

Important User Information

Read this document and the documents listed in the additional resources section about installation, configuration, and operation of this equipment before you install, configure, operate, or maintain this product. Users are required to familiarize themselves with installation and wiring instructions in addition to requirements of all applicable codes, laws, and standards.

Activities including installation, adjustments, putting into service, use, assembly, disassembly, and maintenance are required to be carried out by suitably trained personnel in accordance with applicable code of practice.

If this equipment is used in a manner not specified by the manufacturer, the protection provided by the equipment may be impaired.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.



WARNING: Identifies information about practices or circumstances that can cause an explosion in a hazardous environment, which may lead to personal injury or death, property damage, or economic loss.



ATTENTION: Identifies information about practices or circumstances that can lead to personal injury or death, property damage, or economic loss. Attentions help you identify a hazard, avoid a hazard, and recognize the consequence.

IMPORTANT

Identifies information that is critical for successful application and understanding of the product.

Labels may also be on or inside the equipment to provide specific precautions.



SHOCK HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that dangerous voltage may be present.

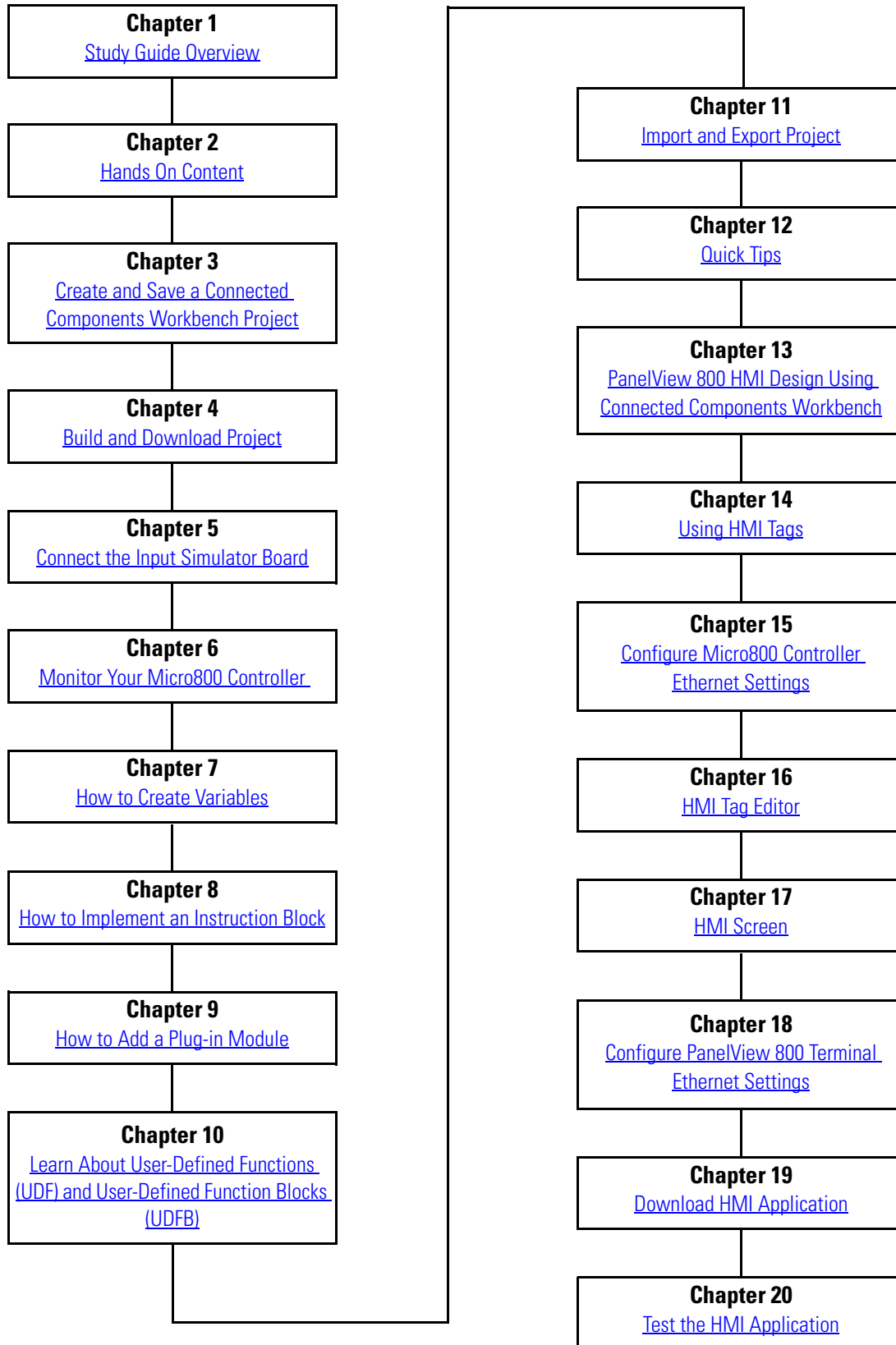


BURN HAZARD: Labels may be on or inside the equipment, for example, a drive or motor, to alert people that surfaces may reach dangerous temperatures.



ARC FLASH HAZARD: Labels may be on or inside the equipment, for example, a motor control center, to alert people to potential Arc Flash. Arc Flash will cause severe injury or death. Wear proper Personal Protective Equipment (PPE). Follow ALL Regulatory requirements for safe work practices and for Personal Protective Equipment (PPE).

Follow this path to learn how to use the Connected Components Workbench™ software with your Micro800™ controllers and PanelView™ 800 terminals.



Notes:

	Important User Information	2
Preface	About This Publication	9
	Additional Resources	9
Study Guide Overview	Chapter 1	
	Objectives	11
	Required Tools.....	11
	Start Connected Components Workbench Software	11
	Connected Components Workbench Design Environment.....	12
	Project Organizer.....	12
	Workspace.....	13
	Toolbox	14
	Output	14
	Status Bar.....	15
Hands On Content	Chapter 2	
	Overview	17
Create and Save a Connected Components Workbench Project	Chapter 3	
	Create a Connected Components Workbench Project	19
	Save Your Connected Components Workbench Project.....	25
Build and Download Project	Chapter 4	
	Build and Download the Project to your Micro800 Controller	27
Connect the Input Simulator Board	Chapter 5	
	31
Monitor Your Micro800 Controller Program	Chapter 6	
	Monitor Your Program in Connected Components Workbench Software	33
	View Real-time Changes in List Format	34
How to Create Variables	Chapter 7	
	Create Local Variables for your Program	37
How to Implement an Instruction Block	Chapter 8	
	Add a TON Instruction Block	39
	Enter Parameters for TON Instruction Block.....	41
	Add an ANY_TO_TIME Instruction Block	44

How to Add a Plug-in Module	Chapter 9 Add a Plug-in Module to the Micro800 Controller..... 49
Learn About User-Defined Functions (UDF) and User-Defined Function Blocks (UDFB)	Chapter 10 Create a UDF for your Program 53 Add the UDF to your Program 55 Get Sample Code from the Rockwell Automation Sample Code Library..... 59 Import Sample Code into Your Project 61
Import and Export Project	Chapter 1165
Quick Tips	Chapter 1267
PanelView 800 HMI Design Using Connected Components Workbench	Chapter 13 Add a PanelView 800 Terminal to Your Project..... 69 Configure Your PanelView 800 Terminal Communication Settings .. 70
Using HMI Tags	Chapter 14 Create Global Variables..... 73 Edit Ladder Diagram Program..... 76
Configure Micro800 Controller Ethernet Settings	Chapter 1579
HMI Tag Editor	Chapter 1681
HMI Screen	Chapter 17 Create a Screen for Your PanelView 800 Application..... 83 Create Objects for Your Screen..... 84
Configure PanelView 800 Terminal Ethernet Settings	Chapter 18 101
Download HMI Application	Chapter 19 Save and Download Project 103

Test the HMI Application	Chapter 20
	Run the HMI Application 105
	Test the HMI application. 105
Set Up Serial Communication Between PC and Micro820 Controller	Appendix A
	Before You Begin..... 107
	Connect the PC to the Micro820 Controller 108
	Configure RSLinx..... 109
	Restore Serial Port to Factory Default Setting
	Using MicroSD Card..... 112

Notes:

About This Publication

Use this quick start to learn how to use a Micro800 controller with a PanelView 800 terminal. You will use the Connected Components Workbench software.

Additional Resources

These documents contain additional information concerning related products from Rockwell Automation.

Resource	Description
Micro800 Controllers Starter Pack Quick Reference, publication 2080-QR003	Provides information for setting up the Micro800 controllers and input simulator boards.
PanelView 800 HMI Terminals User Manual, publication 2711R-UM001	Information in setting up and using the PanelView 800 terminals, and installing the Connected Components Workbench software.
Micro820 20-point Programmable Controllers User Manual, publication 2080-UM005	Information on setting up and using the Micro820™ controllers.
Micro830 and Micro 850 Programmable Controllers User Manual, publication 2080-UM002	Information on setting up and using the Micro830® and Micro850® controllers.
Connected Components Workbench Software Website http://www.rockwellautomation.com/global/support/connected-components/workbench.page	Provides product information on Connected Components Workbench software and links to download the software.
Connected Components Workbench Online Help	Online Help that provides a description of the different elements of the Connected Components Workbench software.

You can view or download publications at <http://www.rockwellautomation.com/literature/>. To order paper copies of technical documentation, contact your local Allen-Bradley distributor or Rockwell Automation sales representative.

Notes:

Study Guide Overview

Objectives

This self study guide is intended for new users of Connected Components Workbench software who have purchased the Micro820 or Micro850 Starter Pack. This document is also a useful self study guide to any new users of Micro800 controllers or PanelView 800 terminals who did not purchase the Micro820 or Micro850 Starter Pack. This self study guide assumes that the user has basic knowledge of a Programmable Logic Controller (PLC).

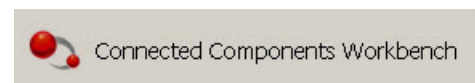
The Micro850 controller is used in the examples shown in this quick start, however they also apply to Micro820 controllers. Any major differences between the Micro820 and Micro850 controllers will be mentioned.

Required Tools

- One of the following Micro800 Starter Packs:
 - Micro820 Starter Pack Lite (2080-LC20-STARTERPACKL) or
 - Micro820 Starter Pack with PanelView 800 (2080-LC20-STARTERPACK) or
 - Micro850 Starter Pack Lite (2080-LC50-STARTERPACKL) or
 - Micro850 Starter Pack with PanelView 800 (2080-LC50-STARTERPACK)
- Micro800 controller firmware revision 10.011 or later
- Connected Components Workbench software version 10.01 or later
Software can be downloaded from the Connected Components Workbench website
<http://www.rockwellautomation.com/global/support/connected-components/workbench.page>

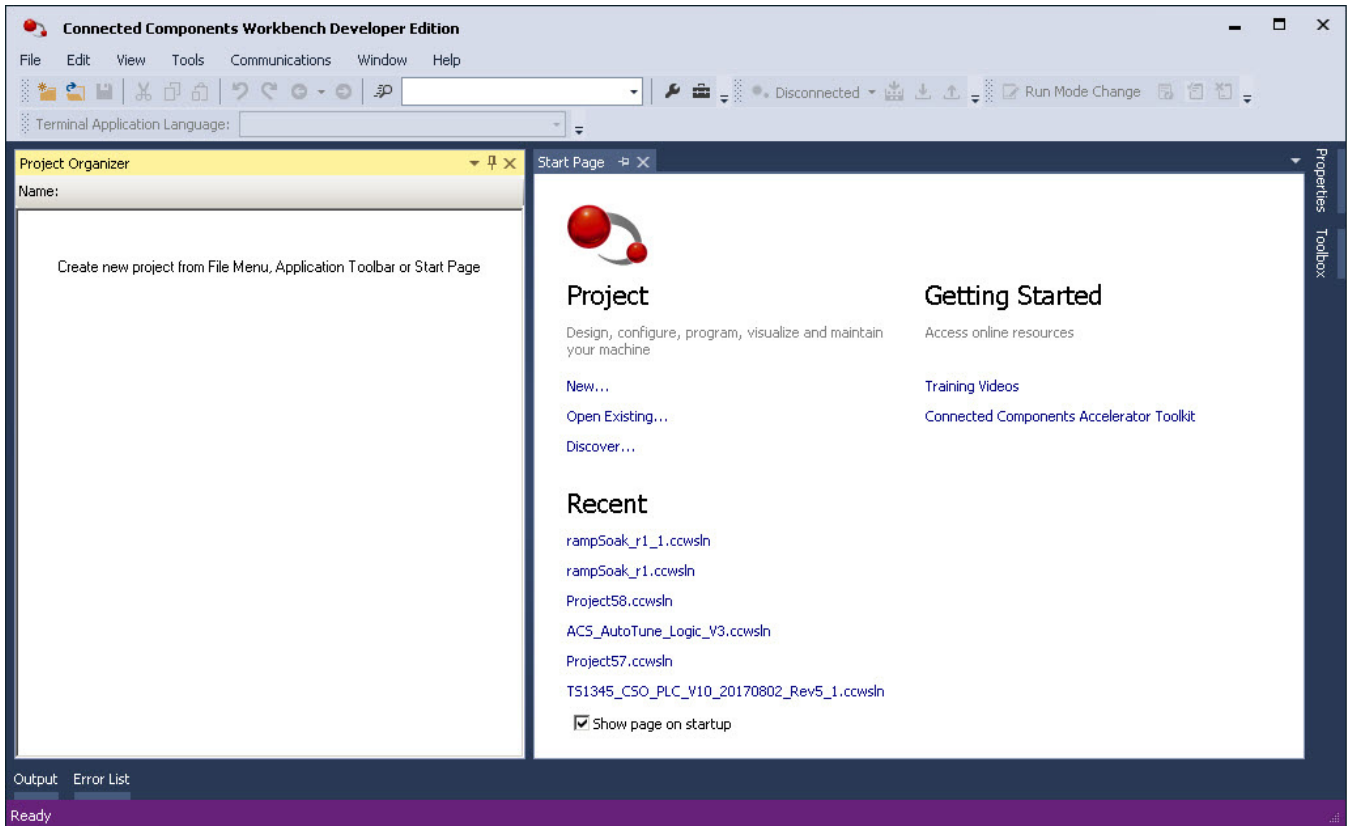
Start Connected Components Workbench Software

To start the Connected Components Workbench (CCW) software, launch the program from your Windows Start Menu by going to: Start -> All Programs -> Rockwell Automation -> CCW -> Connected Components Workbench.



Connected Components Workbench Design Environment

This is the default project layout. The contents of each window and the general task the window is used for are described below.

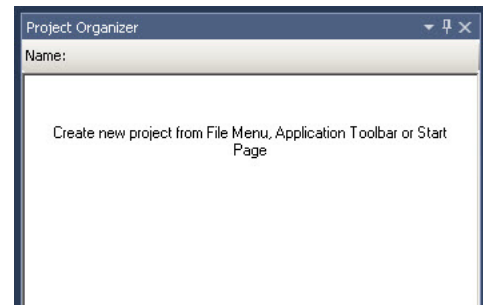


Project Organizer

Project Organizer displays the contents of your project in an organized tree view, providing access to each of the devices and project elements.

