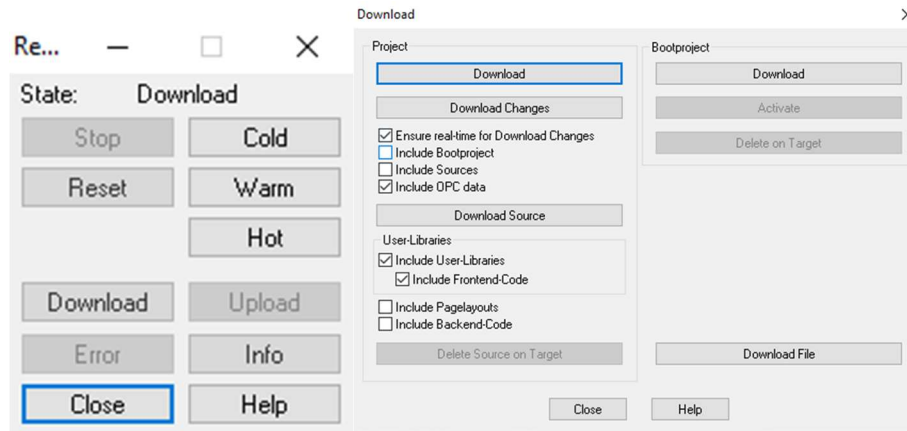


Figure 13: Download and Run Series 1

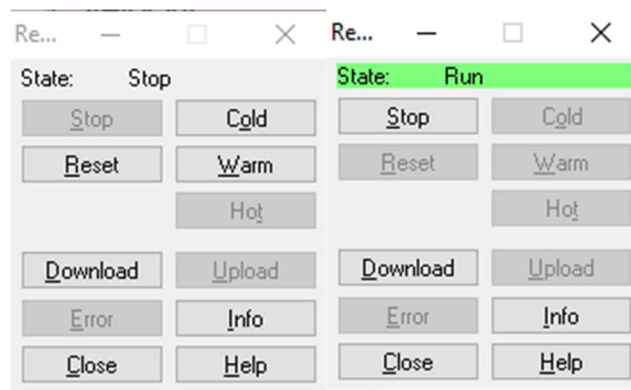


Figure 14: Download and Run Series 2

- Enter Debug Mode clicking the Debug Toggle as shown in Figure 15 and it should look like Figure 16

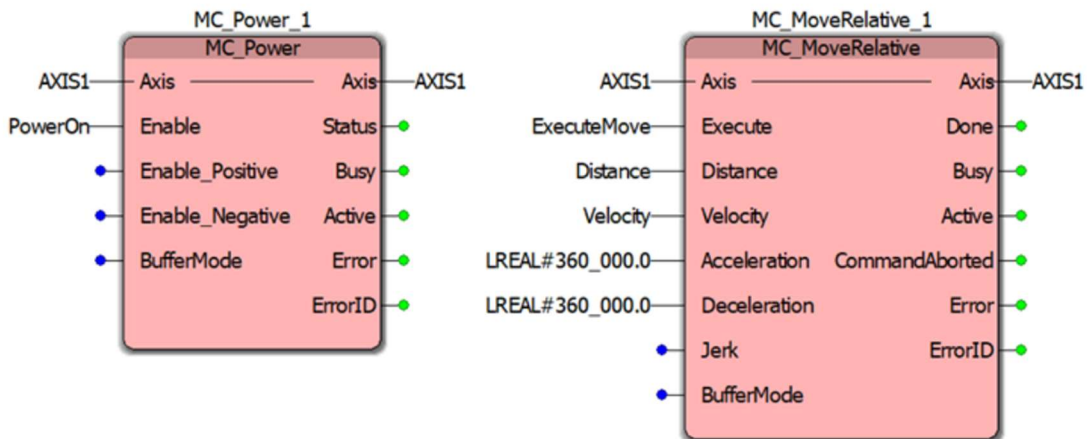
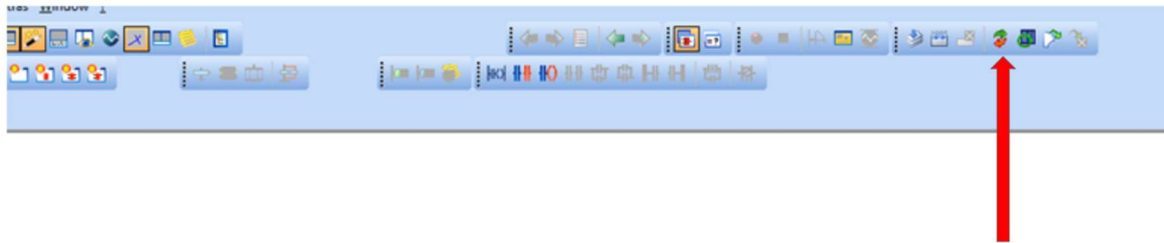


