Robotics, Integration, and Automation

PLC-OPERATED ROBOT JOBS

Name	Class/Period	Date

1. Overview

In this lab activity, you will use a simple ladder logic routine to initiate and monitor the status of a robot job.

> ***Note:*** *In order to differentiate between the programmable logic controller and the robot controller, the two devices are referred to as the PLC and the robot, respectively.*

2. Performance Objectives

After completing this lab activity, you will be able to:

- Identify the robot's input and output groups where data exchange between the robot and the PLC occurs.
- Exchange data between the PLC and the robot.
- Add WAIT and DOUT instructions to a robot job.
- Address ladder diagram instructions to robot module tags.
- Initiate robot jobs and job commands by toggling the values of tags in an online ladder diagram.

3. Required Materials

You need the following materials to complete the lab activity:

- SmartCart 4.0
- Computer
- Ethernet cables

4. Required Software

Logix Designer is required for this lab activity. It is included in the Studio 5000 suite. Ensure that the software is installed on your PC and has a valid license. If you are having problems installing or licensing the software, contact your instructor or IT manager.

5. Inventory and Safety

Before beginning the lab activity, review this checklist and mark off each item as you complete it.

- ☐ All hardware components are available for this lab activity.
- ☐ Hands, hair, and clothing are securely away from the work area.
- ☐ The work area is clean and devoid of food or drink.
- ☐ Review the SmartCart safety guidelines.
- ☐ Read through the entirety of this lab activity to familiarize yourself with the requirements.

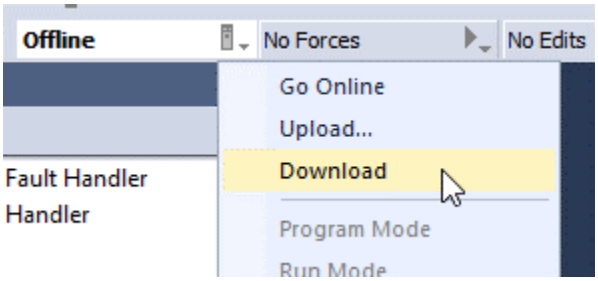
6. Lab Activity

6.1. Downloading and Confirming the Connection

In this task, you will go online with the PLC and ensure that the robot is connected.

Perform these steps.

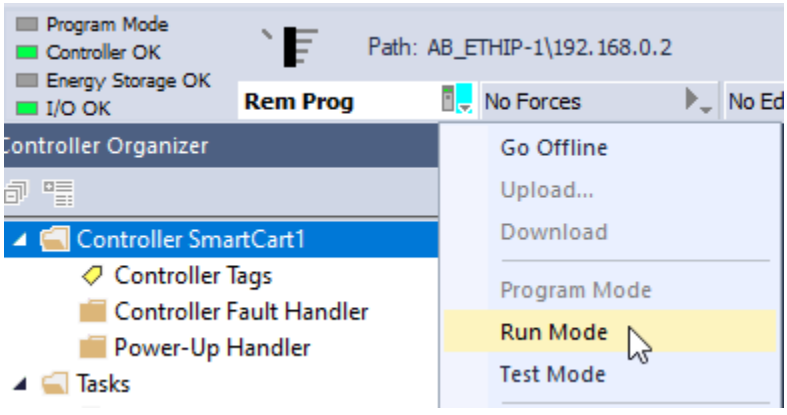
1. Power up the I/O box. Wait for the PLC to power on.
2. Power up the robot. Ensure that the mode key on the pendant is set to Teach mode.
3. Open your PLC programing project from the previous lab activity.
4. Download the project to the PLC.



5. In the PLC status area, ensure that the **I/O OK** LED is on.

ⓘ ***Note:*** *If the PLC status area shows an* ***I/O Not Responding*** *message, troubleshoot the project's module configuration, the network connection to the robot, and the robot's EtherNet/IP board configuration.*

6. Switch to **Rem Run** mode.



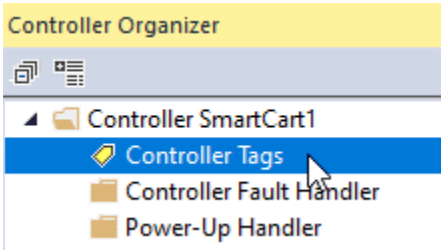
7. Confirm that the I/O is still OK.

6.2. Identifying Robot Input Groups for Data Exchange

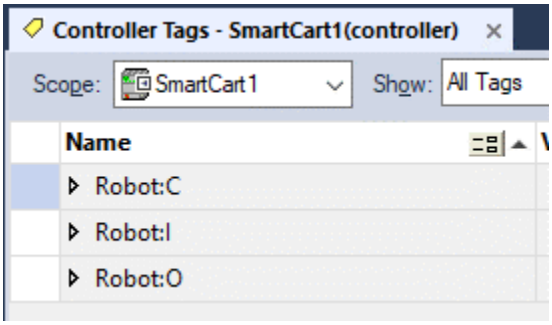
In this task, you will turn on various module output tags in order to identify which robot input groups and bits are used for data exchange with the PLC.

Perform the following steps:

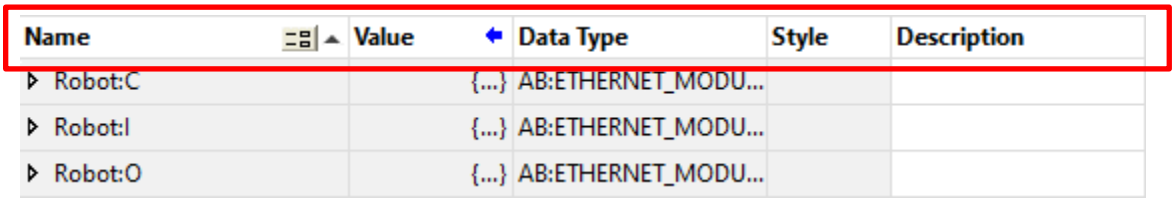
1. In the Controller Organizer, double-click **Controller Tags**.



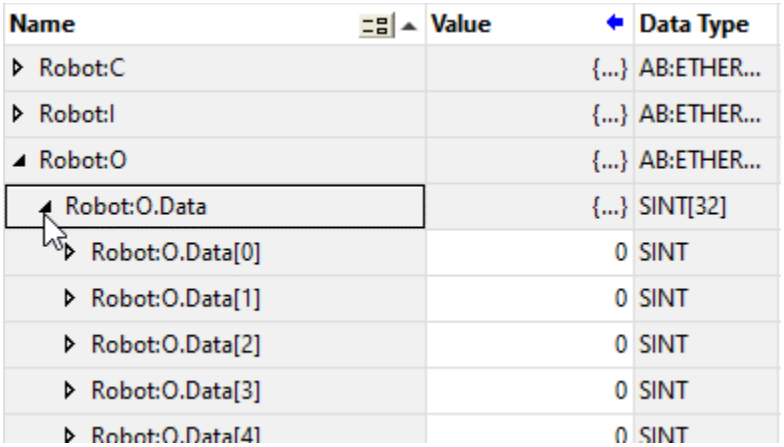
2. The data monitor is displayed. Currently, the only tags are the robot module tags. They are divided into configuration (C), input (I), and output (O). Note that our tags are named “Robot” because that is what we named the module when we created it. Your tags would have different names if you named the module differently.



3. Drag/toggle the columns to display the Value, Data Type, Style, and Description columns next to the Name column.



4. Expand the robot output data tags.

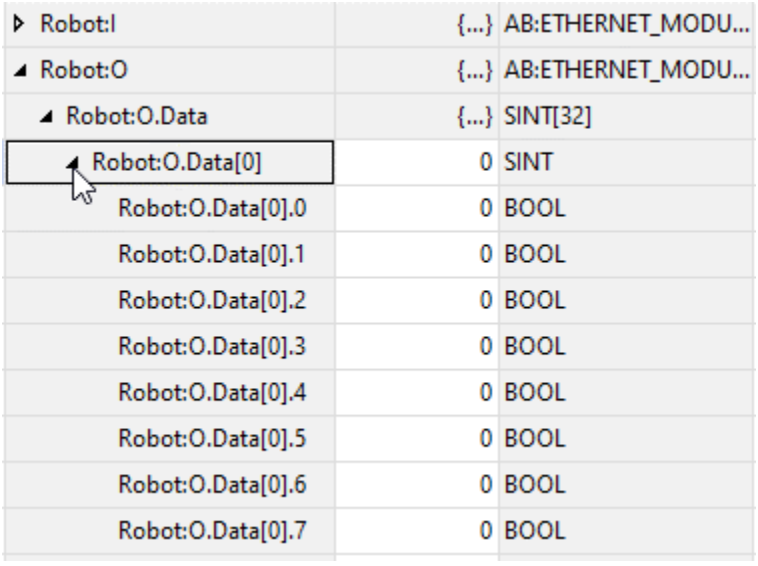


Name	Value	Data Type
Robot:C	{...}	AB:ETHER...
Robot:I	{...}	AB:ETHER...
Robot:O	{...}	AB:ETHER...
Robot:O.Data	{...}	SINT[32]
Robot:O.Data[0]	0	SINT
Robot:O.Data[1]	0	SINT
Robot:O.Data[2]	0	SINT
Robot:O.Data[3]	0	SINT
Robot:O.Data[4]	0	SINT

Here's what you need to know about these tags:

- There are 32 SINT (single integer) module *output* tags, as you defined when creating the module in the previous lab activity.
- Each output SINT is a byte of data, and each output SINT corresponds to an input group (IG) on the robot. (Recall that PLC output is input from the point of view of the robot.)
- Each SINT can be further expanded into 8 BOOL tags as shown in the image below.
- Each BOOL tag corresponds to a single bit in the SINT's corresponding robot IG.