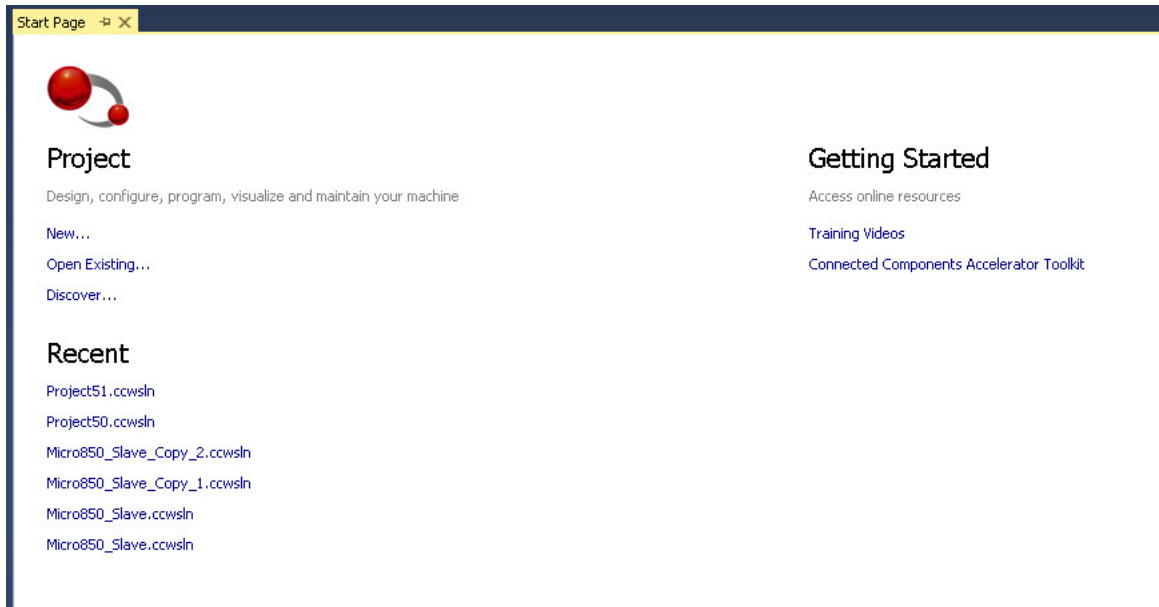
From the Project Organizer, you can add, move, or delete devices and project elements, and double-click them to display their contents.

If your project contains a Micro800 controller, the Project Organizer also displays the logic programs, variables, user-defined functions (UDF) and user-defined function blocks (UDFB) that are associated with that controller.



Workspace

The Start Page in the workspace provides a starting point for your work flow. It is categorized into three sections – Project, Recent and Getting Started.



Under the Project section, you can click:

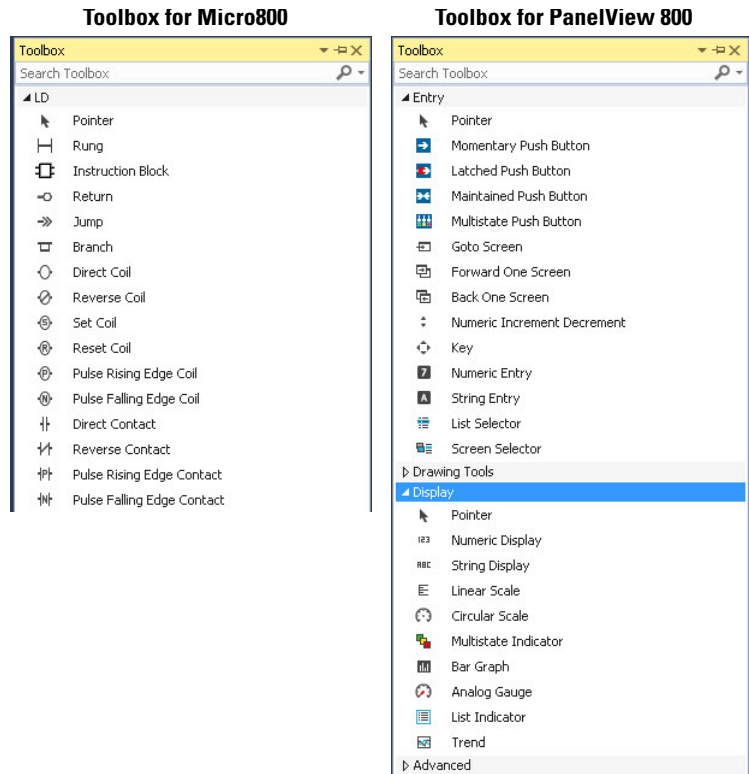
- **New** to create a new project
- **Open Existing** to browse for another project file not listed under Recent
- **Discover** to browse for a device such as a Micro800 controller using USB or Ethernet

The Recent section provide a list of project files that were opened recently. The Getting Started section provides links to online content that will help you with developing applications in Connected Components Workbench software.

Toolbox

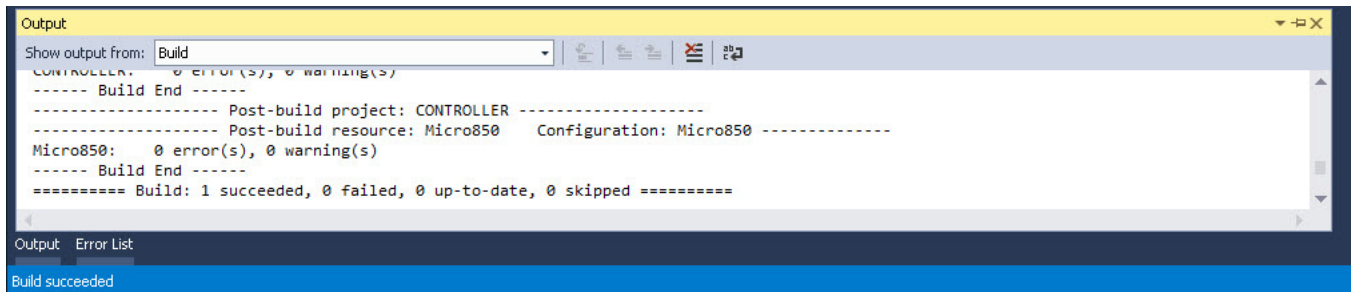
The Toolbox is context sensitive to the device or workspace being edited. The toolbox is used when editing a Micro800 program and when editing a PanelView 800 screen.

From the Toolbox, you can drag and drop Toolbox elements, or copy and paste elements to another window.



Output

In the Output window, you can view and manage general purpose and debug messages that are generated by the various features of Connected Components Workbench software.



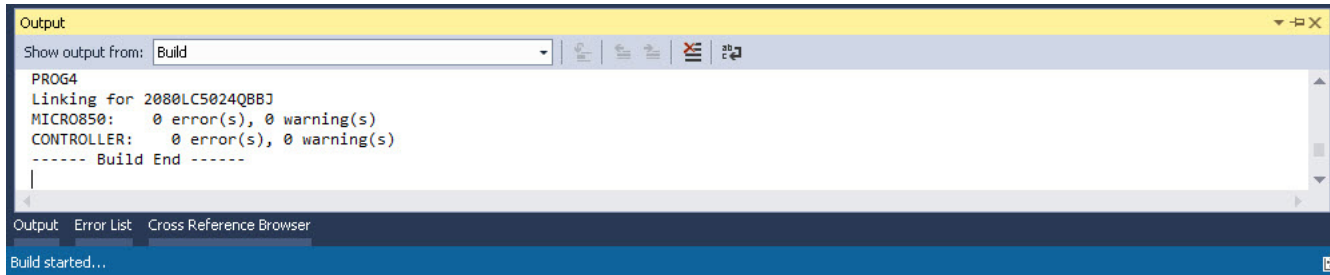
From the Output window, you can do the following:

- Review status messages
- Locate errors within programs

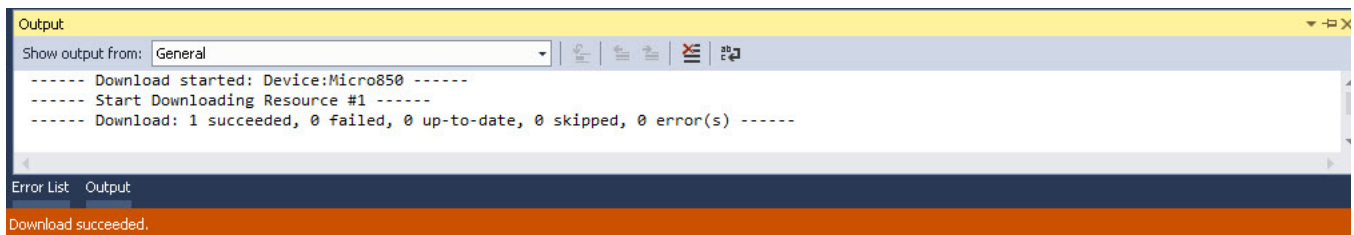
Status Bar

The Status bar shows the task that the Connected Components Workbench software is currently performing.

Connected Components Workbench software is building a project



Connected Components Workbench software has successfully download a project into a controller



Notes:

Hands On Content

Overview

The following chapters provide examples for you to learn how to create a project for your Micro800 controller and PanelView 800 terminal in Connected Components Workbench software.

- Create a Connected Components Workbench project
- Save your Connected Components Workbench project
- Build and download your Micro800 controller application
- Connect the input simulator board to your Micro800 controller
- Monitor your Micro800 controller program
- Learn how to create variables
- Learn how to implement an Instruction Block
- Learn how to add a plug-in module
- Learn about User-Defined Functions and User-Defined Function Blocks
- Import/ Export Project (Connected Components Workbench version 8.00 or later)
- Add a PanelView 800 terminal to your Connected Components Workbench project
- Learn how to create HMI tags
- Configure your Micro800 controller Ethernet port
- Learn how to use the HMI tag editor
- Learn how to create screens for your HMI
- Configure your PanelView 800 terminal Ethernet settings
- Download the HMI application to your PanelView 800 terminal
- Test the HMI application

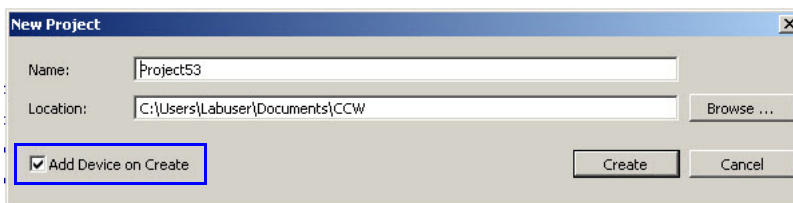
Notes:

Create and Save a Connected Components Workbench Project

Create a Connected Components Workbench Project

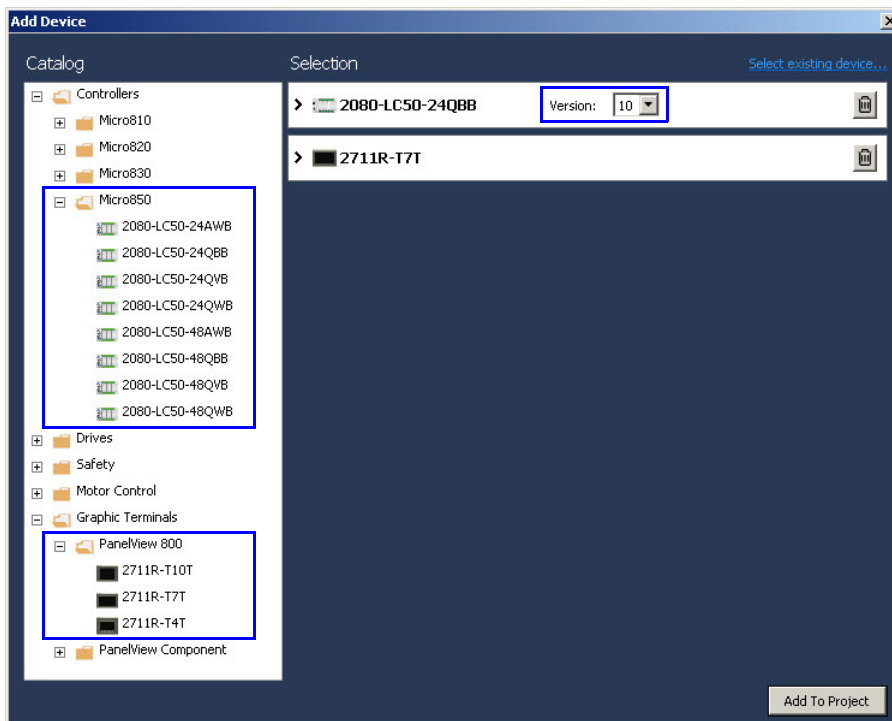
To create a new project, do the following:

1. Click **New** on the Start Page.
Alternatively, go to File -> New or use the keyboard shortcut “Ctrl+N”.
2. Enter a name for your project and click Create.



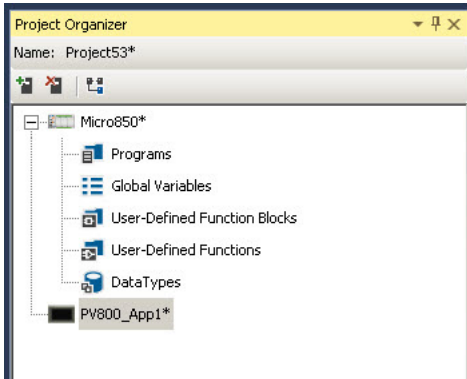
Ensure that the **Add Device on Create** checkbox is selected. This will open an Add Device dialog box for you to add a device to this project.

3. In the left pane, click the “+” sign to expand the list of catalogs and select the device to add into the Selection list.



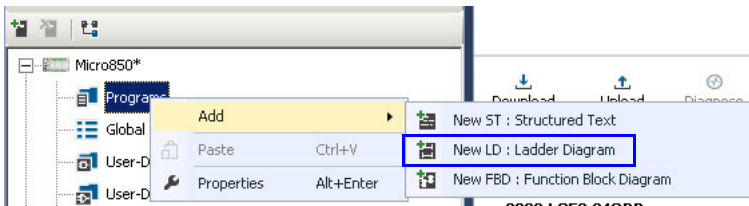
By default, the latest major controller version is selected. Select the required version from the drop-down box.

4. Click Add To Project to complete the device selection.

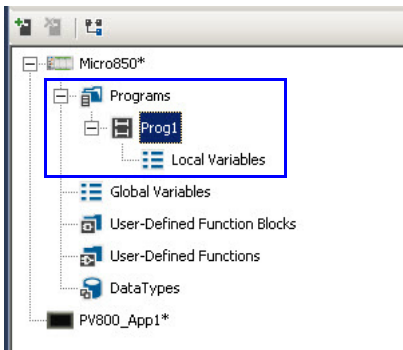


Add a Ladder Diagram program to your project

1. Right-click **Programs** under the Micro850 controller in your Project Organizer, and select Add -> New LD : Ladder Diagram.

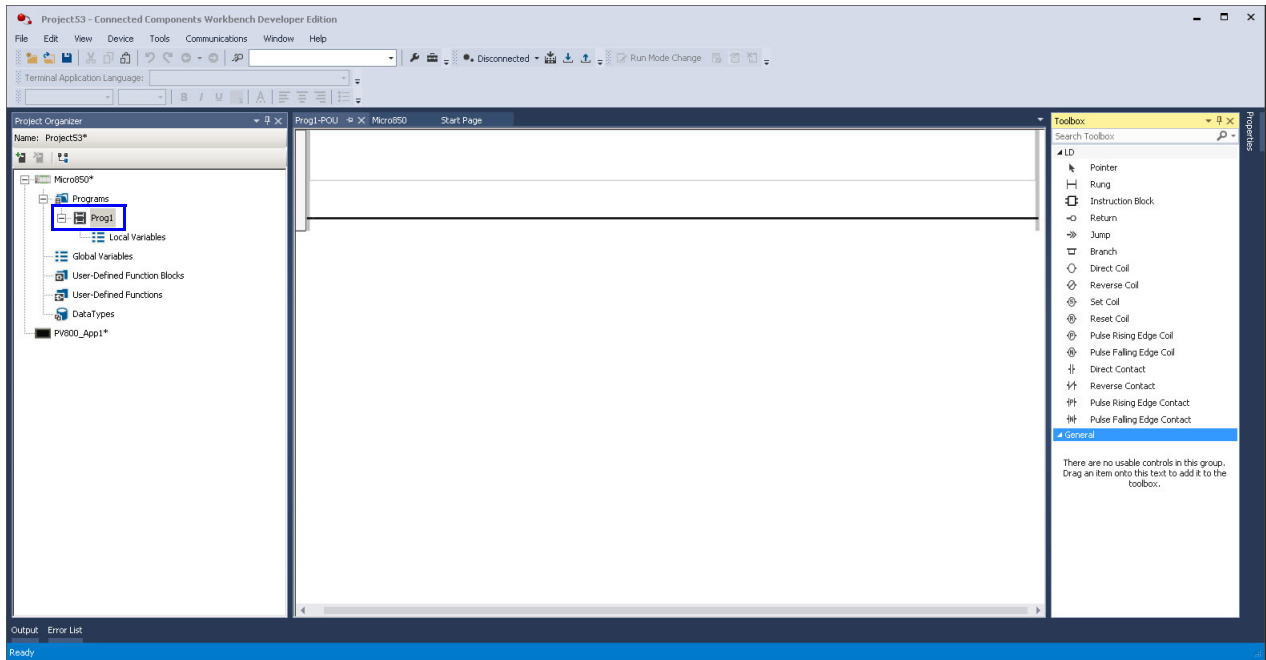


2. Observe that a new Ladder Diagram program called **Prog1** has been added under Programs.

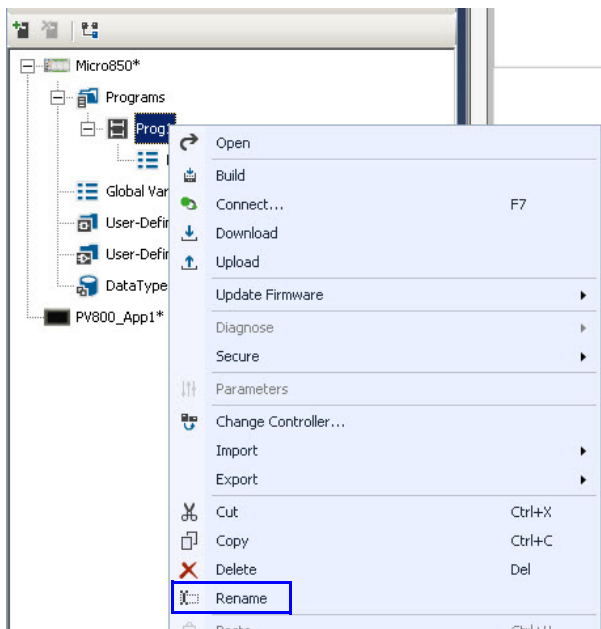


Micro800 controllers allow you to create multiple programs and use multiple types of programs (such as Structured Text or Function Block Diagram) in the same controller application.