Figure 15: Program with Arrow to Debug Mode

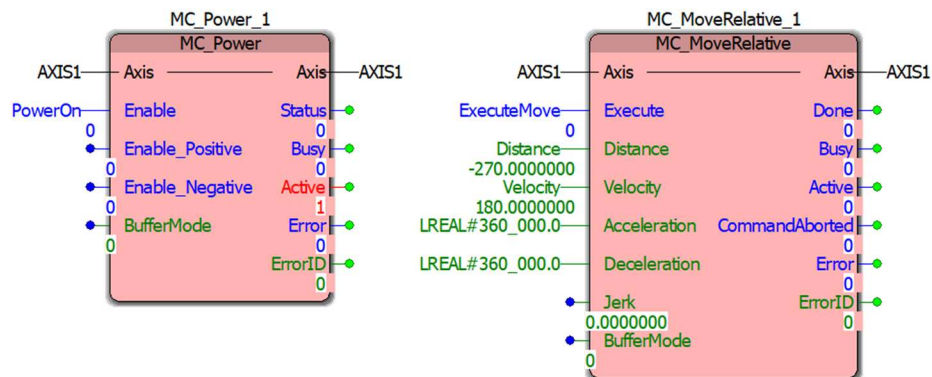


Figure 16: Step 1 in Debug Mode

- Double-click 'PowerOn' and once in the window in Figure 17, hit Overwrite to change the value so that the blocks look like they do in Figure 18.
- This enables the motor axis, and you can check that the motor is on because you should not be able to move it.