5. Expand **Robot:O.Data[0]** to display the SINT tag's composite BOOL tags.

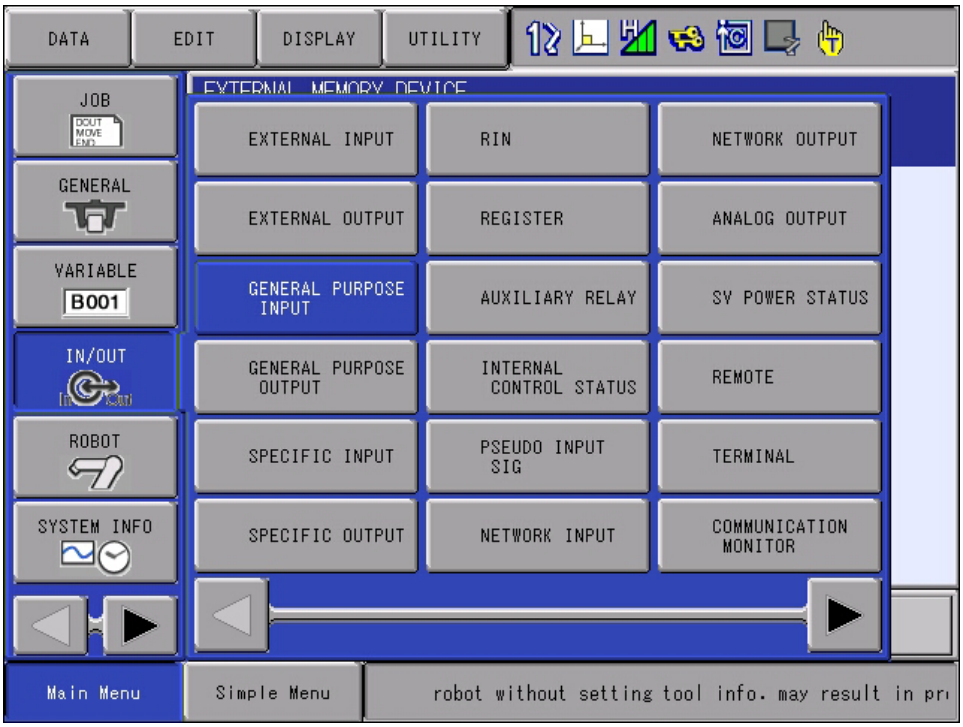


Robot:I	{...}	AB:ETHERNET_MODU...
Robot:O	{...}	AB:ETHERNET_MODU...
Robot:O.Data	{...}	SINT[32]
Robot:O.Data[0]	0	SINT
Robot:O.Data[0].0	0	BOOL
Robot:O.Data[0].1	0	BOOL
Robot:O.Data[0].2	0	BOOL
Robot:O.Data[0].3	0	BOOL
Robot:O.Data[0].4	0	BOOL
Robot:O.Data[0].5	0	BOOL
Robot:O.Data[0].6	0	BOOL
Robot:O.Data[0].7	0	BOOL

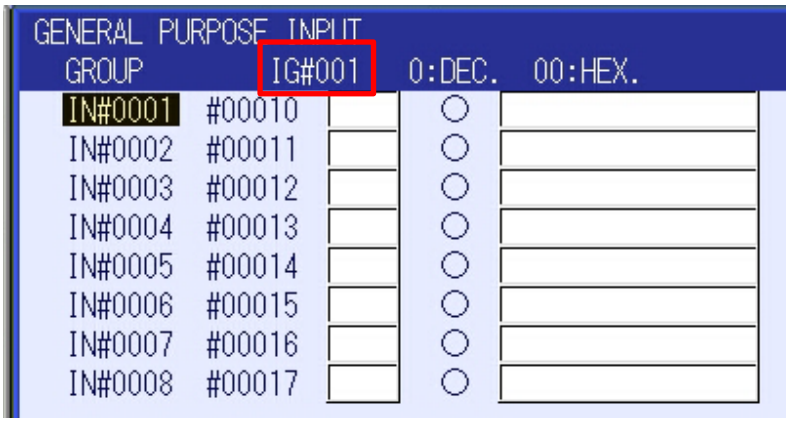
6. Change the values of several of the BOOL tags to **1**. Note the value of Robot:O.Data[0].

◢ Robot:O.Data[0]	52	SINT
Robot:O.Data[0].0	0	BOOL
Robot:O.Data[0].1	0	BOOL
Robot:O.Data[0].2	1	BOOL
Robot:O.Data[0].3	0	BOOL
Robot:O.Data[0].4	1	BOOL
Robot:O.Data[0].5	1	BOOL
Robot:O.Data[0].6	0	BOOL
Robot:O.Data[0].7	0	BOOL

7. On the robot's programming pendant, navigate to **IN/OUT > GENERAL PURPOSE INPUT**.

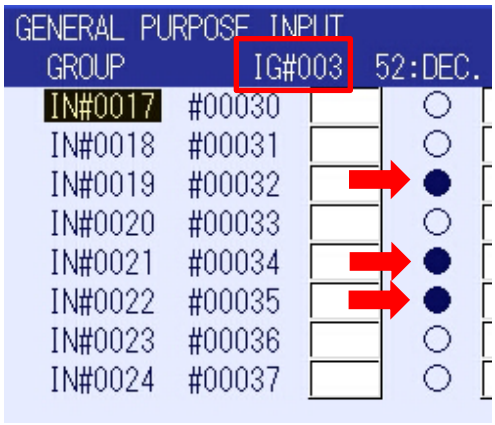


Initially, IG#001 is displayed. If there is nothing activating any of the digital sensors or microswitches, all bits should be off, and the value of the IG is 0.



8. Press the pendant's PAGE key until **IG#003** is displayed. You should see that the value of the IG is equal to the value of the SINT tag, and the values of the bits are equal to the values of the BOOL tags in the SINT tag. (The value 1 signifies that the BOOL tag is on. A darkened circle icon indicates that the bit is on.)

◢ Robot:O.Data[0]	52	SINT
Robot:O.Data[0].0	0	BOOL
Robot:O.Data[0].1	0	BOOL
Robot:O.Data[0].2	1	BOOL
Robot:O.Data[0].3	0	BOOL
Robot:O.Data[0].4	1	BOOL
Robot:O.Data[0].5	1	BOOL
Robot:O.Data[0].6	0	BOOL
Robot:O.Data[0].7	0	BOOL



You can see that Robot:O.Data[0] corresponds to IG#003, and the BOOL tags of Robot:O.Data[0] correspond to the bits of IG#003 - Robot:O.Data[0].0 corresponds to IN#0017, Robot:O.Data[0].1 corresponds to IN#0018, and so on.

ⓘ ***Note:*** *BOOL tag 7 (the highest BOOL tag) of each SINT tag changes the sign (plus or minus) of the SINT value. However, the highest bit of each robot input group does not (it adds 128 to the value, or 2^7).*

ⓘ ***Note:*** *If the SINT whose values you modified does not correspond to IG#003, continue pressing PAGE until you find the corresponding input group. If you cannot find the matching input group, troubleshoot your system configuration.*

ⓘ ***Note:*** *To move to a lower input group, press SHIFT + PAGE.*

9. Change the value of the SINT tag back to **0**. The values of all the BOOL tags in the SINT return to 0. The values of IG#003 also revert back to 0.

◢ Robot:O.Data[0]	0	SINT	Decimal
Robot:O.Data[0].0	0	BOOL	Decimal
Robot:O.Data[0].1	0	BOOL	Decimal
Robot:O.Data[0].2	0	BOOL	Decimal
Robot:O.Data[0].3	0	BOOL	Decimal
Robot:O.Data[0].4	0	BOOL	Decimal
Robot:O.Data[0].5	0	BOOL	Decimal
Robot:O.Data[0].6	0	BOOL	Decimal
Robot:O.Data[0].7	0	BOOL	Decimal

10. Change the values of several other output SINTs to find their corresponding IGs. Note that Robot:O.Data[1] corresponds to IG#004, Robot:O.Data[2] corresponds to IG#005, and so on. Remember that first two IGs are used for the digital input devices that are connected to the I/O box.