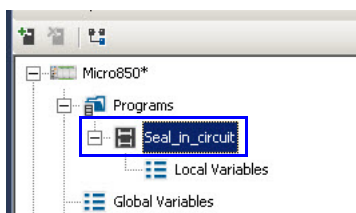
3. Double-click **Prog1**. A ladder diagram editor appears in the main project workspace with one empty rung.



4. Right-click **Prog1** and select Rename to change the name of the program.

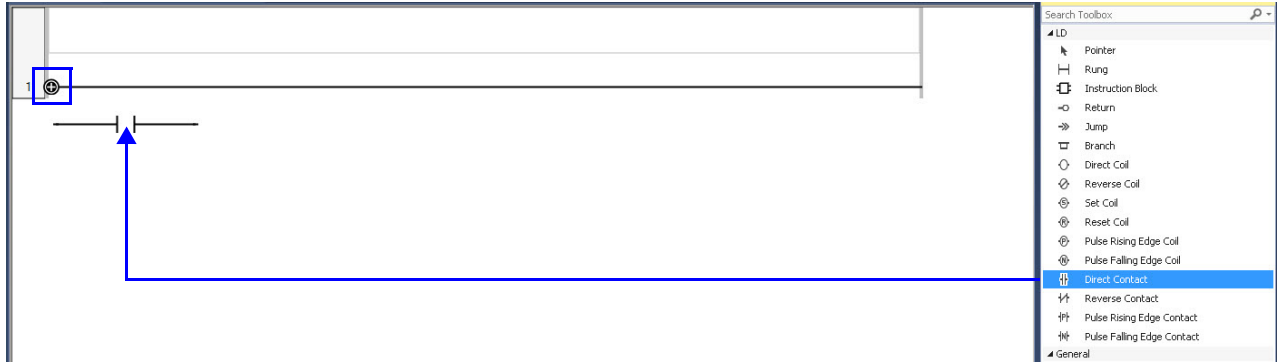


5. Rename the program to **Seal_in_circuit**.

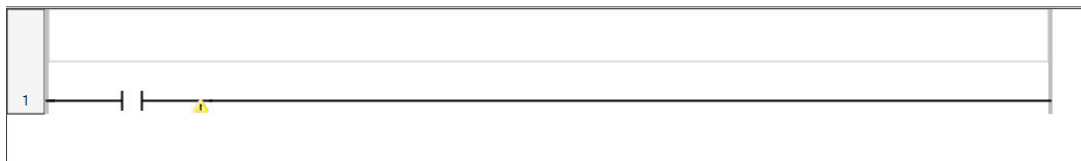


Create the Ladder Diagram program

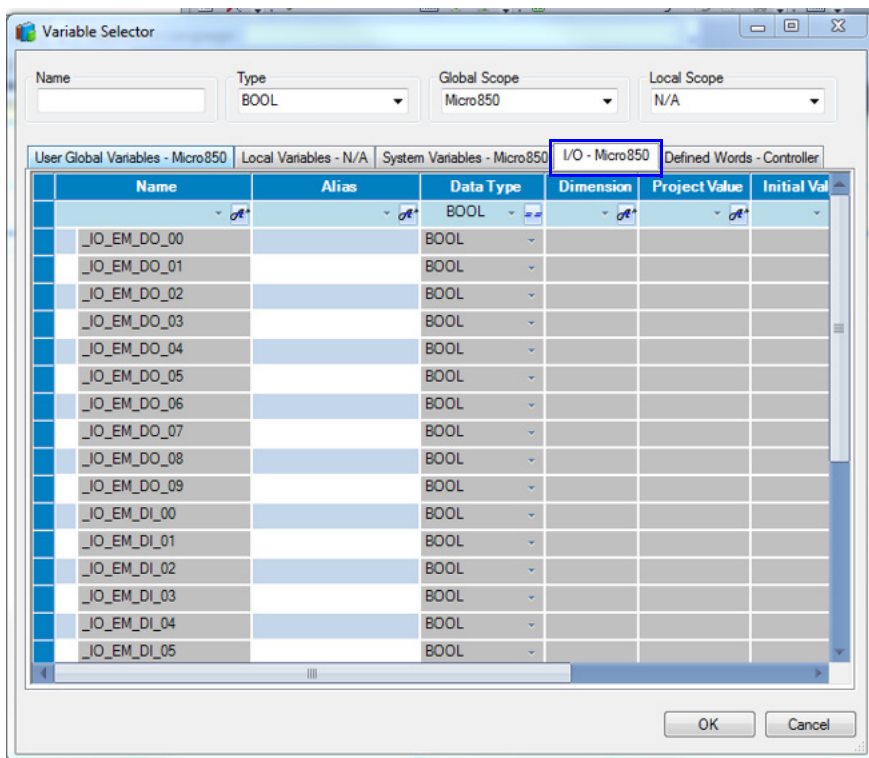
1. Locate the **Direct Contact** instruction in the Toolbox window, and drag-and-drop it onto the left side of the rung.



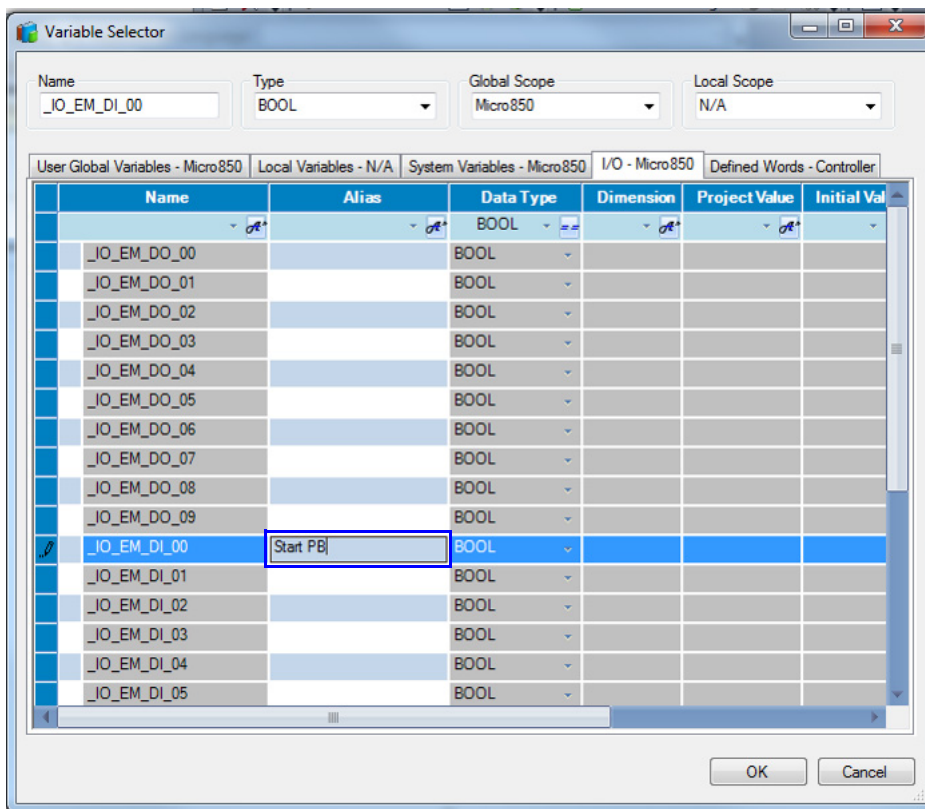
The “+” icon on the rung indicates the location of a drop point for the instruction. After adding the instruction, your rung should look like the following.



2. After inserting the Direct Contact instruction, the Variable Selector dialog box appears and you can select the variable or I/O point to assign to this instruction.
3. In the Variable Selector dialog box, select the **I/O - Micro850** tab to see the list of I/O points.

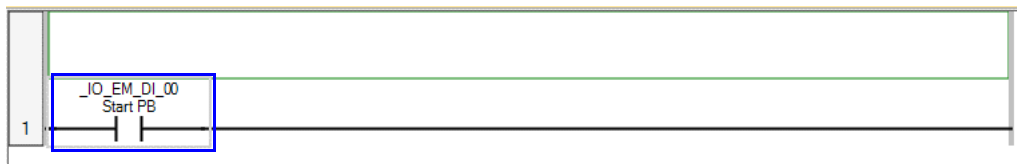


- For this example, assign an embedded I/O point to this instruction. Select **_IO_EM_DI_00**, then in the Alias column of **_IO_EM_DI_00**, type “Start PB” and click OK.

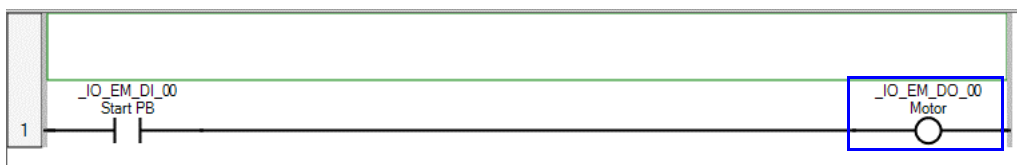


TIP An alias is an optional parameter you can use to further define a local variable or a global variable

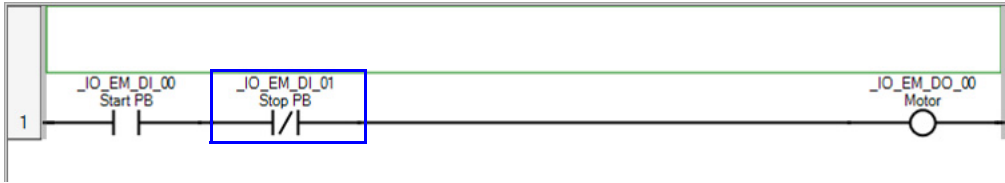
- Your rung should look like the following.



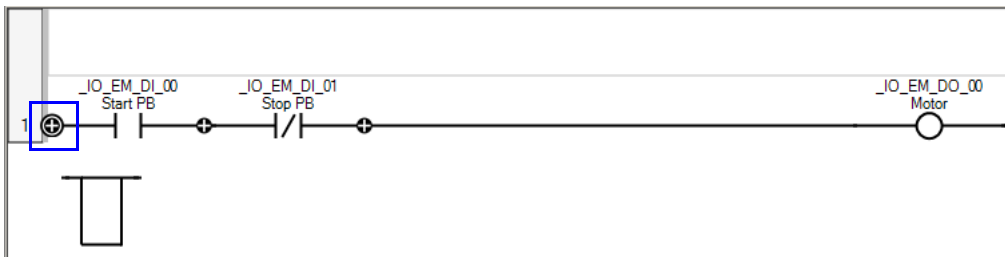
- Locate the **Direct Coil** instruction in the Toolbox, and drag-and-drop it onto the far right side of the rung. Assign it to the embedded I/O point, **_IO_EM_DO_00** with the alias “Motor”. Your rung should look like the following.



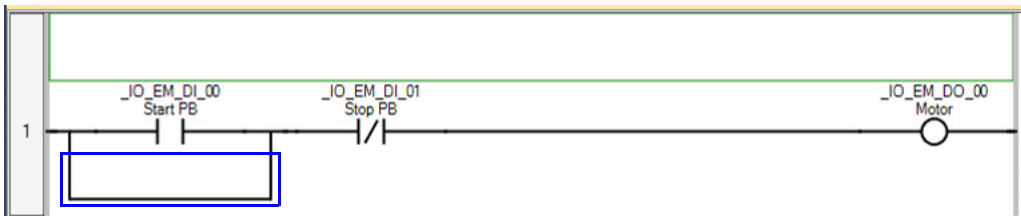
7. Locate the **Reverse Contact** instruction in the Toolbox, and drag-and-drop it onto your rung, just to the right of the Direct Contact instruction. Assign it to the embedded I/O point **_IO_EM_DI_01** with the alias “Stop PB”. Your rung should look like the following.



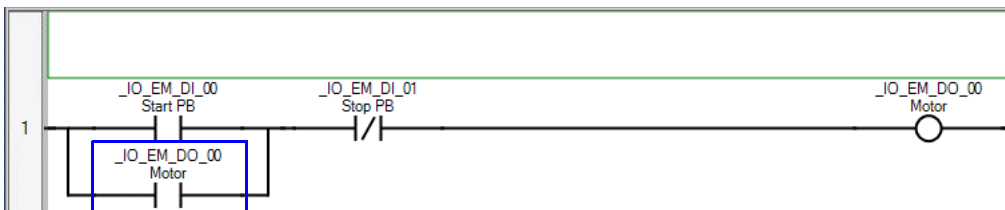
8. Locate the **Branch** instruction in the Toolbox, and drag-and-drop it to the drop point to the left of the Direct Contact instruction on the far left side of the rung.



9. Your rung should look like the following.



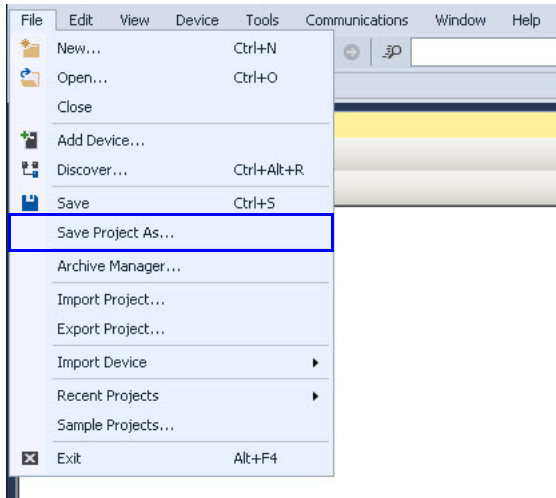
10. Drag-and-drop a **Direct Contact** instruction onto the Branch instruction that you just added. Assign it to the embedded I/O point **_IO_EM_DO_00**. Your rung should look like the following.