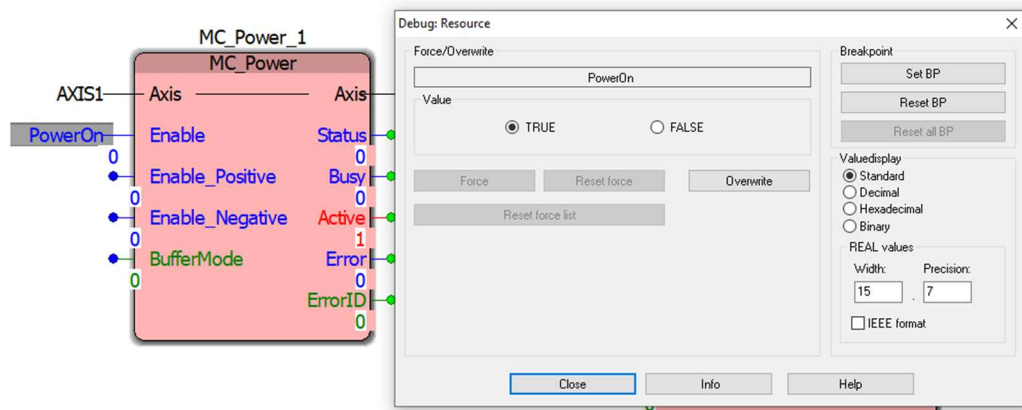


Figure 17: Enabling MC_Power

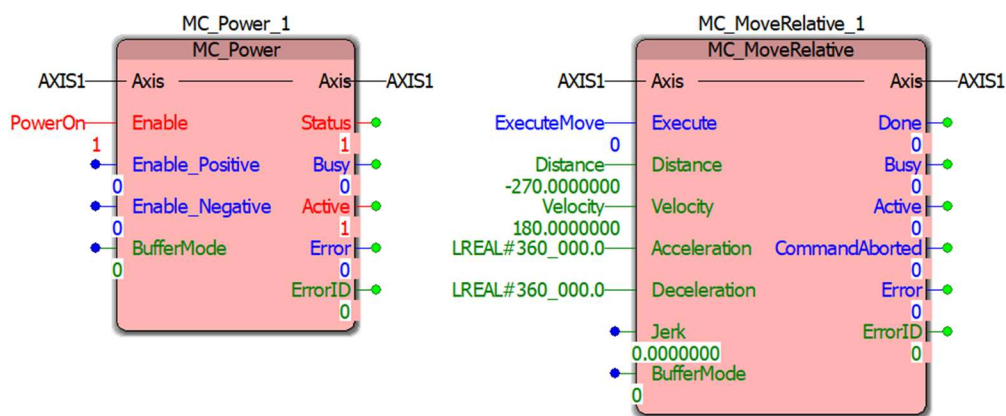


Figure 18: MC_Power Enabled

- Now to move the motor, double-click 'ExecuteMove' and hit Overwrite to change the value as shown in Figure 19.
- This should move the motor 270° every time execute move moves from False to True.

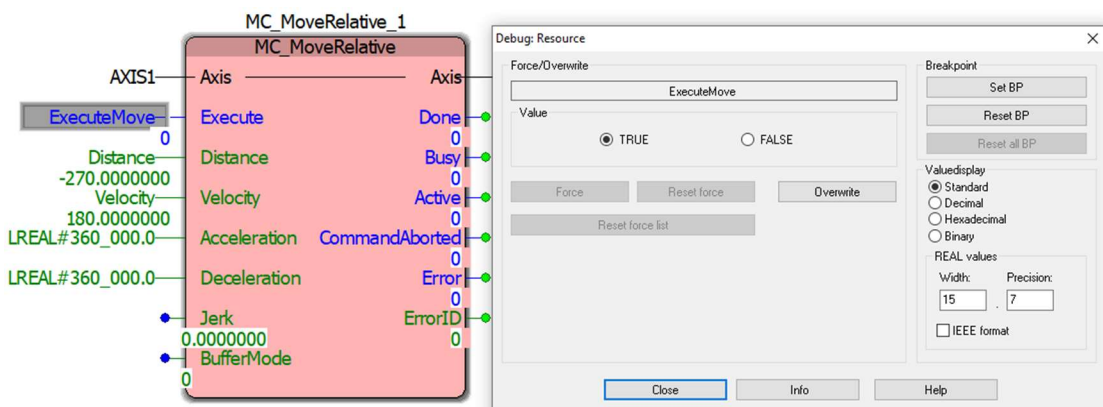


Figure 19: Executing MC_MoveRelative

- Once ready to move on the next step, Exit Debugging Mode

Step 2: Read Position with Code

- The motors also have an encoder that can read the absolute position of the motor, to use it create an MC_ReadActualPosition block and set its Axis to the same Axis in the previous step.
- On the Enable Input Variable, make the name 'TRUE' as seen in Figure 20 to read the position at all times.