11. When you are finished, return all SINT values to **0**.

12. It is good practice to document your project. Label Robot:O.Data[0] by filling in a description for the SINT and BOOLs.

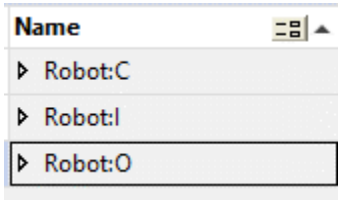
Name	Value	Data Type	Style	Description
◢ Robot:O.Data	{...}	SINT[32]	Decimal	
◢ Robot:O.Data[0]	0	SINT	Decimal	OG#003
Robot:O.Data[0].0	0	BOOL	Decimal	IN#0017
Robot:O.Data[0].1	0	BOOL	Decimal	IN#0018
Robot:O.Data[0].2	0	BOOL	Decimal	IN#0019
Robot:O.Data[0].3	0	BOOL	Decimal	IN#0020
Robot:O.Data[0].4	0	BOOL	Decimal	IN#0021
Robot:O.Data[0].5	0	BOOL	Decimal	IN#0022
Robot:O.Data[0].6	0	BOOL	Decimal	IN#0023
Robot:O.Data[0].7	0	BOOL	Decimal	IN#0024

6.3. Identifying Robot Output Groups for Data Exchange

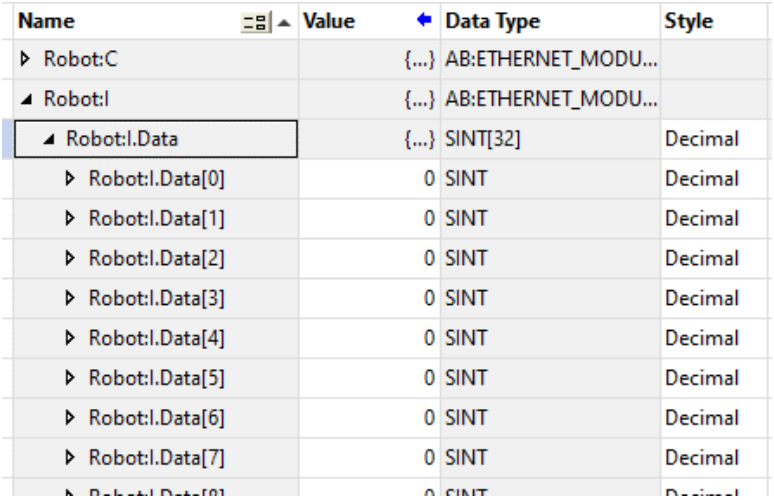
In this task, you will turn on bits in robot output groups (OGs) in order to see which are used for data exchange with the PLC.

Perform these steps:

1. Collapse the module output tags.



2. Expand the input tags.



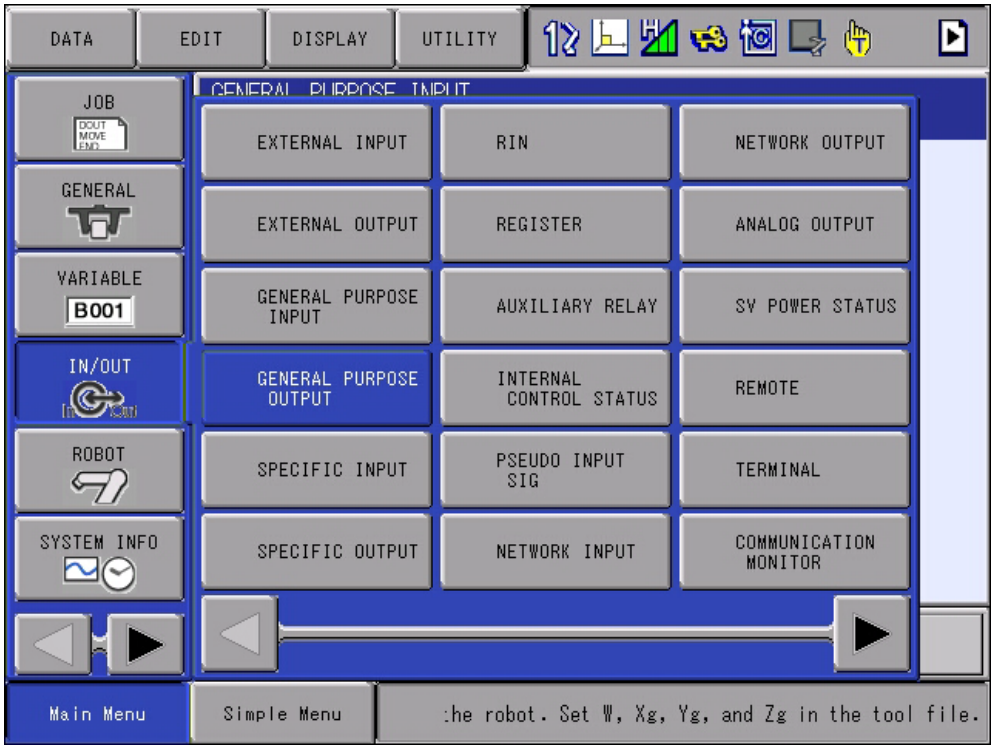
These module input tags work in a parallel but opposite way to the module output tags:

- There are 32 SINT (single integer) module *input* tags, as you defined when creating the module in the previous lab activity.
- Each input SINT is a byte of data, and each input SINT corresponds to an output group (OG) on the robot. (Robot output is input from the point of view of the PLC.)
- Each SINT can be further expanded into 8 BOOL tags.
- Each BOOL tag corresponds to a single bit in the SINT's corresponding robot OG.

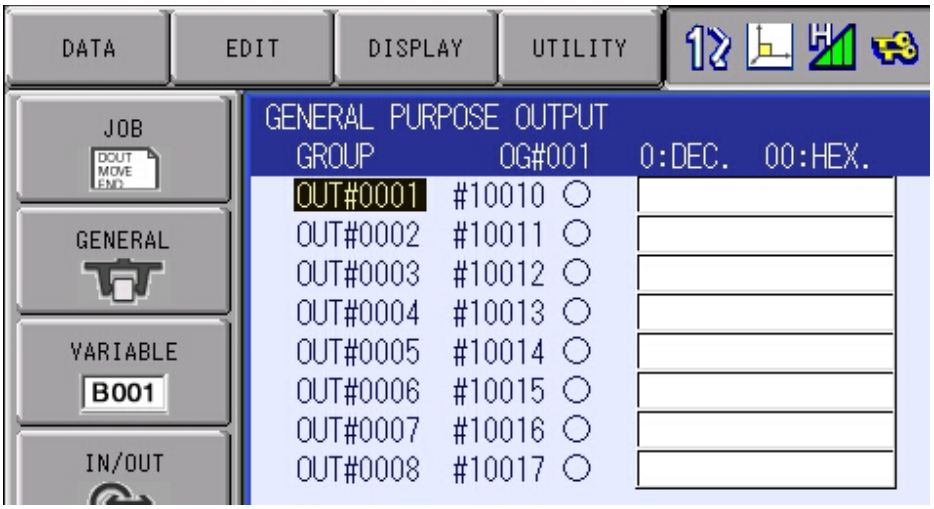
3. Expand the **Robot:I.Data[0]** SINT tag to display the composite BOOL tags.

◢ Robot:I	{...}	AB:ETHE...
◢ Robot:I.Data	{...}	SINT[32]
◢ Robot:I.Data[0]	0	SINT
Robot:I.Data[0].0	0	BOOL
Robot:I.Data[0].1	0	BOOL
Robot:I.Data[0].2	0	BOOL
Robot:I.Data[0].3	0	BOOL
Robot:I.Data[0].4	0	BOOL
Robot:I.Data[0].5	0	BOOL
Robot:I.Data[0].6	0	BOOL
Robot:I.Data[0].7	0	BOOL

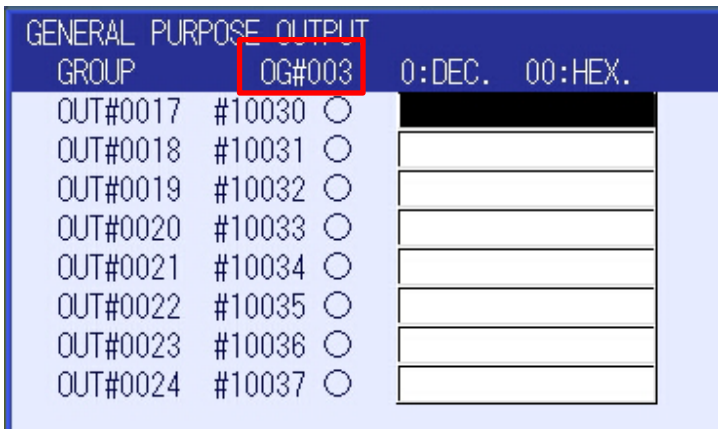
4. On the pendant, navigate to **IN/OUT > GENERAL PURPOSE OUTPUT**.