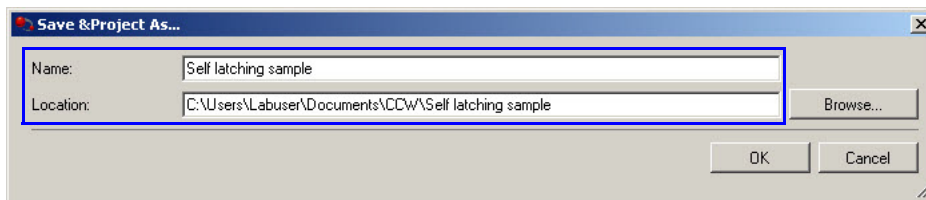
You have completed creating your motor seal-in circuit. When the Start Motor push button is toggled on (while the Stop push button is not being pushed), you complete the rung circuit to the output so that the motor turns on. Once the motor is running, you can release the Start Motor push button because the branch circuit around the push button seals it in. The only way to interrupt the circuit is to push the Stop Motor push button. This breaks the circuit, which turns the motor off and drops out the seal-in branch circuit. The Stop Motor push button can then be released and the motor remains off until the Start Motor push button is pushed again.

Save Your Connected Components Workbench Project

1. Save the project by selecting File -> Save Project As.



2. Save the project under the name “Self latching sample” and click OK.



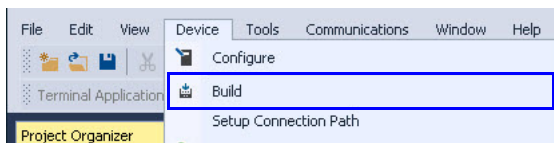
Notes:

Build and Download Project

In this chapter, you will learn how to download a project to your Micro800 controller. Before you can download a project to the controller, you must build it to verify that there are no errors with the programming. The Micro850 controller is used in the examples shown in this chapter.

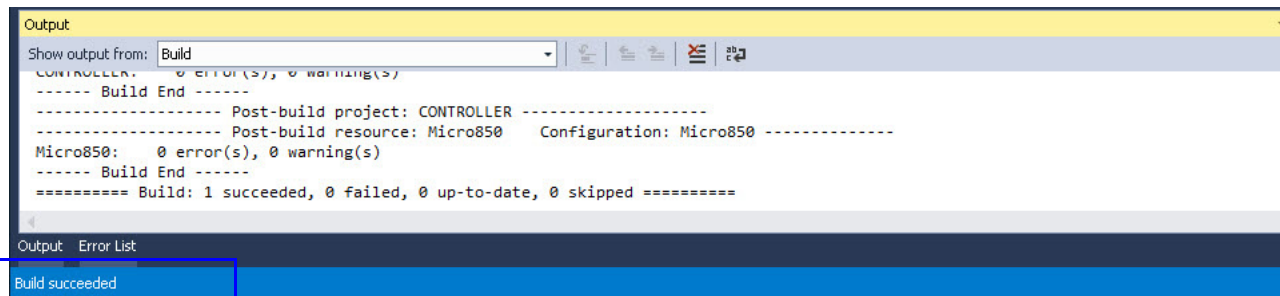
Build and Download the Project to your Micro800 Controller

1. To build the project, click Device -> Build.



When a build is performed, the project is automatically saved before the build is performed.

The Output window and Status bar shows the current status of the build.



2. Connect the PC to your Micro800 controller.

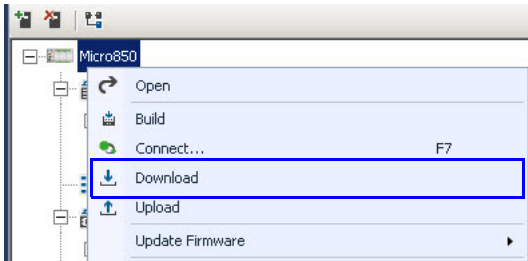
For Micro850 controllers

Connect the USB cable from the PC to the USB port on your Micro850 controller. If this is the first time a specific controller has been connected to this PC over USB, then you must wait for the controller to be detected and the USB driver to be configured.

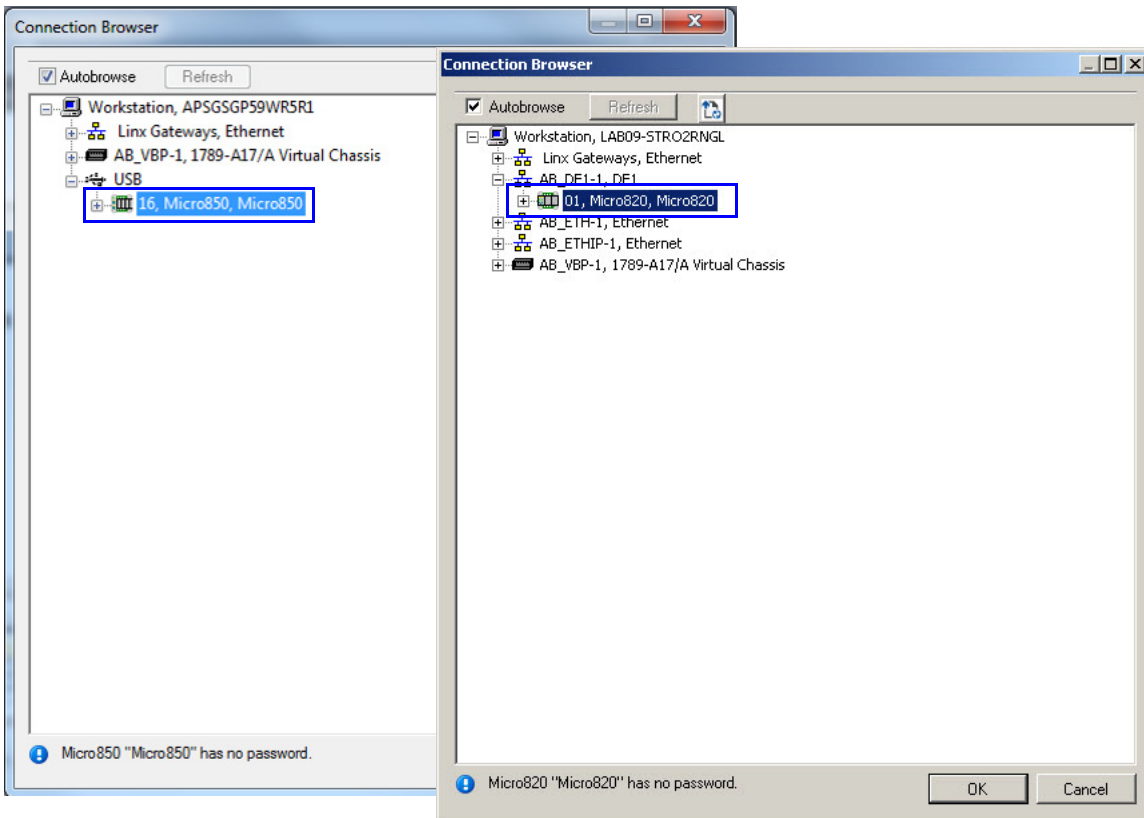
For Micro820 controllers

Connect the serial cable from the PC to the embedded serial port on your Micro820 controller, or connect the USB cable from the PC to the USB port on the 2080-REMLCD. If this is the first time you are connecting your Micro820 controller through serial, see [Appendix A](#) on how to set up a serial communication with your Micro820 controller.

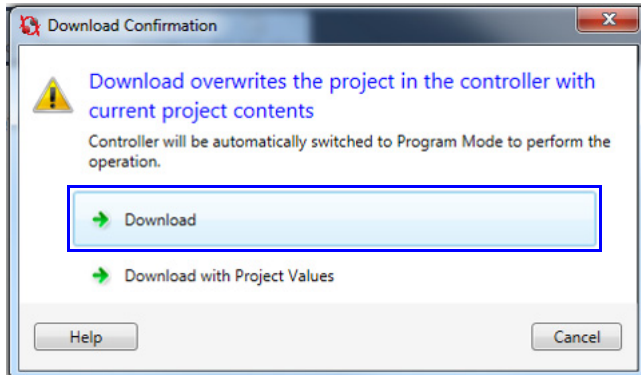
3. Download your project to your Micro800 controller by right-clicking the controller in your Project Organizer, and select **Download**. If this project was modified since the last build, then a build will automatically be executed before the download is performed.



4. If there are no errors in your project, the Connection Browser dialog box appears. Browse for your Micro800 controller by expanding **USB** or **DF1**, then select the controller and click OK.



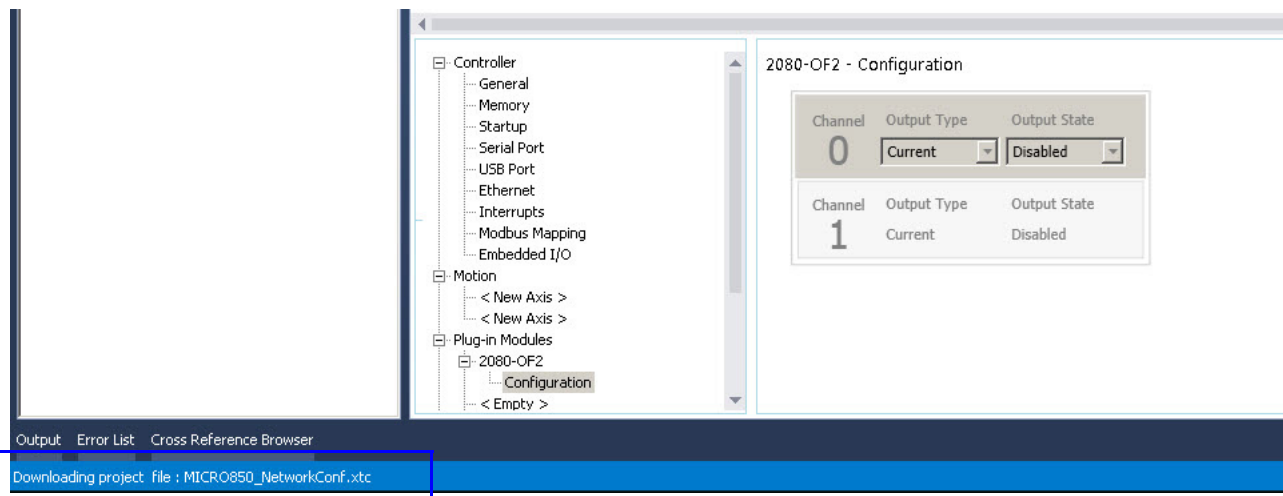
5. In the Download Confirmation dialog box, select **Download**.



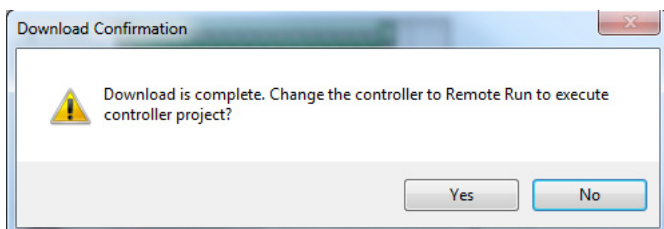
When you download a project, you can choose if you either want to do a typical **Download** (project values are not downloaded and variables are set to their initial value if it exists, or reset if no initial value exists) or **Download with Project Values**. Downloading with project values may take longer than a typical download without project values. Project values generally are populated by an upload with project values or can be entered manually in the variable editor.

Initial value has priority, even if there is a project value and you have chosen to download with project values. If a variable has been configured for Data Protection, then neither initial value or project value will change its value upon download. The logical value in the controller will be preserved after a download.

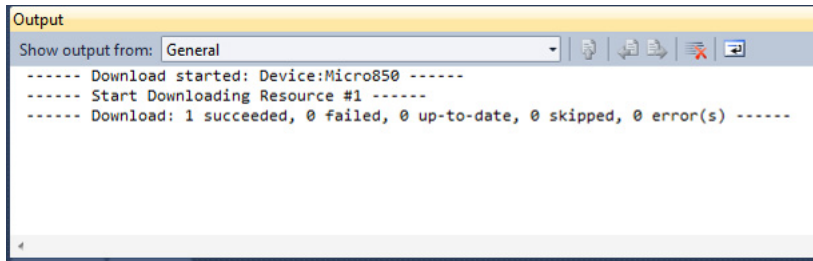
6. Downloading in progress.



7. When the download is complete, you are prompted to put the controller back in Remote Run Mode. Click Yes.



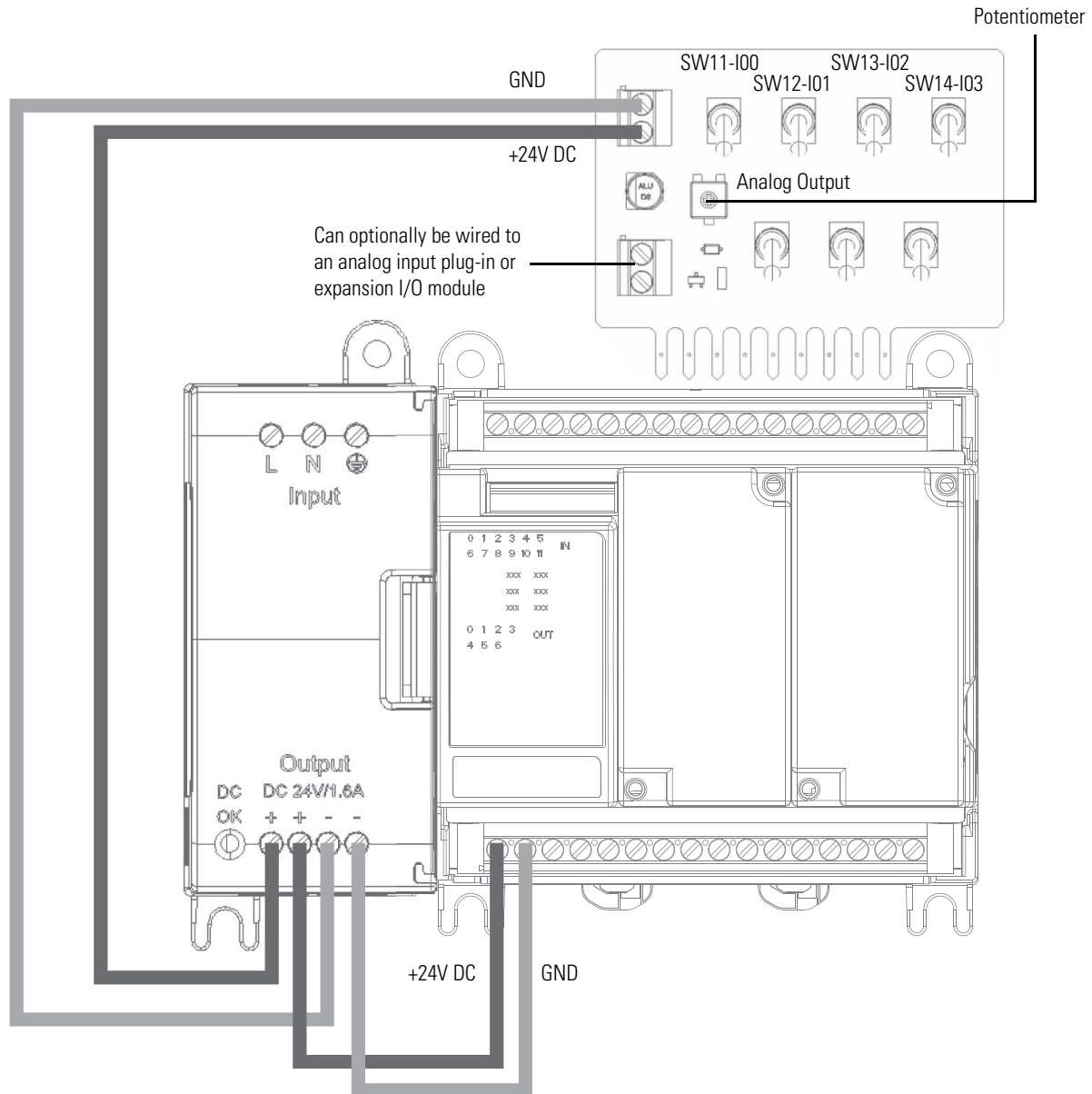
8. Observe the messages in the Output window indicate that the Download has completed successfully.