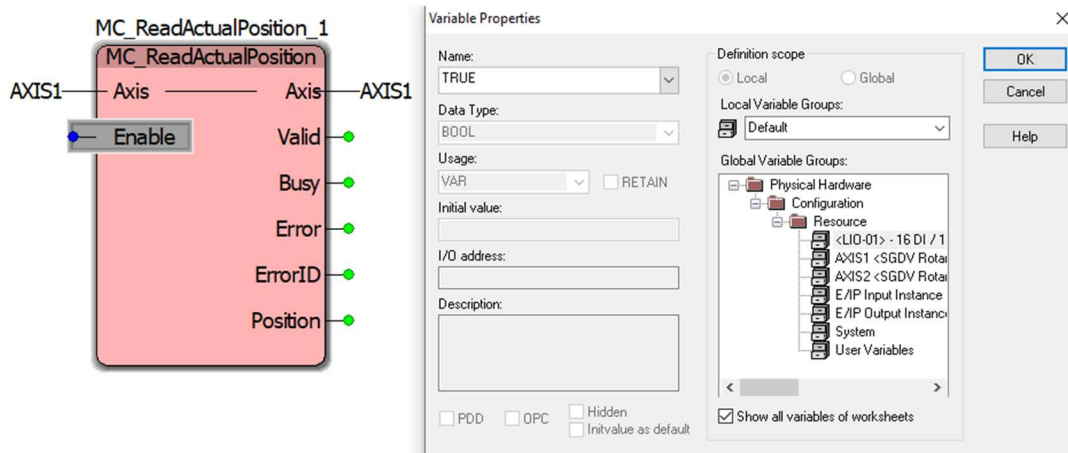


Figure 20: MC_ReadActualPosition Enable Properties

- Download Changes, Warm Run the program, and switch to Debugging Mode like in Step 1 and it should resemble Figure 21 with the Position shown at the bottom left of the block in degrees

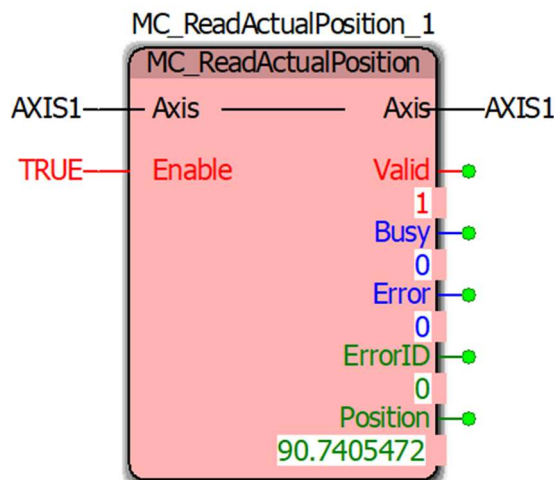


Figure 21: MC_ReadActualPosition in Debug Mode

Step 3: Relative Move into Absolute Move

- Now lets do 2 moves in sequence, add an MC_MoveAbsolute block and position it such that it connects to the MC_MoveRelative block as shown in Figure 22.
- Configure the Position to match Figure 13 and then use the same Velocity and Acceleration/Deceleration settings as MC_MoveRelative, they can be copied over by Ctrl+clicking and dragging the variable to make a copy and inserting it where needed.