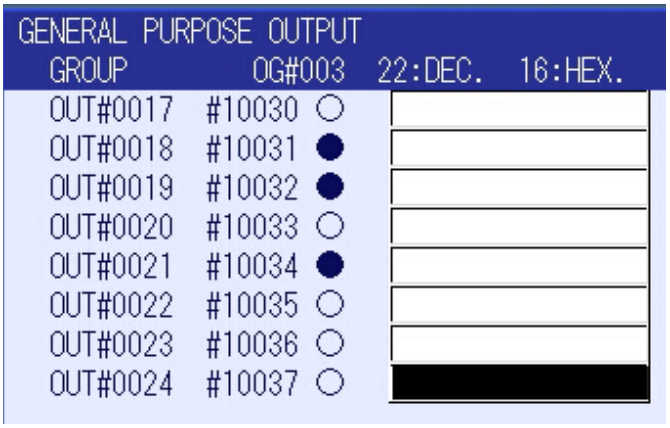
5. OG#001 is initially displayed.



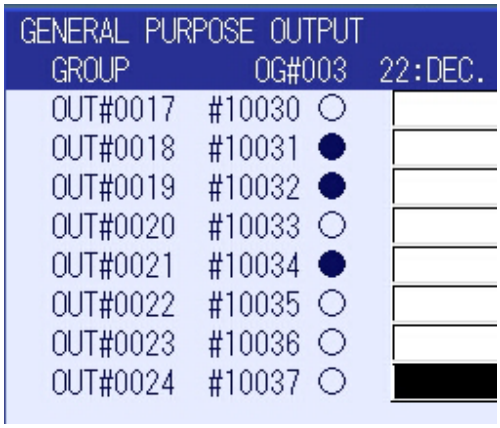
6. Press PAGE until OG#003 is displayed.



7. Change the values of (turn on) several of the bits. To turn a bit on or off, move the cursor to the bit's circle icon and then press **INTERLOCK + SELECT**. In the example below, OUT#0018, OUT#0019, and OUT#0021 have been turned on.



8. Observe the values of the BOOL tags of Robot:I.Data[0]. Note how the BOOL tags with a value of 1 match the OG#003 bits that have been turned on.



Robot:I.Data[0]	22	SINT
Robot:I.Data[0].0	0	BOOL
Robot:I.Data[0].1	1	BOOL
Robot:I.Data[0].2	1	BOOL
Robot:I.Data[0].3	0	BOOL
Robot:I.Data[0].4	1	BOOL
Robot:I.Data[0].5	0	BOOL
Robot:I.Data[0].6	0	BOOL
Robot:I.Data[0].7	0	BOOL

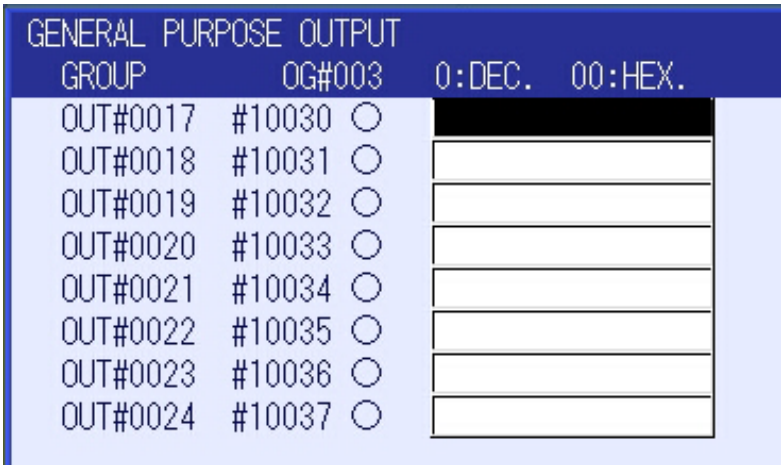
You can see that OG#003 corresponds to Robot:I.Data[0], and the bits in OG#003 correspond to the BOOL tags of Robot:I.Data[0] - OUT#0017 corresponds to Robot:I.Data[0].0, OUT#0018 corresponds to Robot:O.Data[0].1, and so on.

ⓘ ***Note:*** *BOOL tag 7 (the highest BOOL tag) of each SINT tag changes the sign (plus or minus) of the SINT value. However, the highest bit of each robot output group does not (it adds 128 to the value, or 2^7, when on).*

ⓘ ***Note:*** *If the OG whose values you modified does not correspond to Robot:I.Data[0], continue pressing PAGE and modifying the values of bits until you find the corresponding output group for Robot:I.Data[0]. If you cannot find the matching output group, troubleshoot your system configuration.*

ⓘ ***Note:*** *To move to a lower output group, press SHIFT + PAGE.*

9. Turn off all output bits (return all output bit values to 0).



10. Change the values of various bits on several other OGs and observe the changes in module input tag values. Note that Robot:I.Data[1] corresponds to OG#004, Robot:I.Data[2] corresponds to OG#005, and so on. The first two OGs are used for the digital output devices that are connected to the I/O box.

11. When you are finished, return all bits to **0**.

12. Label Robot:I.Data[0] by filling in a description for the SINT and BOOLs.

Name	Value	Data Type	Style	Description
Robot:I.Data[0]	0	SINT	Decimal	OG#003
Robot:I.Data[0].0	0	BOOL	Decimal	OUT#0017
Robot:I.Data[0].1	0	BOOL	Decimal	OUT#0018
Robot:I.Data[0].2	0	BOOL	Decimal	OUT#0019
Robot:I.Data[0].3	0	BOOL	Decimal	OUT#0020
Robot:I.Data[0].4	0	BOOL	Decimal	OUT#0021
Robot:I.Data[0].5	0	BOOL	Decimal	OUT#0022
Robot:I.Data[0].6	0	BOOL	Decimal	OUT#0023
Robot:I.Data[0].7	0	BOOL	Decimal	OUT#0024

6.4. Adding DOUT and WAIT Instructions to a Robot Job

The objective of the next two tasks is to create a PLC control program for initiation and monitoring of a robot job. In this task, you will add commands to a robot job that will allow the job to receive data from the PLC program and to send data to it.

1. Ensure that the mode key is set to Teach mode.
2. On the pendant, navigate to **JOB > CREATE NEW JOB**.
3. Create a simple motion or pick and place job. The instructions shown below represent a generic motion job.

```
0000 NOP
0001 MOVJ P000 VJ=10.00
0002 MOVL P001 V=60.0
0003 MOVL P002 V=60.0
0004 MOVJ P000 VJ=10.00
0005 END
```

4. Use the INFORM LIST to *insert* a wait command at the beginning of the job (**INFORM LIST > IN/OUT > WAIT**). For the instruction parameters, set **IN#(17)** to be **ON**. (Recall that IN#0017 corresponds to Robot:**O**.Data[0].0.) This command will force the robot to wait until Robot:O.Data[0].0. turns on before starting the motion commands of the job.

```
0000 NOP
0001 WAIT IN#(17)=ON
0002 MOVJ P000 VJ=10.00
0003 MOVL P001 V=60.0
0004 MOVL P002 V=60.0
0005 MOVJ P000 VJ=10.00
0006 END
```

5. At the end of the job, add a DOUT command (**INFORM LIST > IN/OUT > DOUT**) to set **OUT#(17)** to be **ON**. Recall that OUT#0017 corresponds to Robot:**I**.Data[0].0. This new instruction will notify the PLC that the job is complete.

```
0000 NOP
0001 WAIT IN#(17)=ON
0002 MOVJ P000 VJ=10.00
0003 MOVL P001 V=60.0
0004 MOVL P002 V=60.0
0005 MOVJ P000 VJ=10.00
0006 DOUT OT#(17) ON
0007 END
```

6. Ideally, there should be a way for OT to turn off. Otherwise, OUT#0017 will stay on, and the PLC will think that the job is complete, even if it is initiated again and running. Therefore, add two more instructions, a 10 second timer (INFORM LIST > CONTROL > TIMER) and a DOUT that turns OUT#0017 off.

```
0000 NOP
0001 WAIT IN#(17)=ON
0002 MOVJ P000 VJ=10.00
0003 MOVL P001 V=60.0
0004 MOVL P002 V=60.0
0005 MOVJ P000 VJ=10.00
0006 DOUT OT#(17) ON
0007 TIMER T=10.000
0008 DOUT OT#(17) OFF
0009 END
```

7. Finally, add instructions that allow the job to run on a loop (if enabled by the PLC). Insert a label (INFORM LIST > CONTROL > LABEL) at the top of the program, and a command to jump to the label at the end.