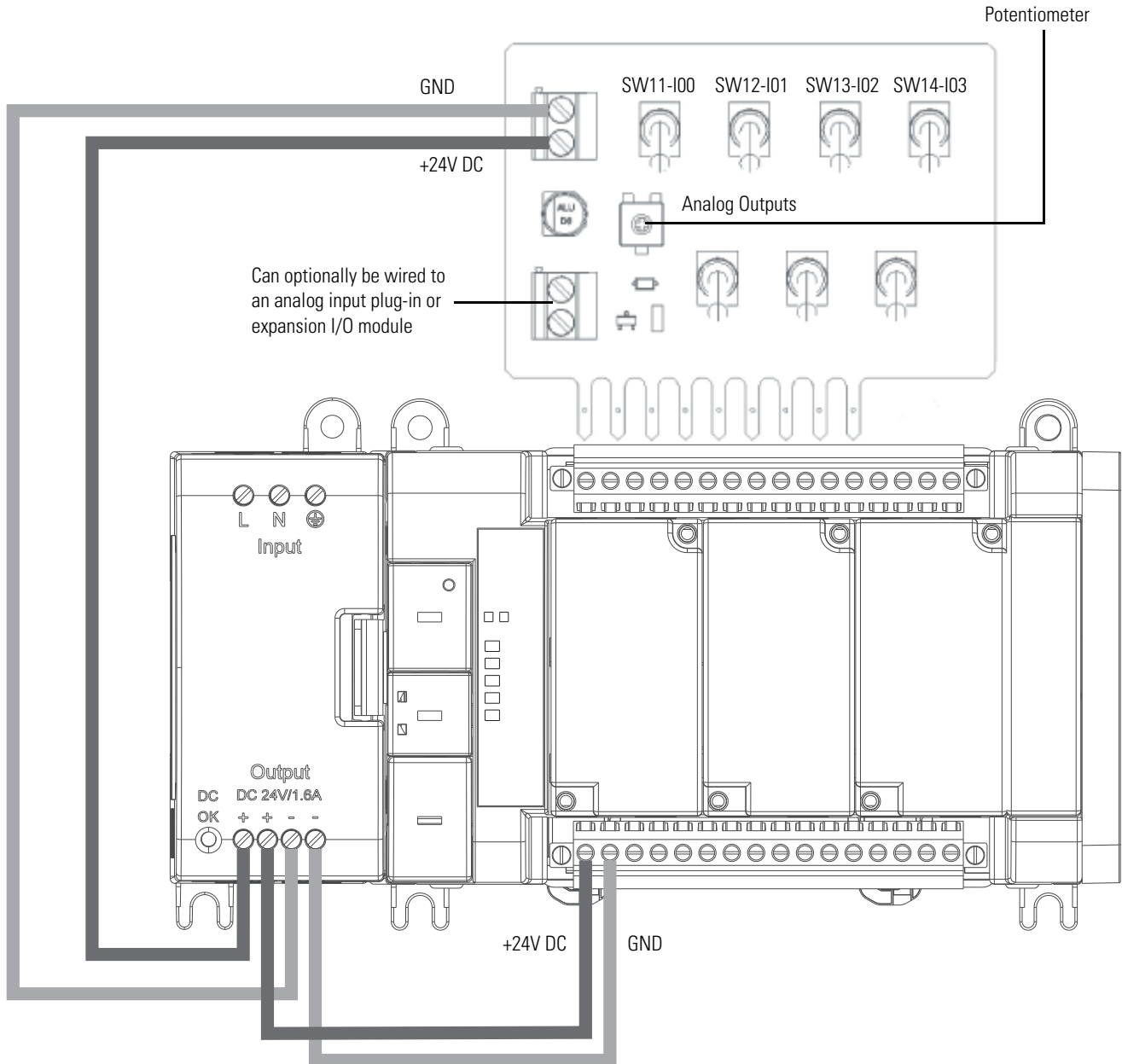
You have completed downloading the project to your Micro800 controller. Proceed to the next chapter to test your project. Connect the input simulator found in your starter pack to the input terminals of the Micro800 controller.

Connect the Input Simulator Board

Micro820 Controller Input Simulator Board Wiring Diagram



Micro850 Controller Input Simulator Board Wiring Diagram

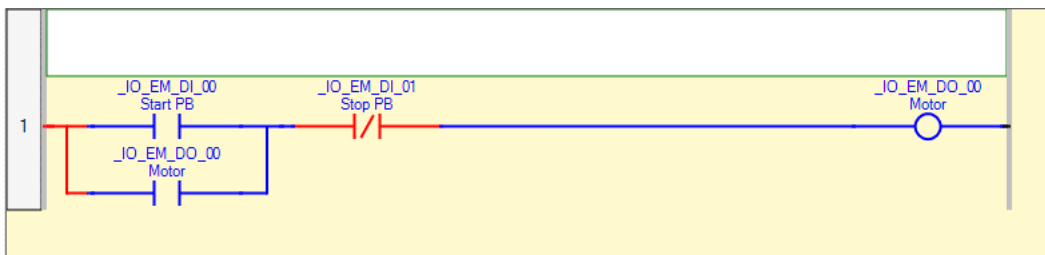


Monitor Your Micro800 Controller Program

Monitor Your Program in Connected Components Workbench Software

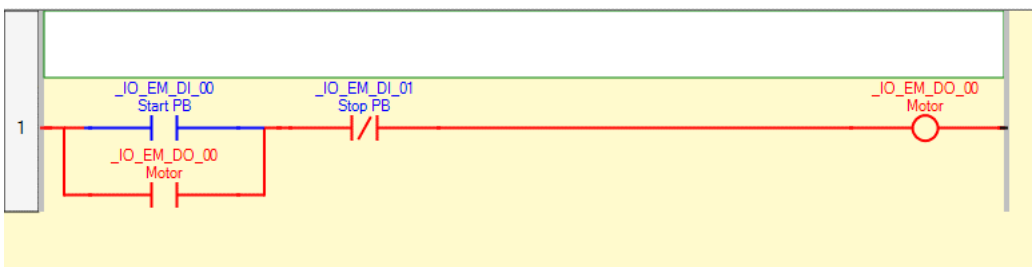
When connected to your Micro800 controller using Connected Components Workbench software:

- You can view your program visually in real-time and watch values change in the program.
- You should see the Ladder Diagram change color as the inputs and outputs change state.



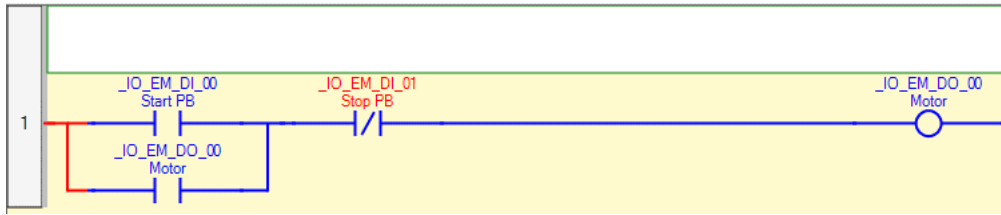
Individual Boolean instructions are red if the instruction result is TRUE/ON or blue if the result is FALSE/OFF. The rung starts from the left power rail with the color red, indicating TRUE/ON, and flows to the right. Any FALSE/OFF instruction, which is blue, interrupts the flow. There must be at least one complete red path to the output for the output to be TRUE/ON.

1. Toggle the simulator board switch **SW11** ON and OFF. Observe the `_IO_EM_DI_00` Direct Contact instruction turns red as you toggle on the switch, and then turn blue as you release it (if you toggle and release the switch too fast, you may not see it update in the ladder diagram). Then observe the `_IO_EM_DO_00` Direct Contact and Direct Coil instructions turn red. You should also observe that the output indicator light 0 on the controller is now lit.



This is a typical motor seal-in circuit (and can also be applied in non-motor circuits as well). The Output Coil is turned on using a Direct Contact and then the active state of the Output Coil seals in the circuit. The circuit is unsealed when a Reverse Contact (normally closed) is opened.

2. Toggle the simulator board switch **SW12** ON to turn off the output. Observe the output `_IO_EM_DO_00` on your controller turns off and the corresponding changes in your Ladder Diagram.

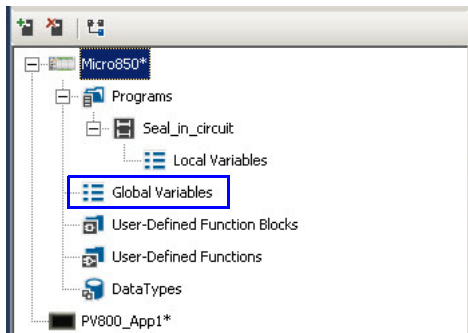


So far, you have monitored your program primarily by viewing real-time changes in the Ladder Diagram editor. In some instances, you may want to view the real-time changes in a list format instead. You can do this by looking at them in the Variables list.

View Real-time Changes in List Format

The variables we are working on are embedded I/O points, and are viewed in the Global Variables list.

1. Double-click **Global Variables** in your Project Organizer.



The Global Variables list launches in a new tab in the main project workspace.

Name	Alias	Data Type	Dimension	Project Value	Initial Value	Comment	String Size
_IO_EM_DO_00	Motor	BOOL	-				
_IO_EM_DO_01		BOOL	-				
_IO_EM_DO_02		BOOL	-				
_IO_EM_DO_03		BOOL	-				
_IO_EM_DO_04		BOOL	-				
_IO_EM_DO_05		BOOL	-				
_IO_EM_DO_06		BOOL	-				
_IO_EM_DO_07		BOOL	-				
_IO_EM_DO_08		BOOL	-				
_IO_EM_DO_09		BOOL	-				
_IO_EM_DI_00	Start PB	BOOL	-				
_IO_EM_DI_01	Stop PB	BOOL	-				
_IO_EM_DI_02		BOOL	-				
_IO_EM_DI_03		BOOL	-				
_IO_EM_DI_04		BOOL	-				
_IO_EM_DI_05		BOOL	-				
_IO_EM_DI_06		BOOL	-				
_IO_EM_DI_07		BOOL	-				
_IO_EM_DI_08		BOOL	-				
_IO_EM_DI_09		BOOL	-				
_IO_EM_DI_10		BOOL	-				
_IO_EM_DI_11		BOOL	-				
_IO_EM_DI_12		BOOL	-				
_IO_EM_DI_13		BOOL	-				

- Locate the `_IO_EM_DO_00` embedded I/O variable in the Global Variable list, observe that the Logical Value checkbox is not selected.

Name	Alias	Logical Value	Physical Value	Initial Value
<code>_IO_EM_DO_00</code>	Motor	<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_01</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_02</code>		<input type="checkbox"/>	<input type="checkbox"/>	

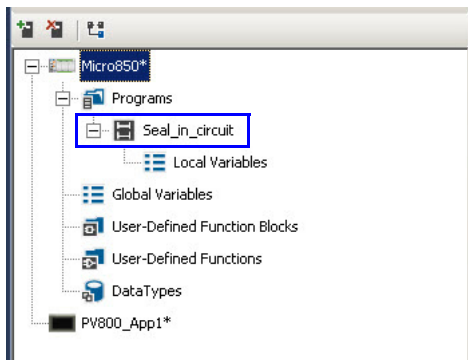
- Toggle the simulator board switch SW12 OFF, then toggle SW11 ON and OFF. Observe the Logical Value checkbox for `_IO_EM_DO_00` is now selected, and the output light on the controller is on. You may have also observed a checkmark appear in the `_IO_EM_DI_00` Logical Value checkbox as you toggled the switch on, and the checkmark disappear as you released the switch.

Name	Alias	Logical Value	Physical Value	Initial Value
<code>_IO_EM_DO_00</code>	Motor	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<code>_IO_EM_DO_01</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_02</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_03</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_04</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_05</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_06</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_07</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_08</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DO_09</code>		<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DI_00</code>	Start PB	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<code>_IO_EM_DI_01</code>	Stop PB	<input type="checkbox"/>	<input type="checkbox"/>	
<code>_IO_EM_DI_02</code>		<input type="checkbox"/>	<input type="checkbox"/>	

- Toggle the simulator board switch SW12 ON. Observe the checkmark in the Logical Value checkbox for `_IO_EM_DO_00` disappears and the light on the output indicator turns off.

You have now completed the real-time monitoring of your program.

Double-click **Seal_in_circuit** to return back to your program view.



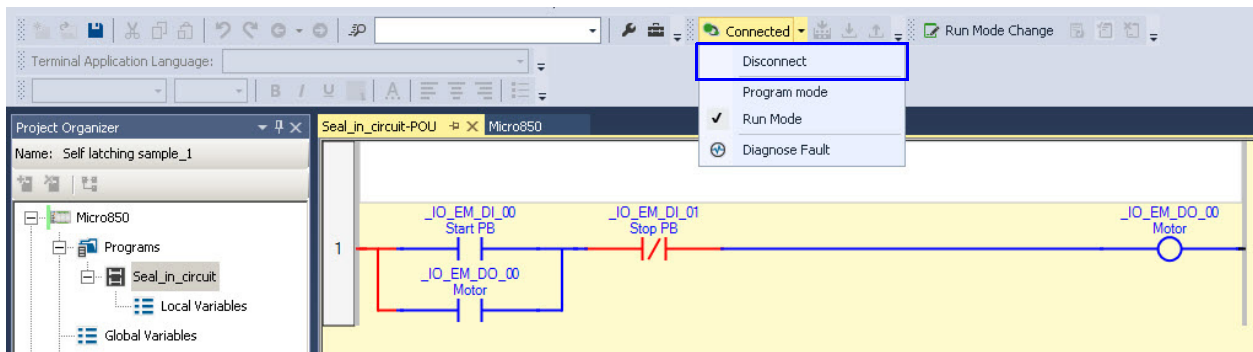
Notes:

How to Create Variables

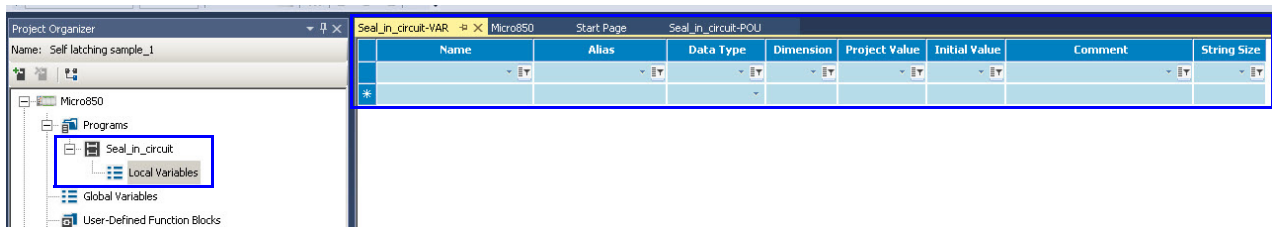
In this chapter you will learn how to create variables to use in your program. The variables you create here will be used in the next chapter.

Create Local Variables for your Program

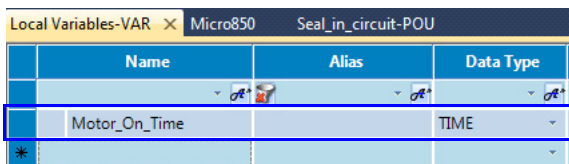
1. If your controller is connected, make sure to disconnect before proceeding.



2. Double-click **Local Variables** in your Seal_in_circuit program to open the Variables tab.



3. Create a variable called **Motor_On_Time** with the TIME data type.



4. Create another variable called **Motor_On_Time_ms** with the INT data type and with an initial value of “5000” (5 seconds).

Name	Alias	Data Type	Dimension	Project Value	Initial Value	Comment	String Size
Motor_On_Time		TIME					
Motor_On_Time_ms		INT			5000		

5. Create a variable called **Motor_Timer** with the TON data type.

Name	Alias	Data Type	Dimension	Project Value	Initial Value	Comment	String Size
Motor_On_Time		TIME					
Motor_On_Time_ms		INT			5000		
Motor_Timer		TON			

6. A TON data type is actually the data structure of a Timer-on-Delay Instruction Block. Instruction Blocks shall be discussed in the next chapter.

You have completed creating variables to be used in the next chapter.