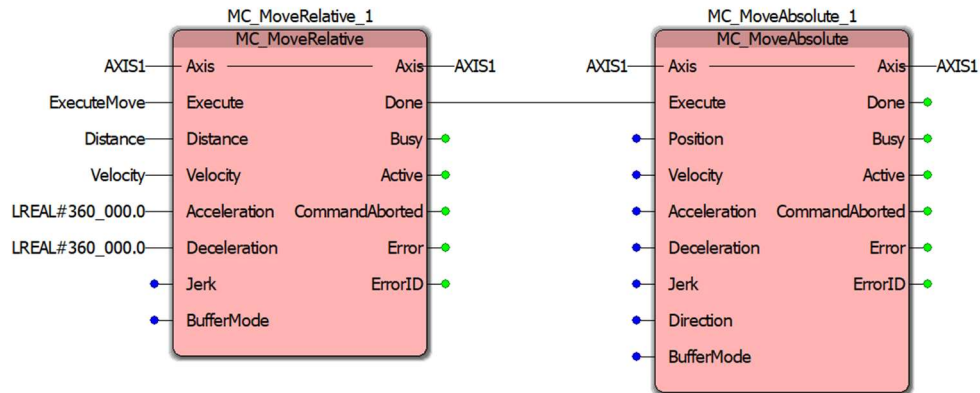


Figure 22: MC_MoveRelative and MC_MoveAbsolute Linked in Series

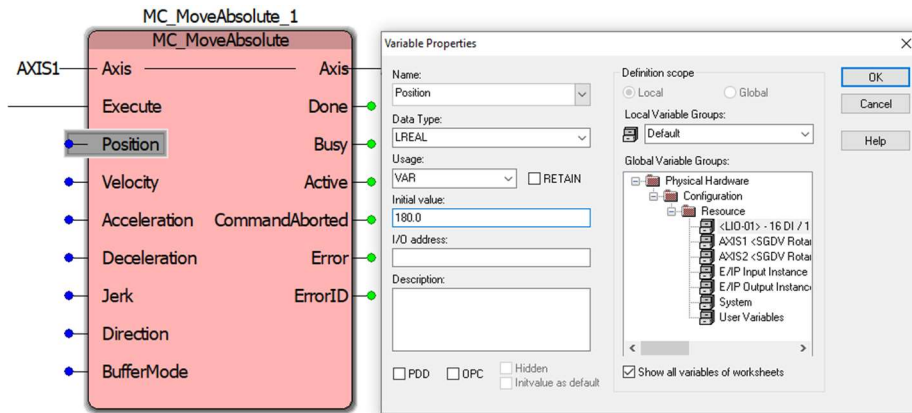


Figure 23: MC_MoveAbsolute Position Properties

- The final product should look like Figure 24 and when Executed correctly, it should move twice while reading the position.