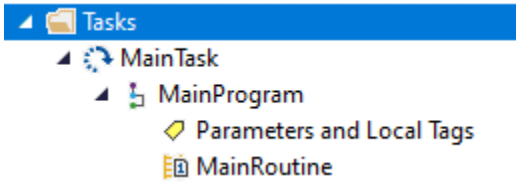
```
0000 NOP
0001 *START
0002 WAIT IN#(17)=ON
0003 MOVJ P000 VJ=10.00
0004 MOVL P001 V=60.0
0005 MOVL P002 V=60.0
0006 MOVJ P000 VJ=10.00
0007 DOUT OT#(17) ON
0008 TIMER T=10.000
0009 DOUT OT#(17) OFF
0010 JUMP *START
0011 END
```

6.5. Creating the Logic Routine

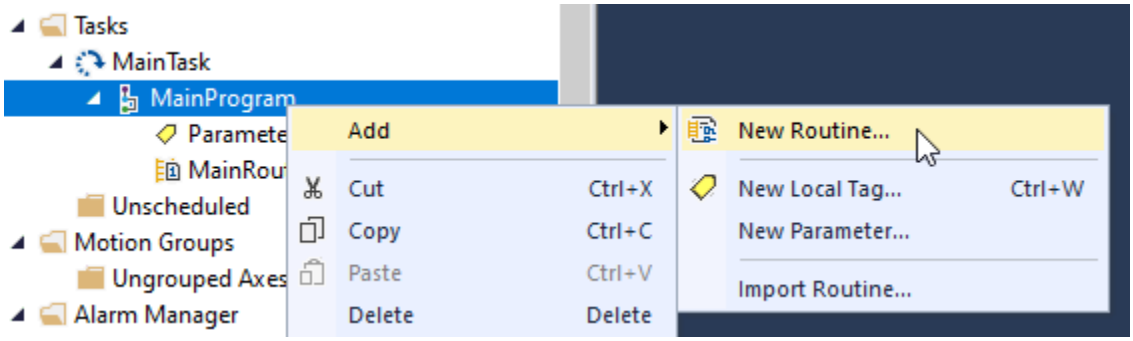
In this task, you will write the PLC logic for job initiation and monitoring.

Perform the following steps:

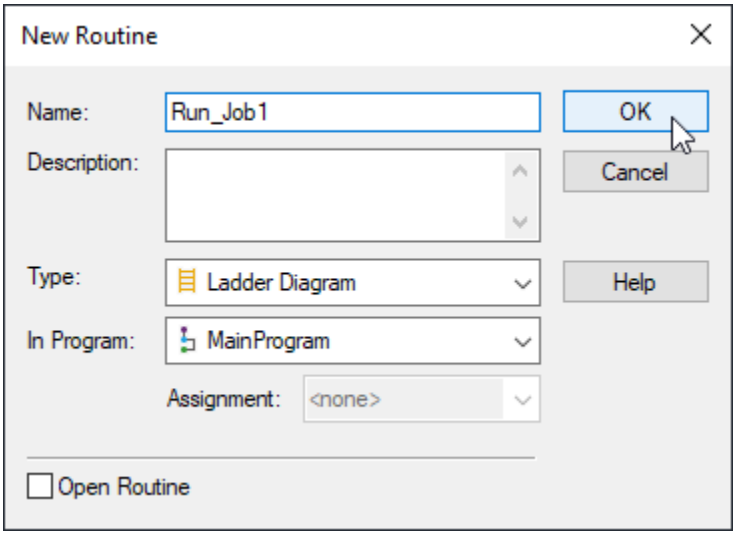
1. In the online Logix Designer project's Controller Organizer, expand all of the elements of the **Tasks** folder.



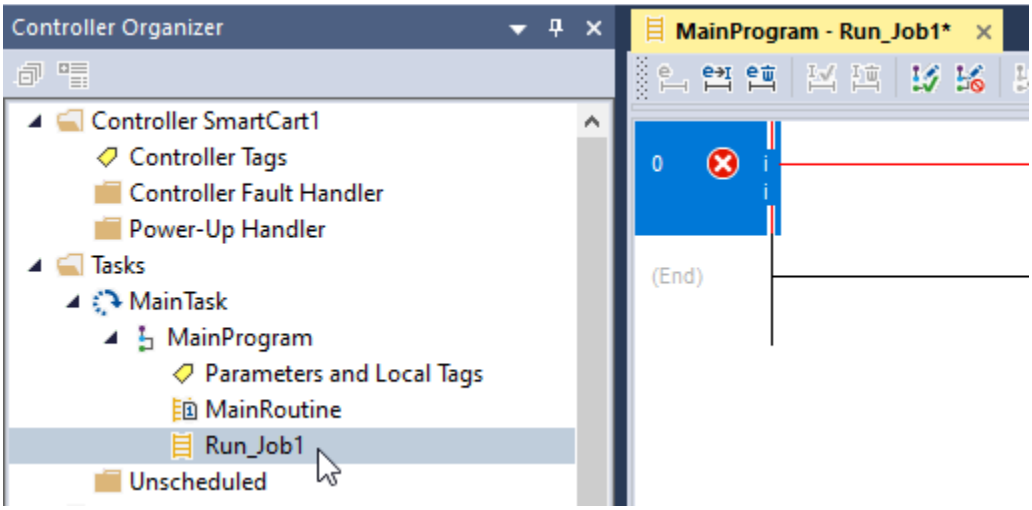
2. Right-click **MainProgram** and add a new routine.



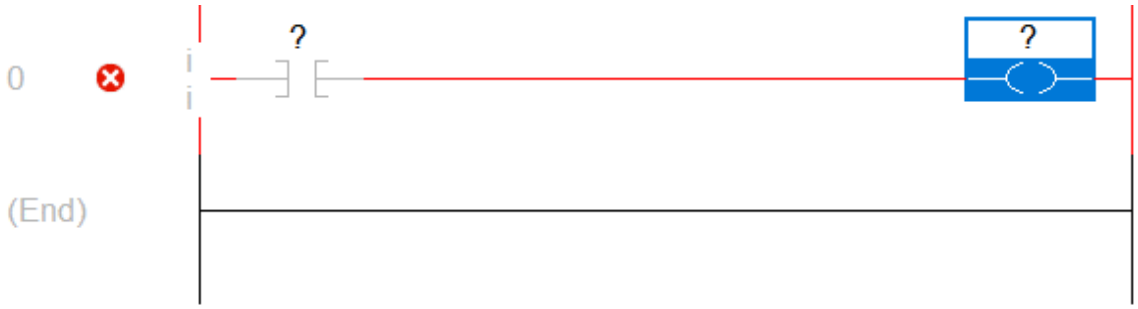
3. Name the routine Run_Job1, or however you've named the job, and then click **OK**.



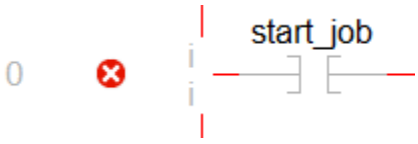
4. The routine is added to the MainProgram. Double-click it to open its ladder editor.



5. From the instruction library, add an **Examine On (XIC)** instruction and an **Output Energize (OTE)** instruction to rung 0.

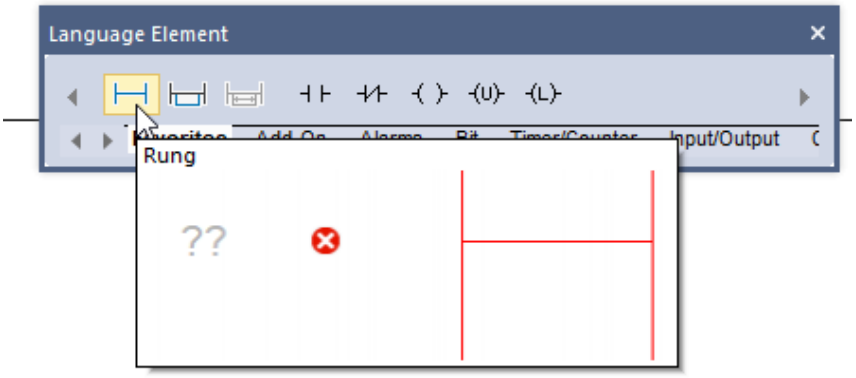


6. Address the XIC to a new BOOL tag called **start_job** or similar. You can create new tags in the Tag Editor (Logic > Edit Tags), or by selecting the XIC instruction and then pressing Ctrl + W on your keyboard.