How to Implement an Instruction Block

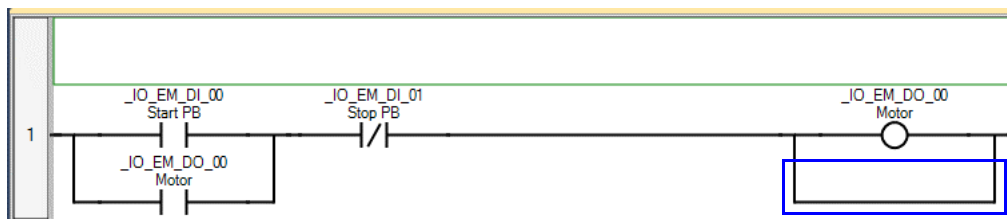
An Instruction Block is essentially a function block that has been predefined to perform a specific task or function. Instruction Blocks include functions such as Timer-on-delay, Timer-off-delay, Math instructions, Data-type conversions, Motion instructions, and so forth.

In this chapter, you will learn how to implement a Timer-On-Delay Instruction Block (TON). This instruction block is inserted into your motor circuit and turns on the motor coil which automatically turns off after five seconds.

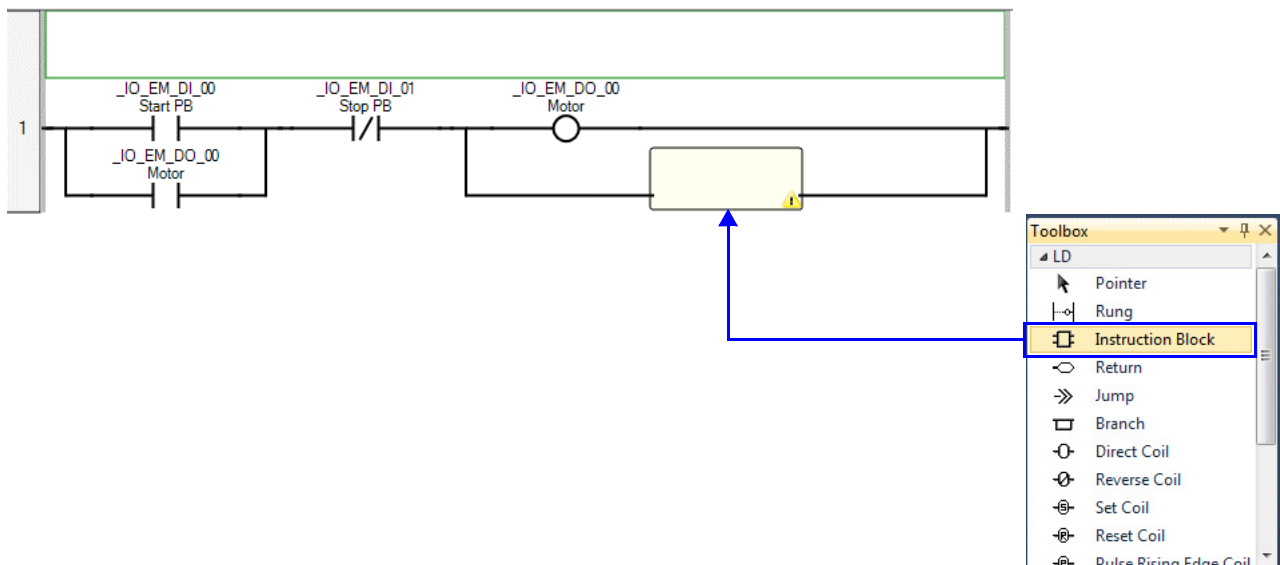
You will also learn how to implement an ANY_TO_TIME Data Conversion Instruction Block to convert an Integer to a Time value.

Add a TON Instruction Block

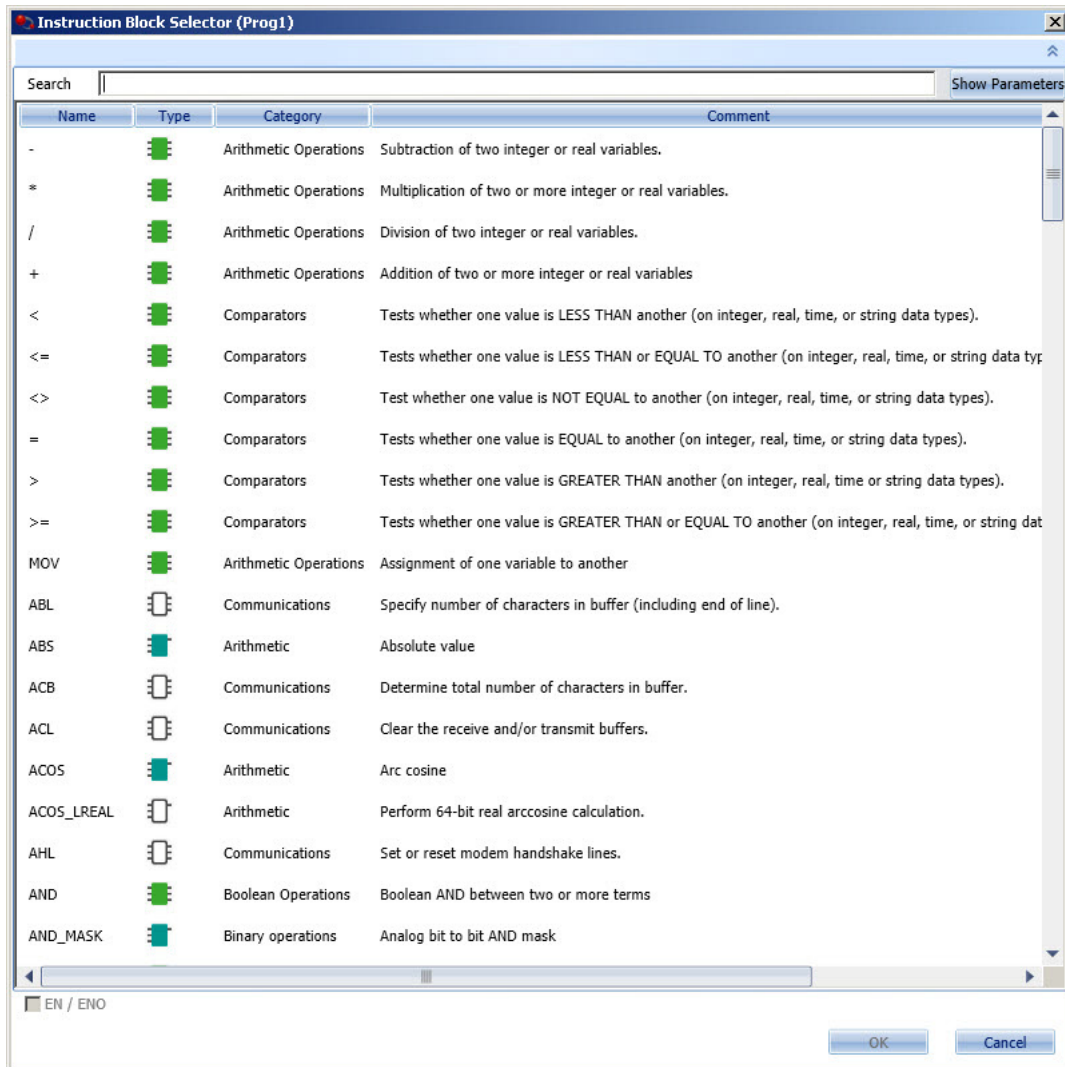
1. Drag-and-drop a **Branch** instruction to the drop point on the left side of the output coil, wrapping around the coil instruction. If you drop the branch on the right side of the rung, it will not wrap around the coil.



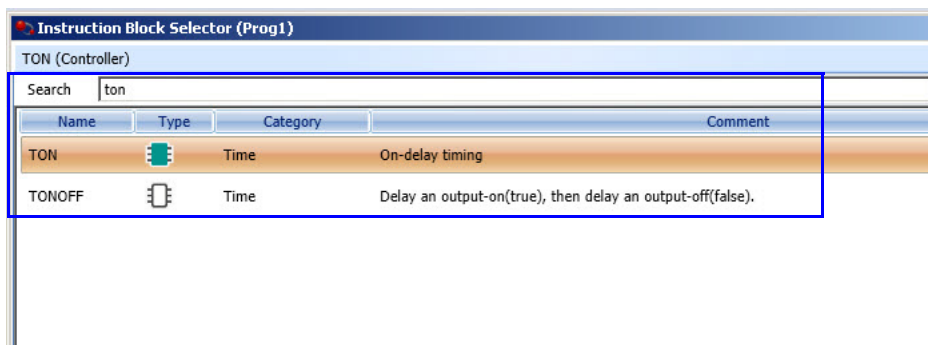
2. Locate the **Block** instruction in the Toolbox. Drag-and-drop the instruction into the branch you have just added.



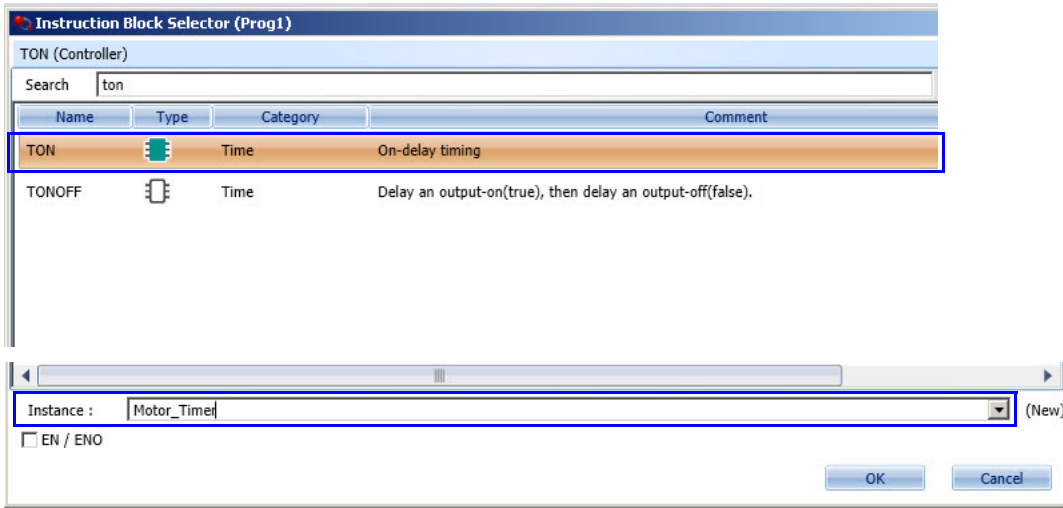
The Instruction Block Selector dialog box appears. This is where you can select the type of instruction block you want to use. As shown, there are many different types of instruction blocks that you can choose from.



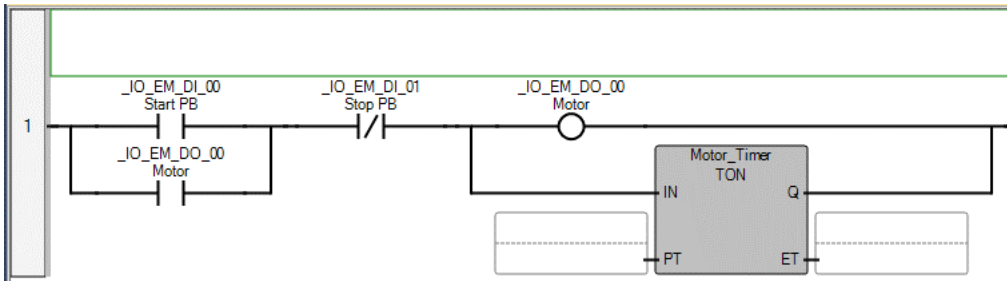
- You can filter the instruction blocks by Name, Category, or Type. Since we want to use a Timer-On-Delay instruction block, type “TON” in the Search filter box. This will filter the choices to only Instruction Blocks that start with TON.



4. Select the **TON** Instruction Block – this is the Timer-on-Delay. Next, click the **Instance** combo box pull-down, and select the variable instance, **Motor_Timer** that you created in the previous chapter, then click OK.



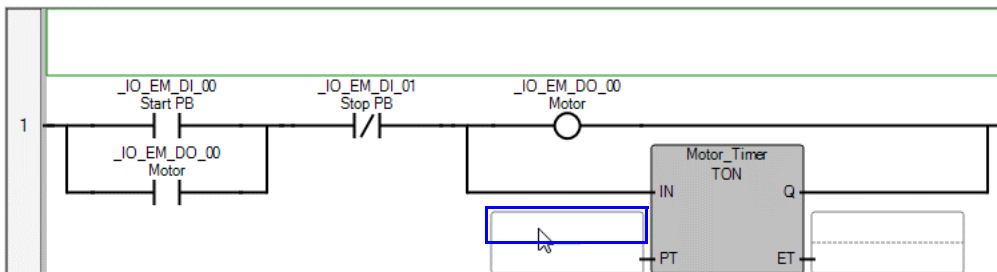
Your ladder program should look like this.



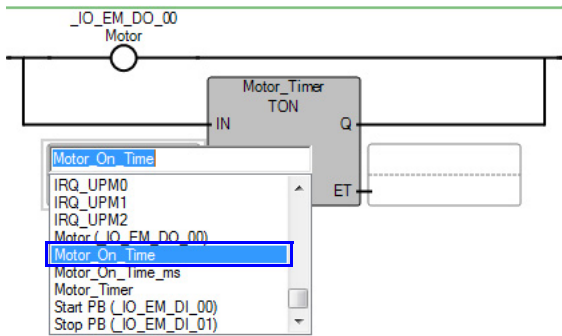
Enter Parameters for TON Instruction Block

Next, let us fill in the parameters for the TON instruction block.

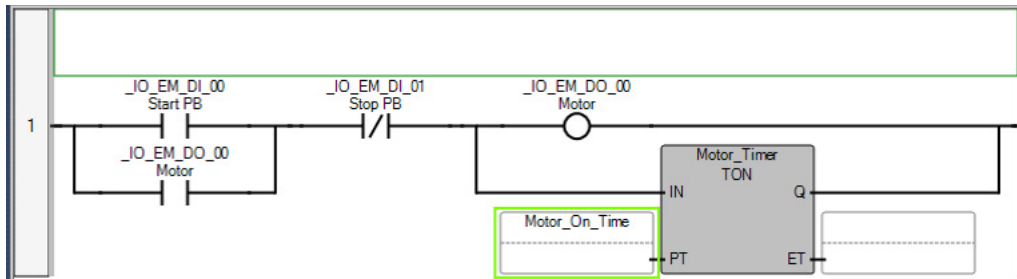
1. Hover your mouse cursor over the blue box next to the PT parameter of the Motor_Timer TON instruction. A light blue highlighted box appears.



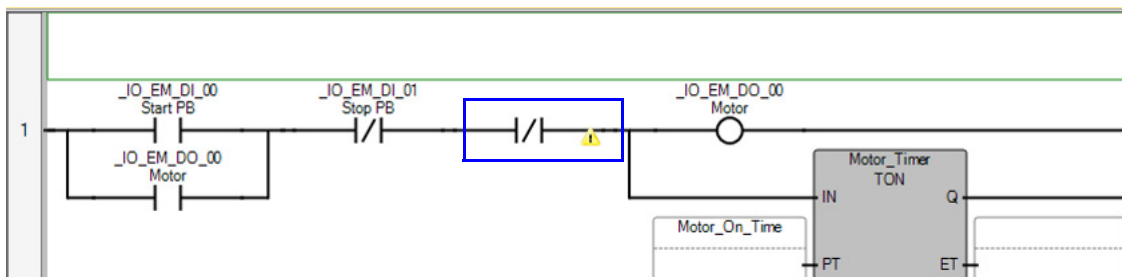
- Click this box and a pull down combo box appears. Find and select the variable **Motor_On_Time**, then press the Enter key. Alternatively, you can double-click the box below the blue box to bring up the Variable Selector.