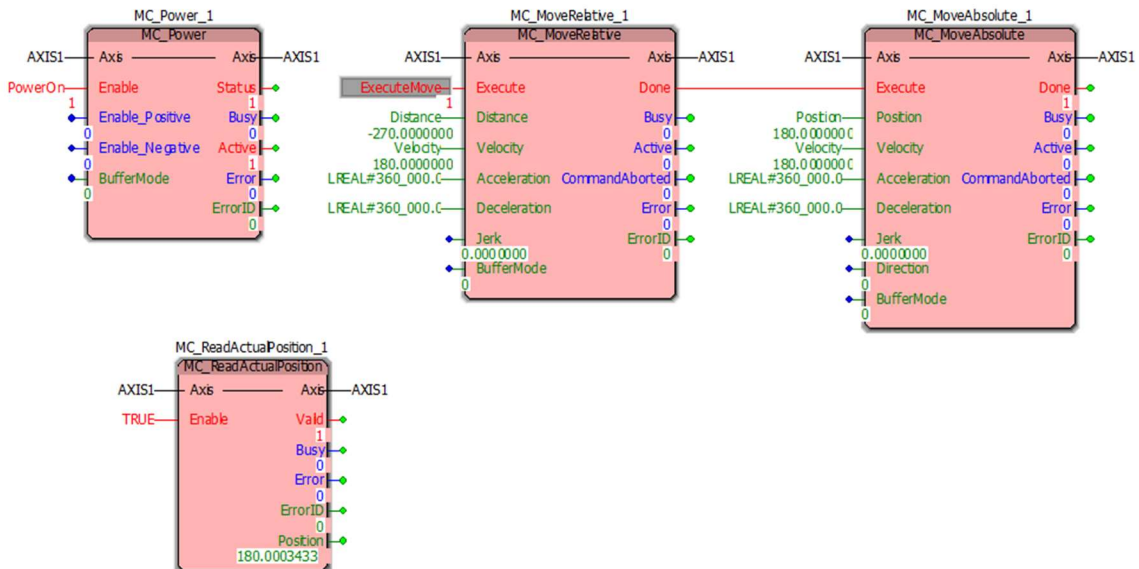


Figure 24: Step 3 Final Product in Debug Mode

Challenge: Electronic Gearing

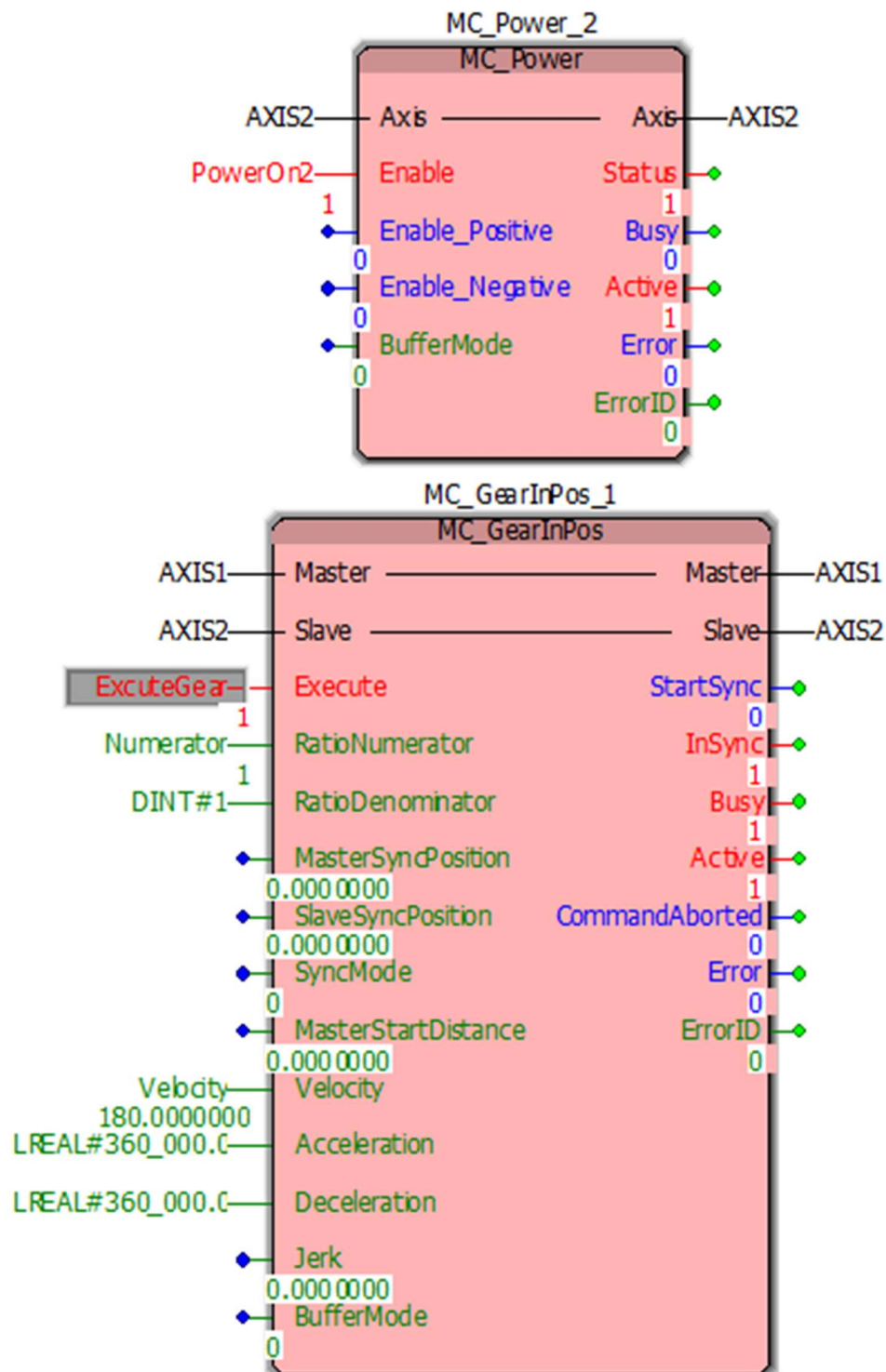
Electronic gearing is a method of mimicking the behavior of 2 gears meshed together by synchronizing the movement of 2 motors ([Example](#)). One of the main benefits of electronic gearing is the ability to change gear ratios easily. This challenge will have you code:

1. Gears that move 1:1
2. Gears that move 1:-1
3. Gears that move 1:2

Start with the MC_GearInPos Block and use documentation and knowledge from the lab to complete and execute the block(s) necessary.

Deliverables

Check off each case of the challenge with the lab instructor.

Solution:

Comment: Cycle through different cases by Overwriting value in RatioNumerator and re-Executing the function block. If not re-Executed, program will continue to run previous ratio.