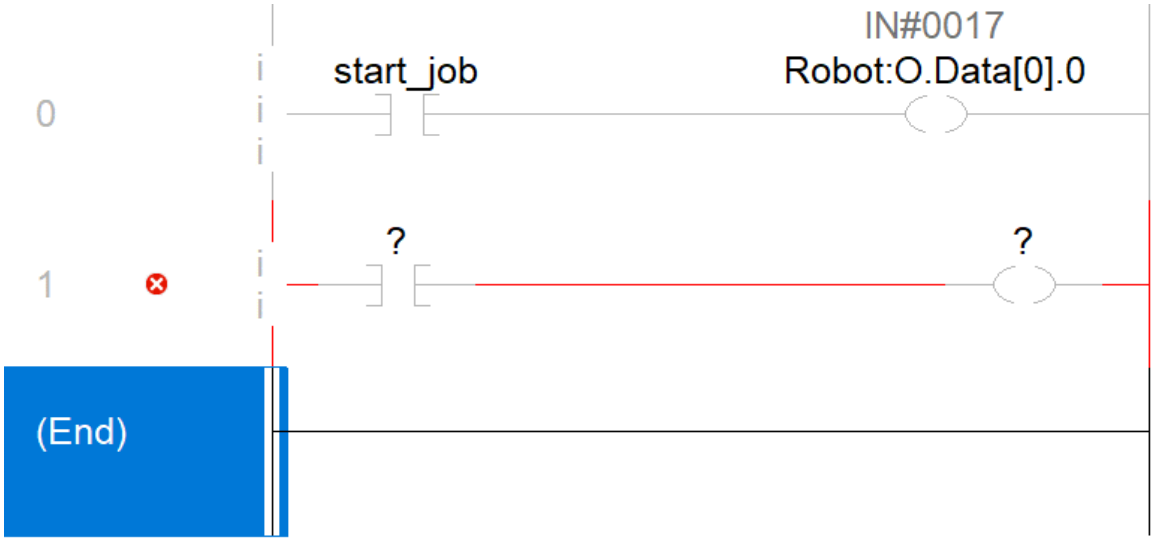


7. Address the OTE to **Robot:O.Data[0].0**. Robot:O.Data[0].0 exchanges data with IN#0017, so when start_job is turned on and Robot:O.Data[0].0 is energized, IN#0017 will turn on and the motion commands of the robot job will begin.

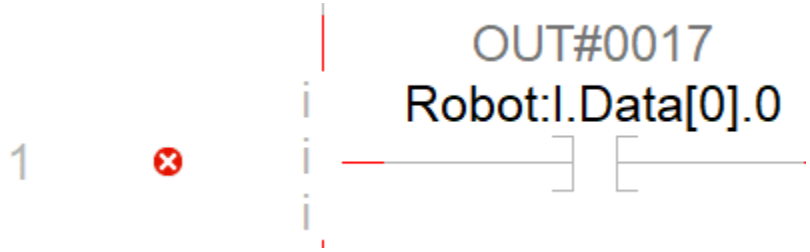
8. Add another rung to the ladder diagram.



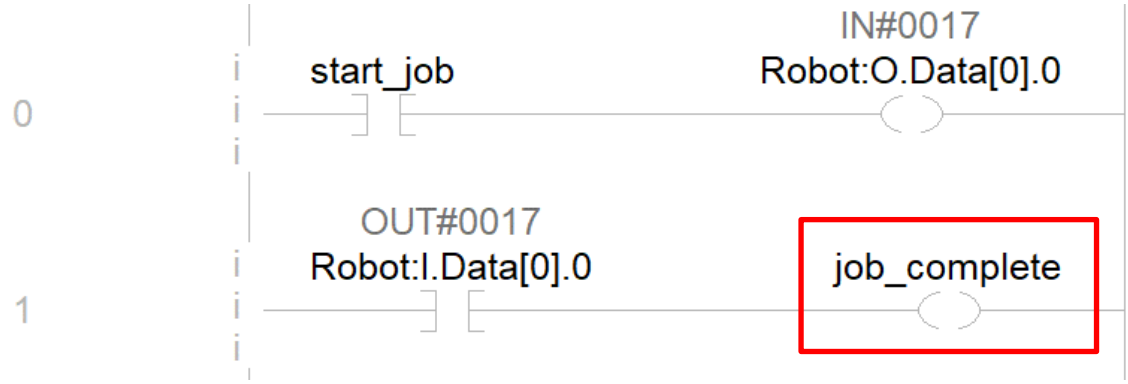
9. This new rung will be responsible for monitoring job progress, and it will let us, and the PLC know when the motion aspects of the robot job have come to an end. On rung 1, add an XIC and an OTE.



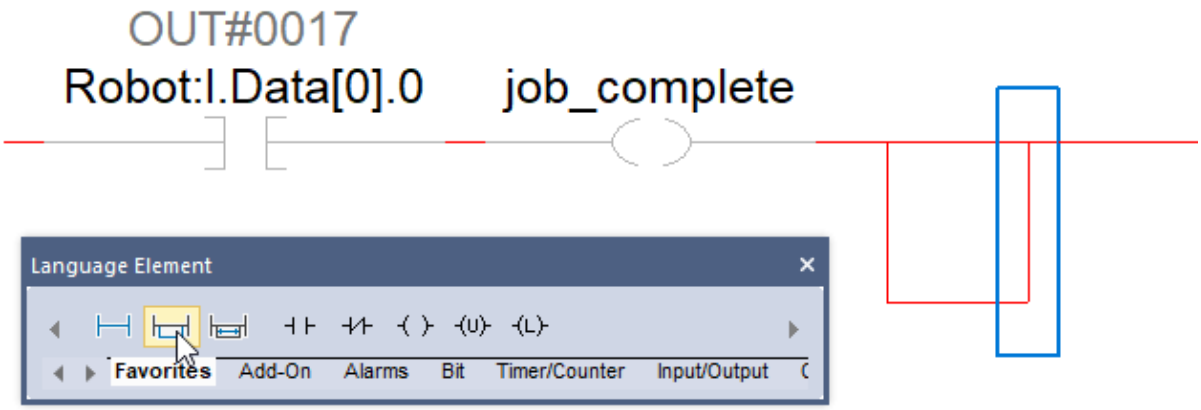
10. Address the XIC to Robot:I.Data[0].0. This BOOL tag exchanges data with OUT#0017. Recall from the previous task that OUT#0017 is turned on after the job's motion commands.



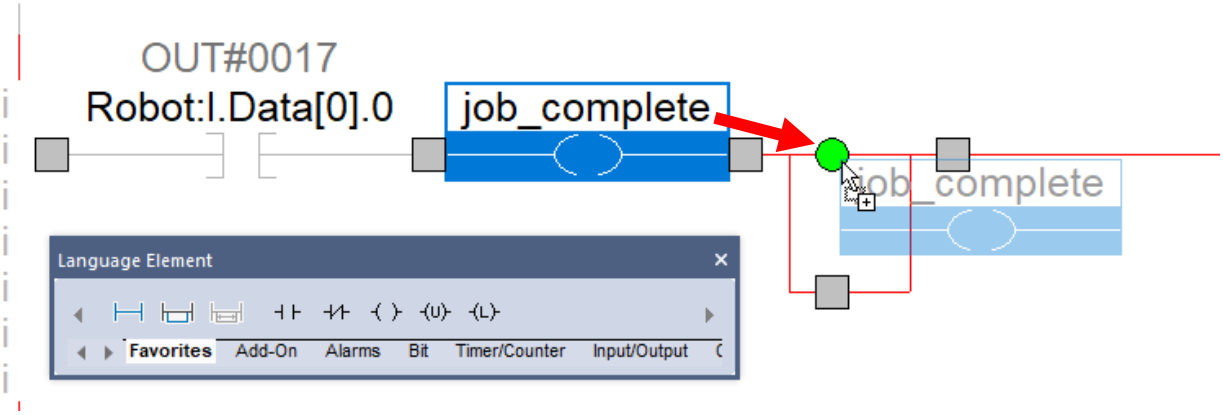
11. Address the OTE to a new BOOL tag named **job_complete** or similar.