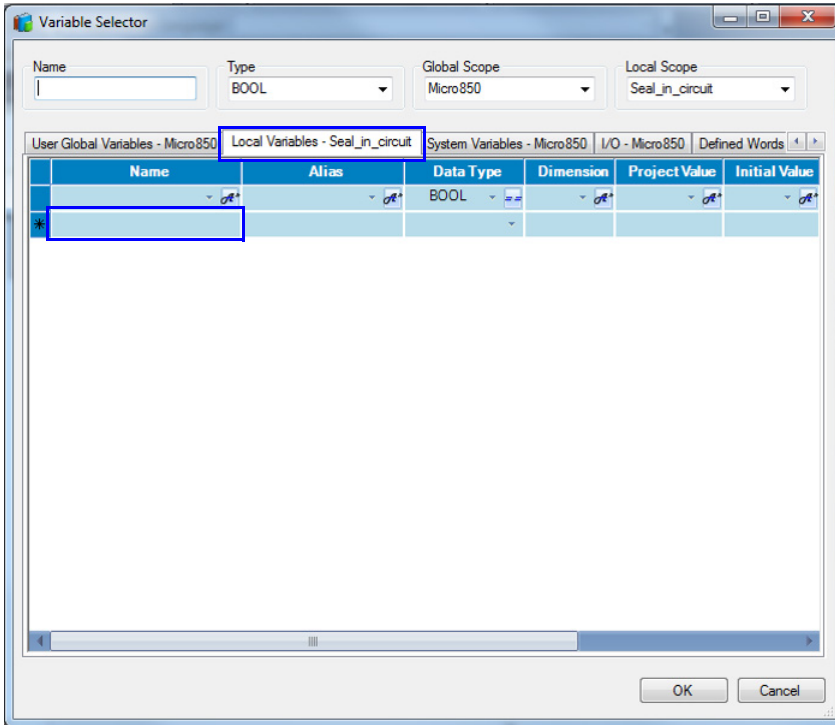
Your program should look like this.



- Insert a **Reverse Contactor** instruction after the `_IO_EM_DI_01` Reverse Contactor.

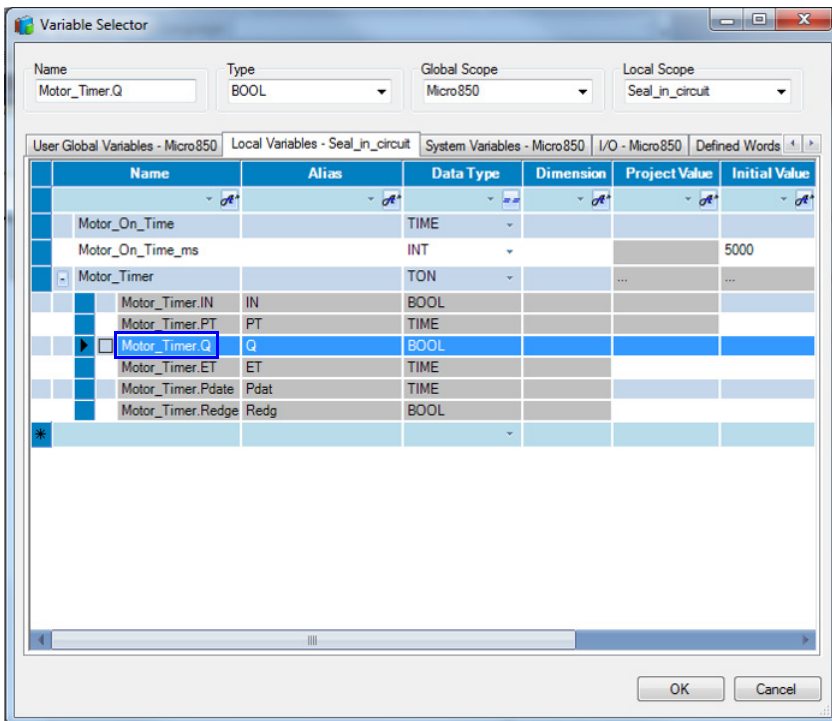


- The Variable Selector dialog box appears. Select the **Local Variables - Seal_in_circuit** tab, then click the empty cell as shown below.

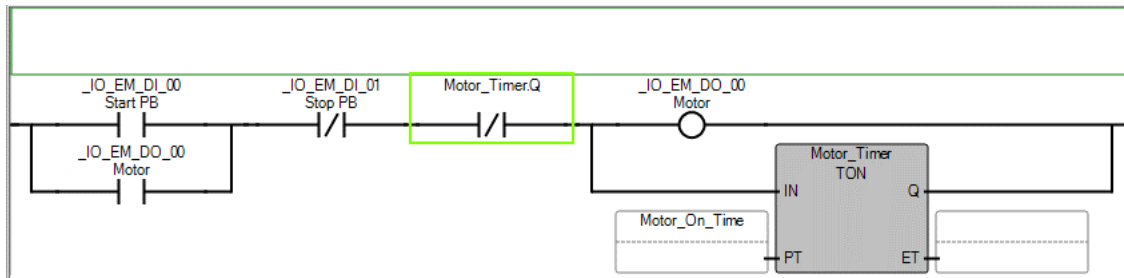


- Expand the variable **Motor_Timer** and select **Motor_Timer.Q**, then click OK.

The **Timer.Q** is an output bit from the **Timer** instruction that turns on when the programmed time has elapsed.



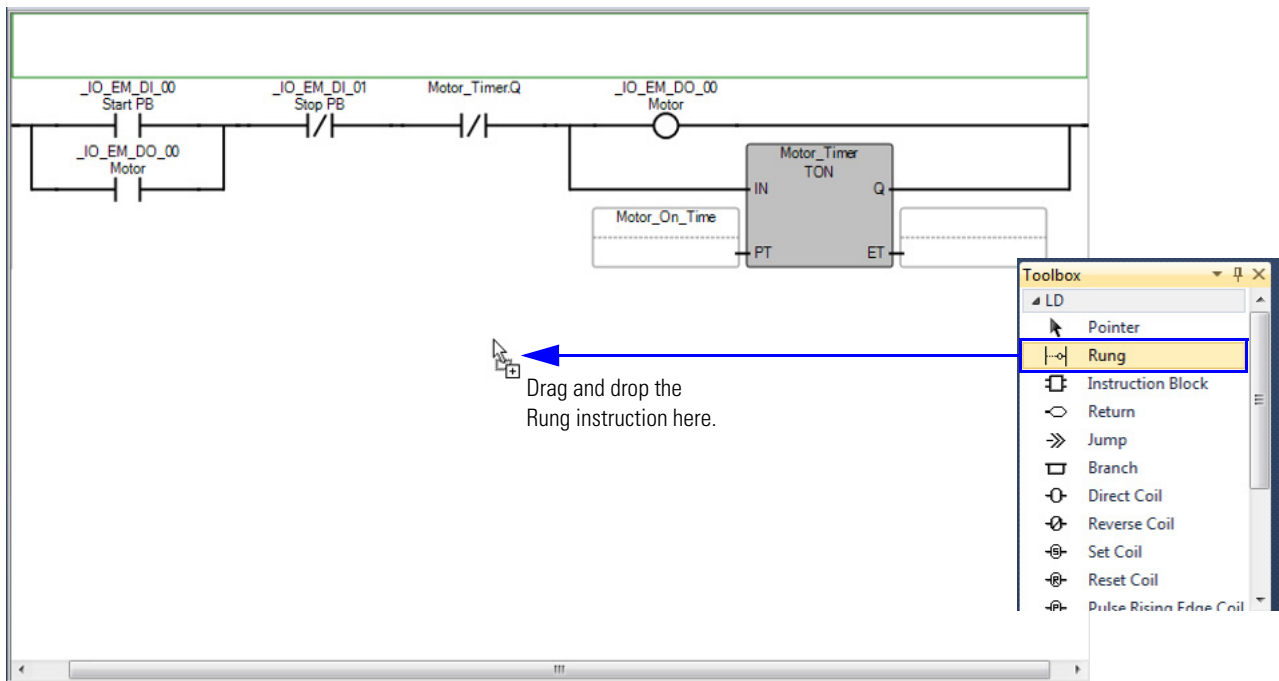
Your program should look like this.



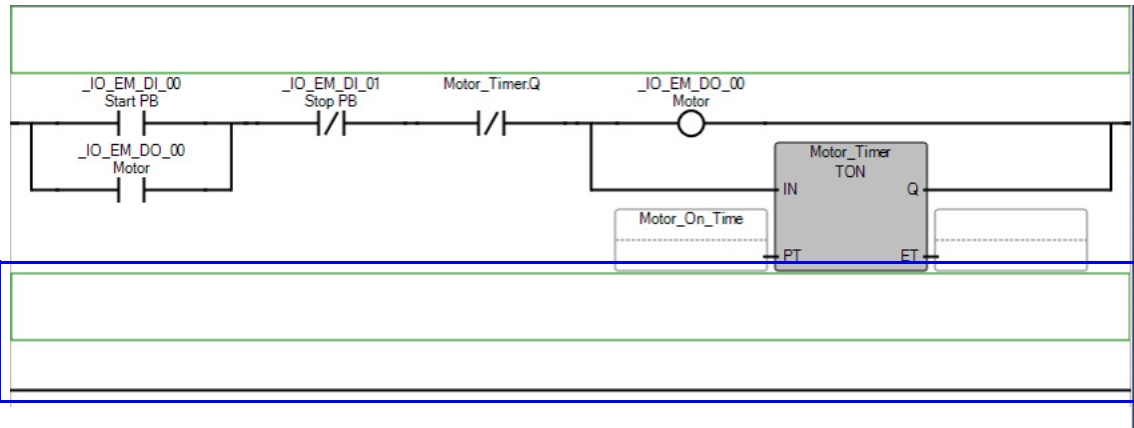
Add an ANY_TO_TIME Instruction Block

Next, let us add a new rung below this existing rung.

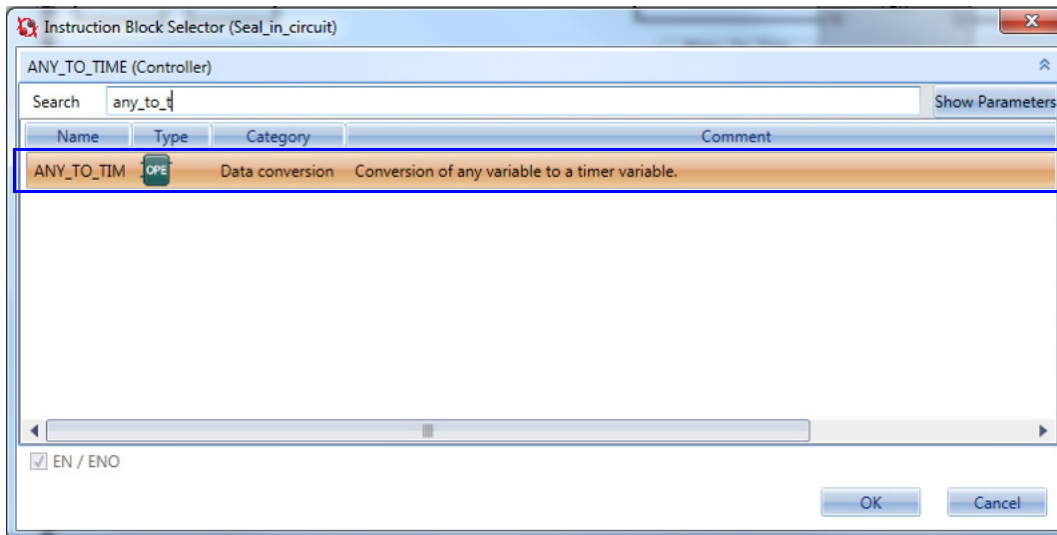
1. Locate the **Rung** instruction in the Toolbox. Drag-and-drop the instruction below Rung 1.



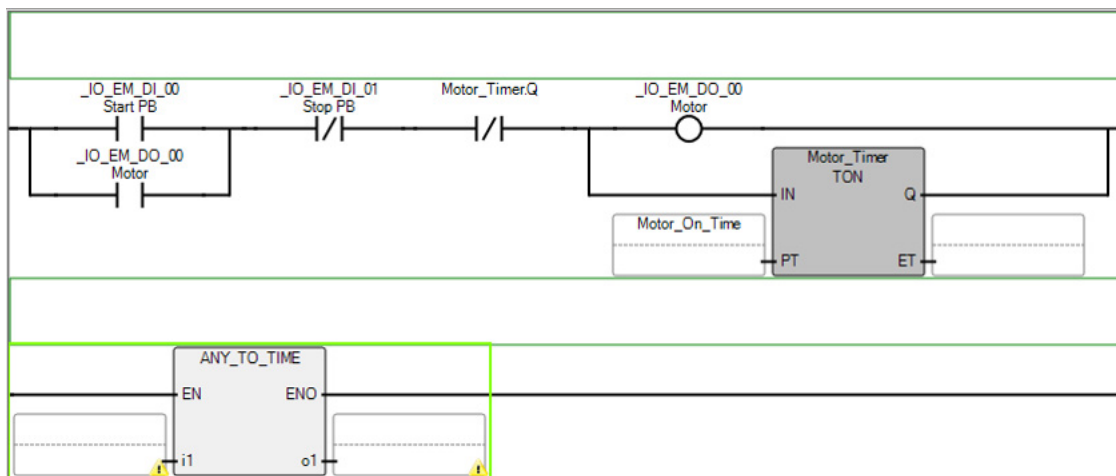
Your program should look like this.



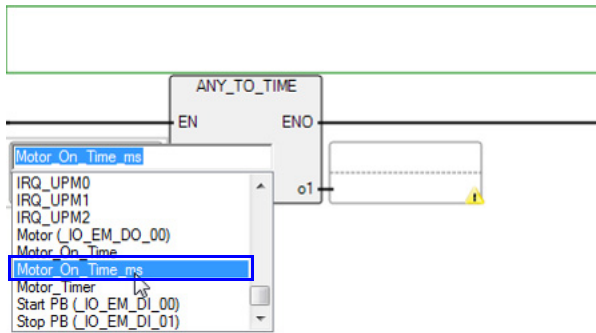
2. Insert a Block instruction onto the new rung. In the Instruction Block Selector dialog box, search for the **ANY_TO_TIME** Instruction Block, select it, then click OK.



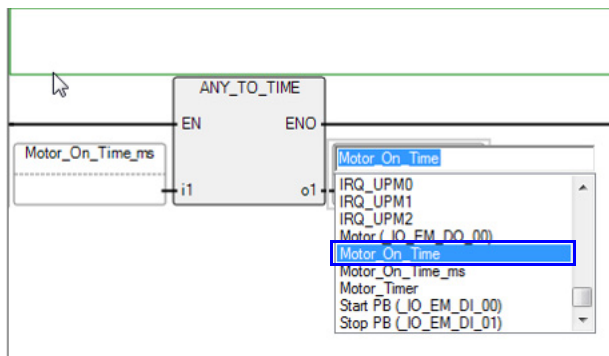
Your program should look like this.



3. Select the variable **Motor_On_Time_ms** for the i1 parameter.

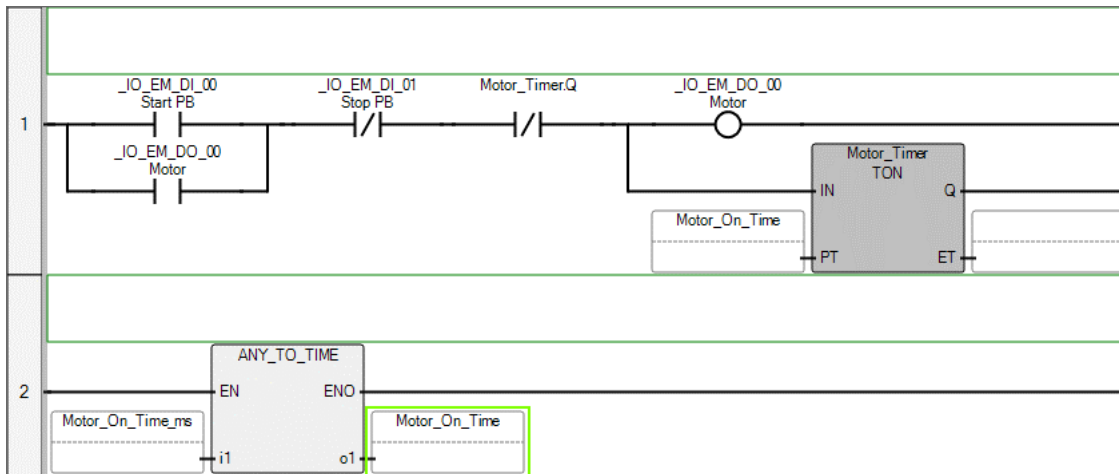



4. Select the variable **Motor_On_Time** for the o1 parameter

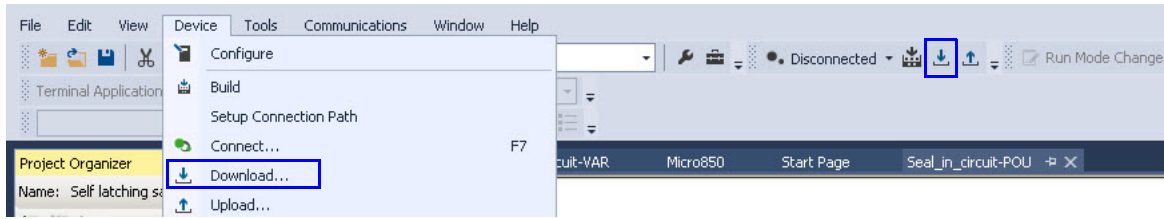


The ANY_TO_TIME instruction block is used to convert an integer value into a time value that is used as the preset time for the Motor_Timer. The integer value represents time in milliseconds.