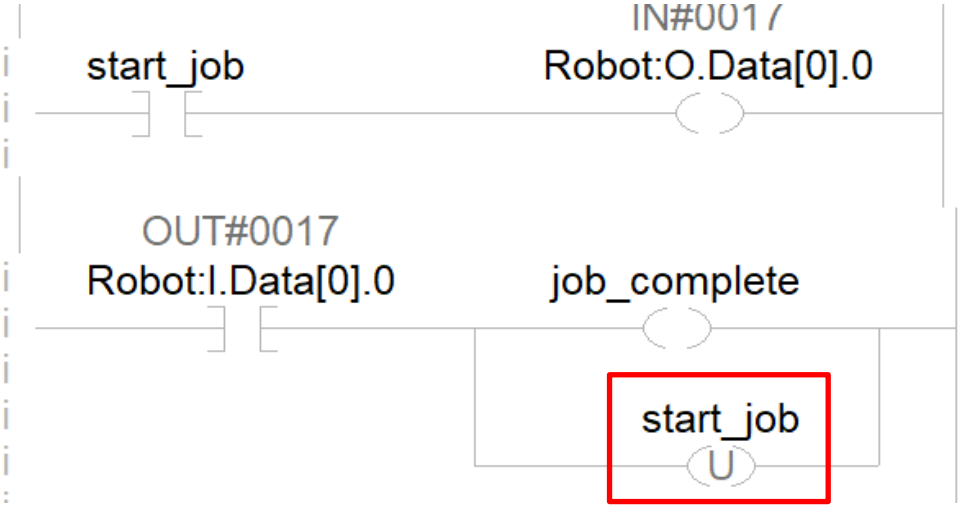
12. Add a branch at the end of rung 1.



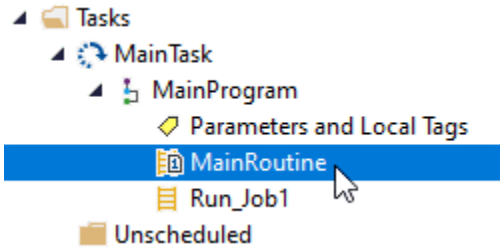
13. Drag the OTE address to job_complete to the top of the branch on the main rung.



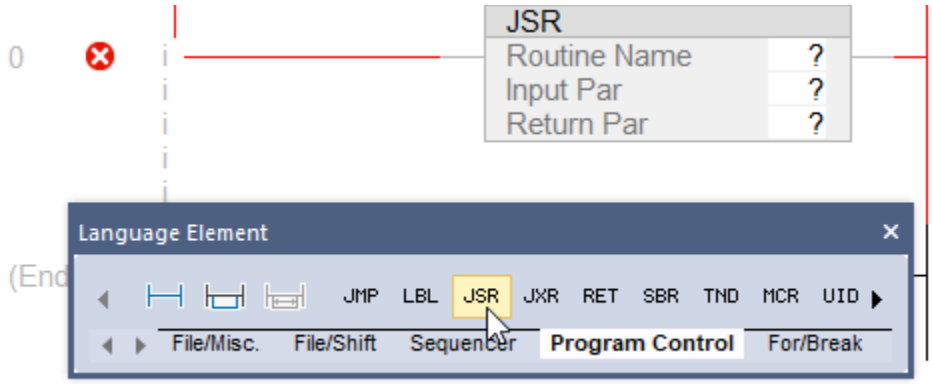
14. To the bottom branch, add an **Output Unlatch (OTU)** instruction. Address the instruction to **start_job**. This instruction will serve to turn the start_job trigger off automatically. The job will be allowed to be initiated only after the 10 second timer has expired and OUT#0017 turns back off.



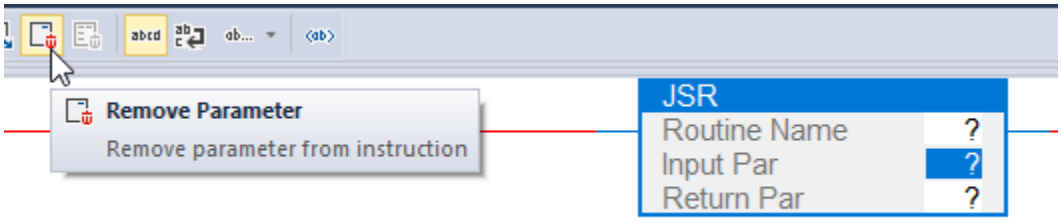
15. Double-click MainRoutine to open its ladder diagram.



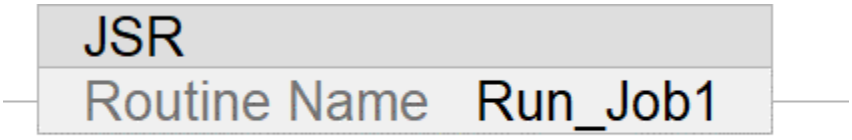
16. To the empty rung 0, add a **Jump to Subroutine (JSR)** instruction.



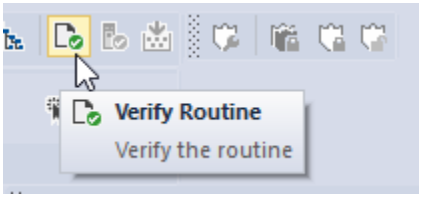
17. Select and remove the Input Par and Return Par.



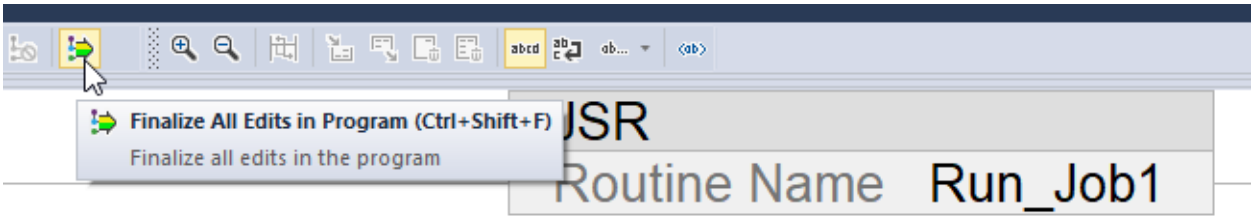
18. For the Routine Name, enter the name of the routine you created above.



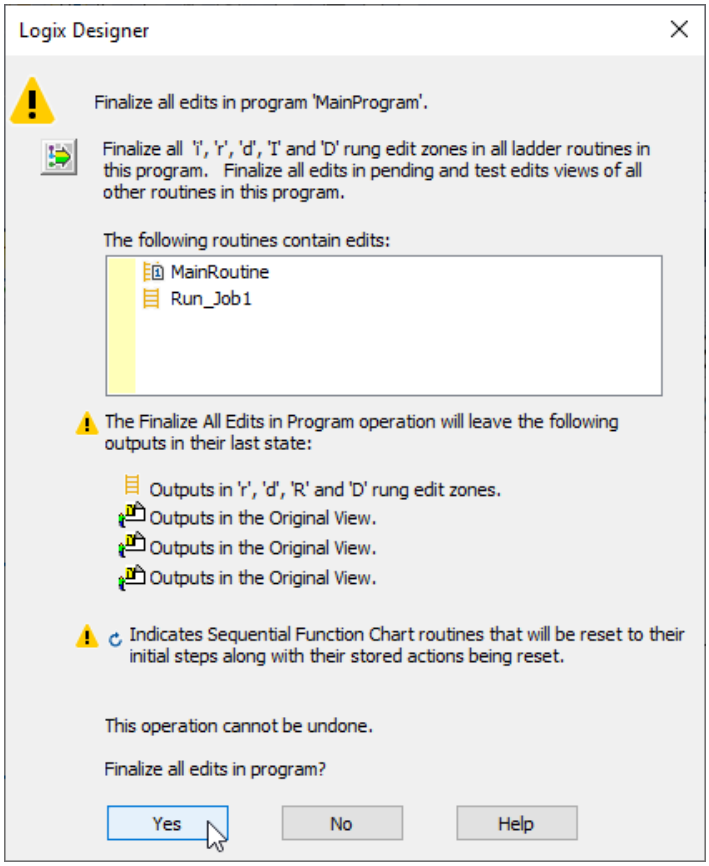
19. Ensure that there are no compile errors in either of the routines. You can use the Verify Routine button if you wish.



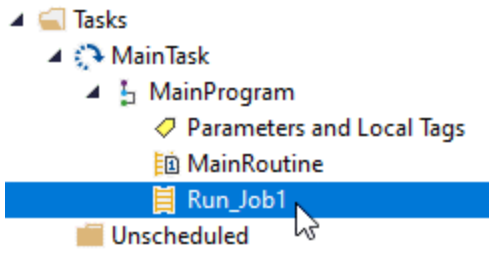
20. Click the **Finalize all Edits in Program** button.



21. In the popup dialog, click **Yes** to finalize the edits in both routines and to download the changes to the PLC.



22. Double-click the subroutine to open its ladder diagram.

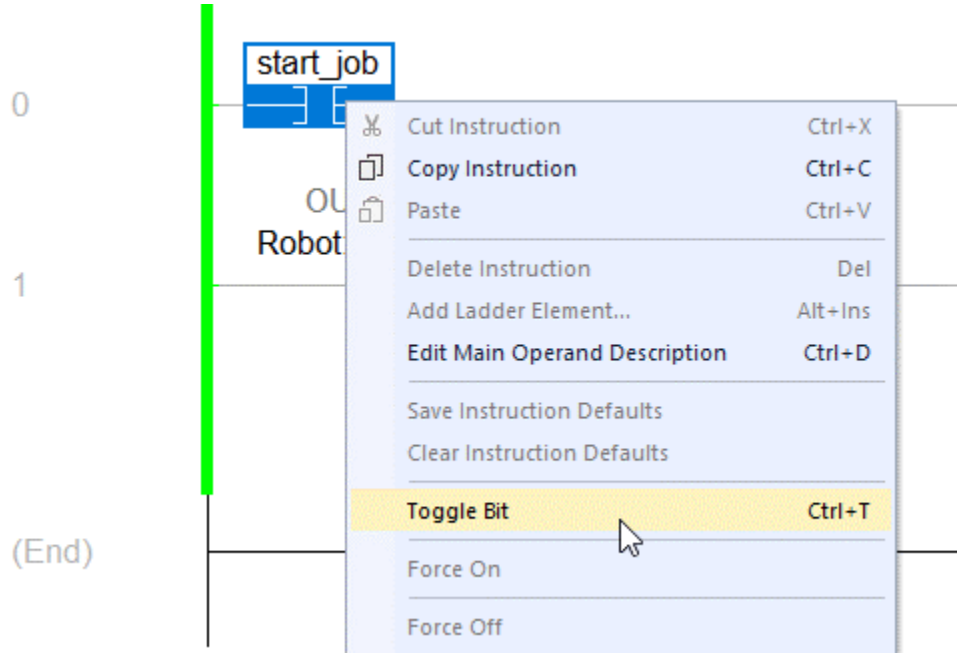


6.6. Running the Job

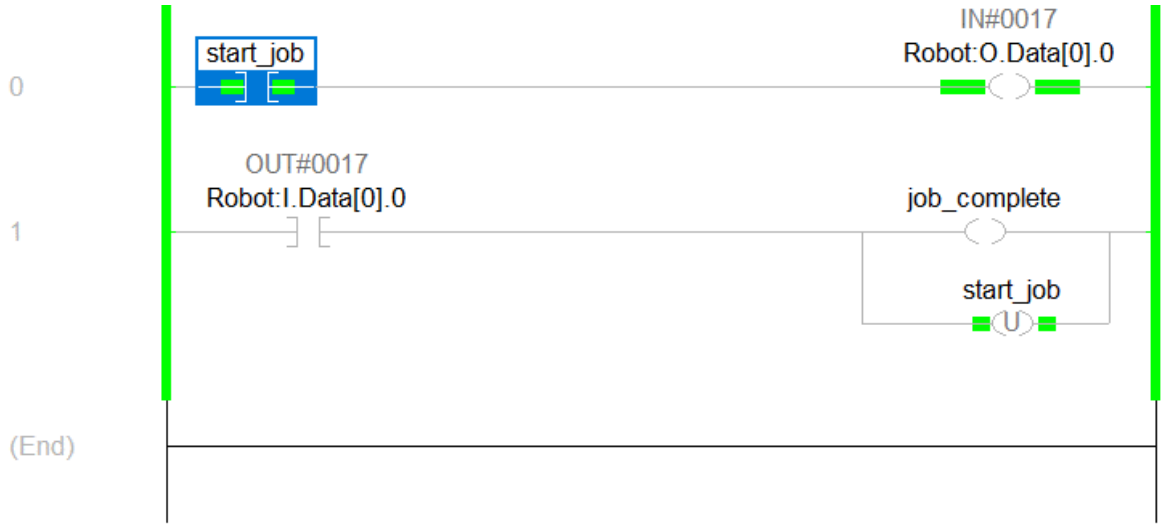
In this task, you will run the robot job on the pendant while using the PLC logic to initiate the motion commands of the job.

Warning: *Be prepared to press the pendant's Emergency Stop button should anything unexpected happen.*

1. On the pendant, turn the mode key to Play mode (central position).
2. Press SERVO ON READY to turn servo power on.
3. Press the green play button.
4. In the subroutine, right-click the start_job XIC and then select **Toggle Bit**. Alternatively, select the XIC and then press **Ctrl + T** on your keyboard.

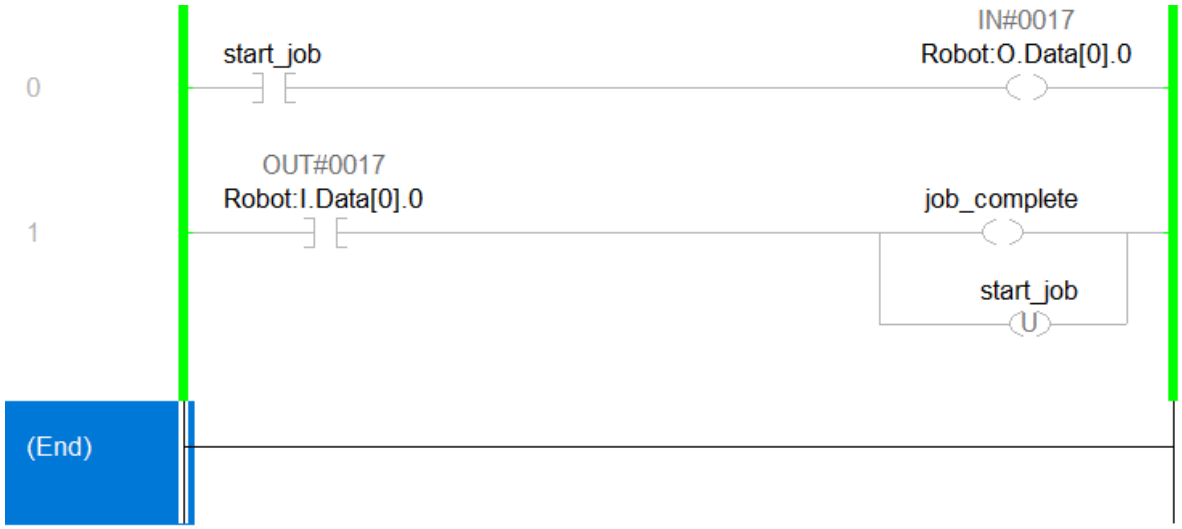


Robot:O.Data[0].0 is energized, IN#0017 turns on, and the job runs.



Once the motion commands of the job are complete, OUT#0017 turns on. This energizes the job_complete tag. During the 10 second timer, start_job is unlatched: the tag is switched off and it cannot be toggled back on during this period.

After the 10 seconds elapse, OUT#0017 and Robot:I.Data[0].0 turn off. The job can be initiated once again by the start_job XIC.



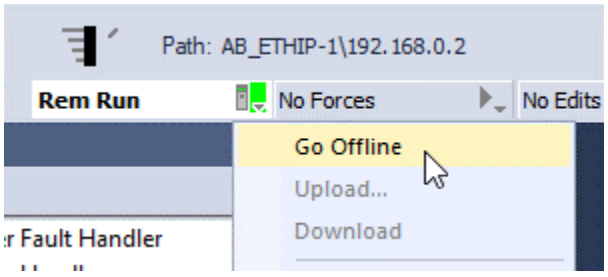
5. Toggle the start_job XIC on again and observe the behavior of the PLC program and robot job.

Several notes about this PLC program / robot job combination:

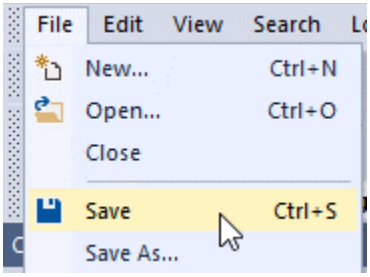
- The job itself runs on a loop. This is only for the convenience of this particular lab activity. The jobs used in the project may not necessarily need to be looped, as they will be jumped to and from other jobs. You will learn about this in the next lab activity.
- Toggling BOOL tags and observing the values of BOOL tags on a ladder diagram is fine for testing, but if you really want to operate and modify this type of system, the tags should be bound to the graphic objects of an HMI. You will experience this in the section on HMIs.

6. Switch the pendant mode key back to Teach mode.

7. Go offline.



8. Save the project.



7. Authentic Skill Assessment

Have your instructor verify that your work meets the requirements in the performance objectives and sign below. Keep this lab activity sheet for future reference.

Instructor Signature	Date

8. Reset Steps

This lab activity does not have any reset steps.

9. Shutdown

Unless instructed otherwise by your instructor, review and complete each of the items on the checklist below.

- ☐ Jog the robot to a safe position with the gripper jaws pointing downwards.
- ☐ Return the pendant to its storage hook on the side of the SmartCart.
- ☐ Power down the robot.
- ☐ Power down the I/O box.
- ☐ Close Logix Designer.