Your program should look like the following.

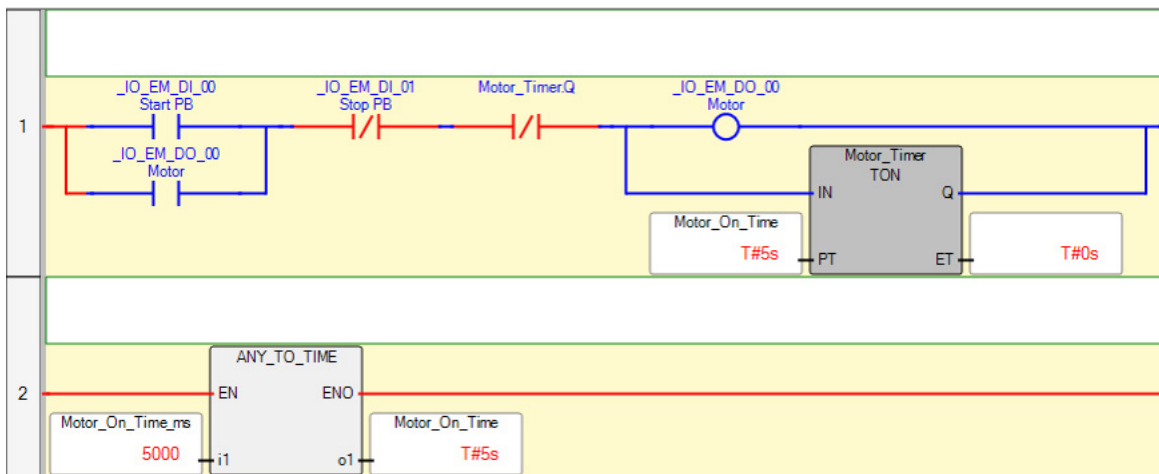


- You can now download the program to the Micro800 controller. You can click on the Download icon , or click Device Tab -> Download.



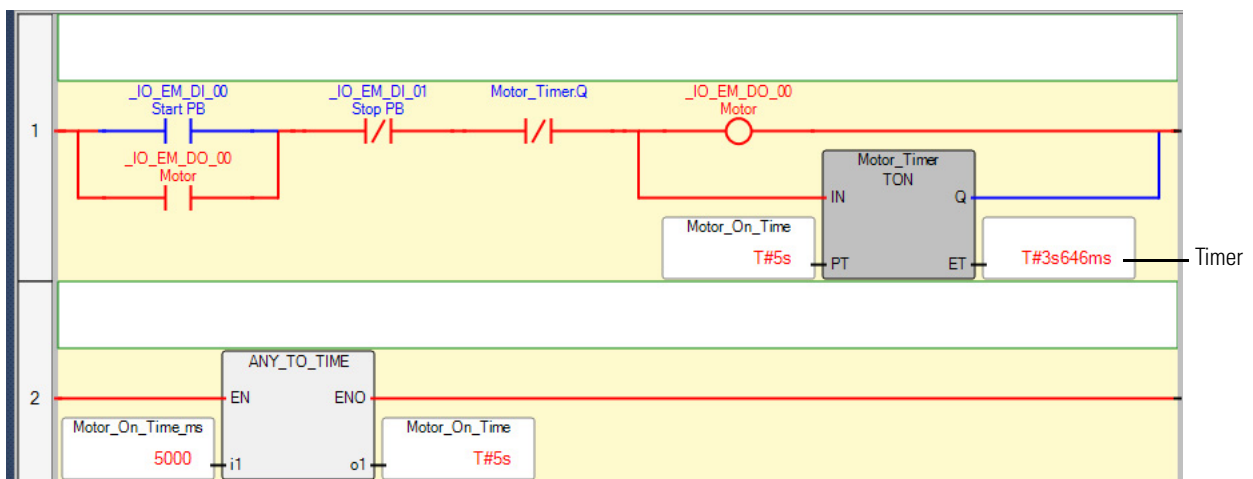
- After completing the download, put your controller to remote run mode and test your program by toggling the simulator board switch **SW11** ON and OFF.

Your program should look like this before toggling the switch SW11 (`_IO_EM_DI_00`).

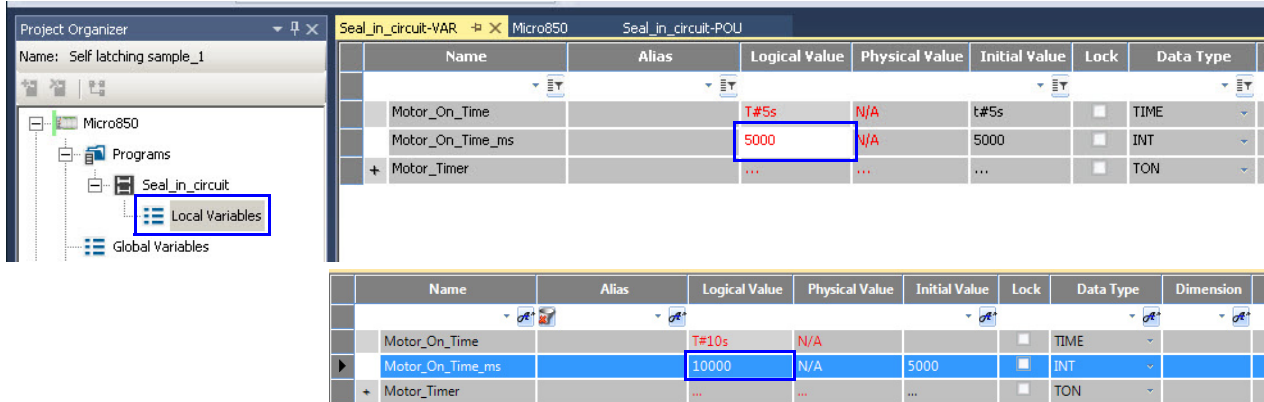


- Toggle the switch **SW11** ON and observe the DO0 light turns on. After five seconds, the light should turn off.

Output indicator 0 light is on while the timer is running.



8. You can change the value of the variable **Motor_On_Time_ms**, to change the amount of time the light stays on to 10 seconds (remember we enter the value in milliseconds). Make sure to press the Enter key after changing the value.
9. Double-click Local Variables. Type in “10,000” under Logical Value and press the Enter key.



10. Toggle the switch **SW11** ON. The DO0 light should now stay on for 10 seconds before turning off.

You have learned how to implement a TON and ANY_TO_TIME instruction block in your program.

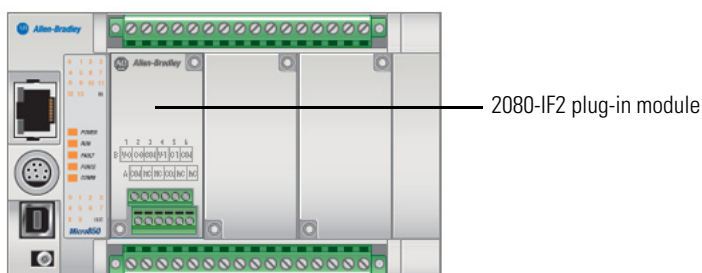
How to Add a Plug-in Module

In this chapter you will learn how to add an analog input plug-in module (2080-IF2). A plug-in module is a module that you can plug into the Micro800 controller chassis to allow you to add additional I/O or communications options to your controller.

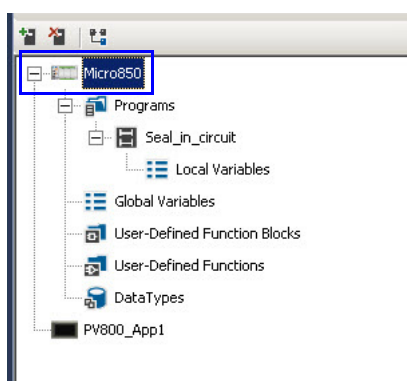
Add a Plug-in Module to the Micro800 Controller

1. Disconnect from the controller (if you are currently connected to it).
2. Power off the controller and remove the empty cover from the first slot.
3. Plug in the 2080-IF2 to the first empty slot of the controller.

Micro850 controller

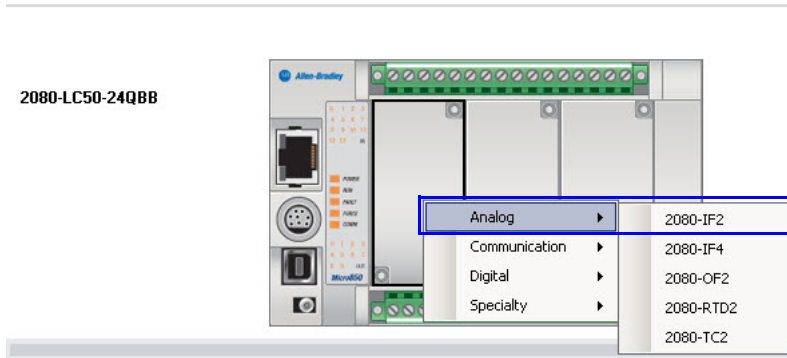


4. Double-click your Micro800 controller in the Project Organizer.

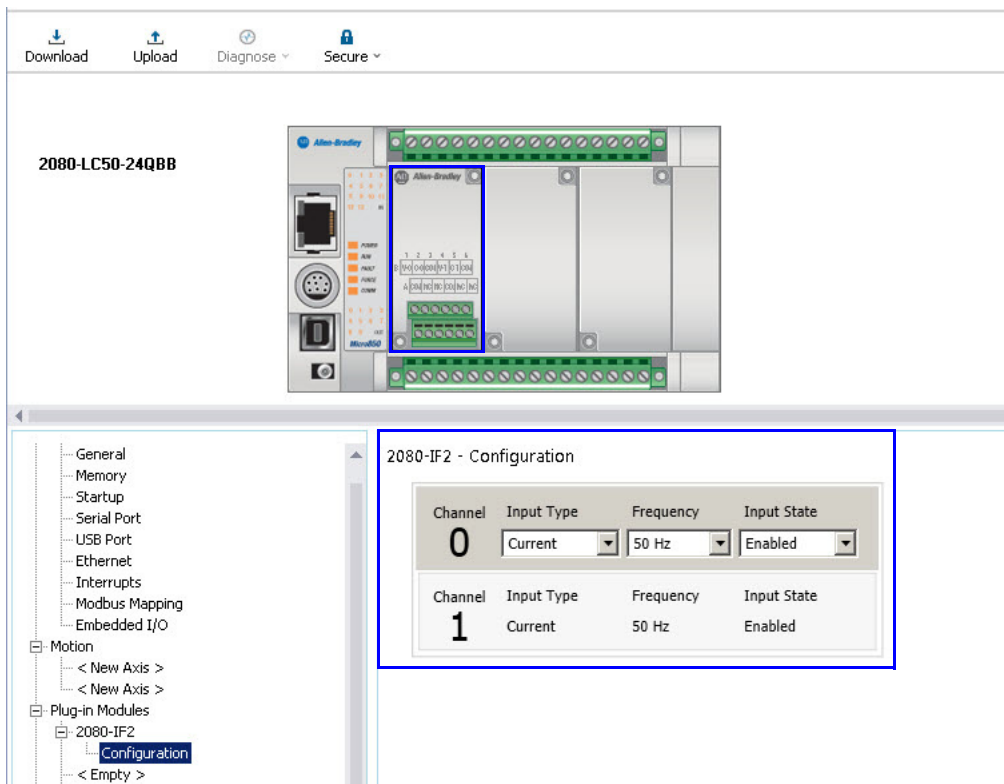


This brings up the General Controller Properties in the main project window.

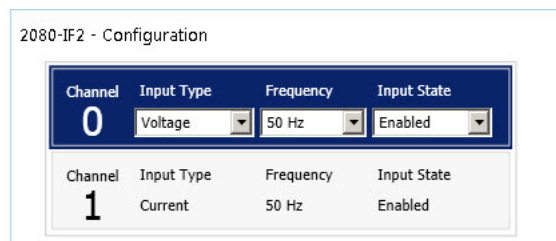
- Right-click the first plug-in module slot, and select 2080-IF2 under Analog.



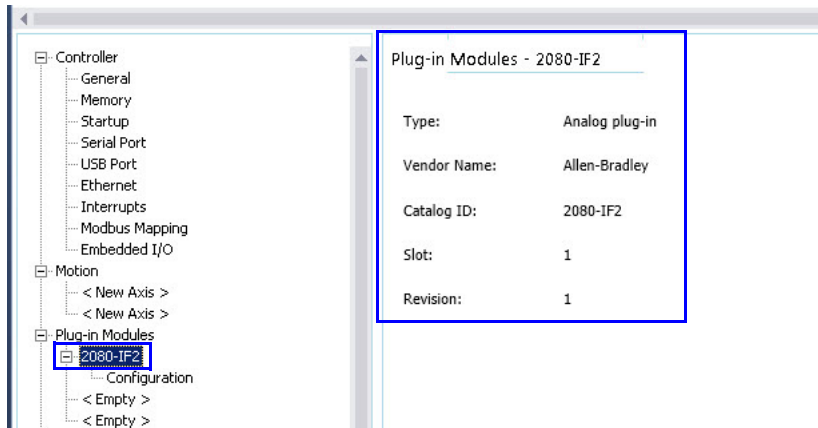
- The 2080-IF2 module is added to the chassis. The configuration properties should also show up in the window below it.



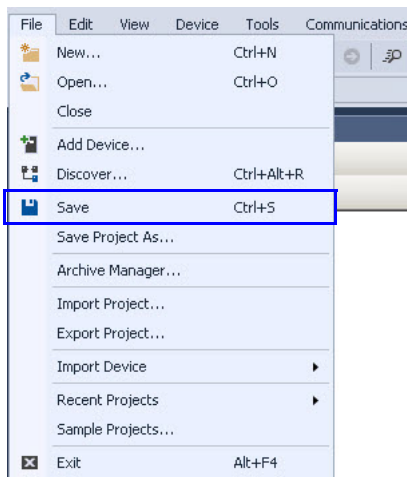
- Configure the Input Type for Channel 0 to “Voltage” and Input State to “Enabled”. Configure the Input State for Channel 1 to “Disabled”.



8. Click 2080-IF2 to show the plug-in module properties.



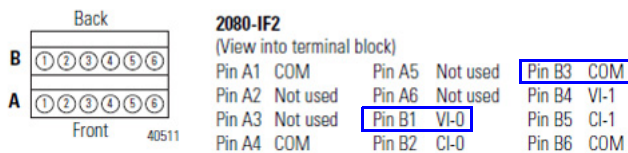
9. Save your project and download it to the controller. You can save your project by selecting File -> Save.



10. Connect the analog voltage output from the simulator board to Channel 0 of the 2080-IF2 module. For this example, we will make use of this to provide analog output voltage input to Channel 0.

11. Connect the analog output to Pin B1 (VI-0) and analog output ground to Pin B3 (COM).

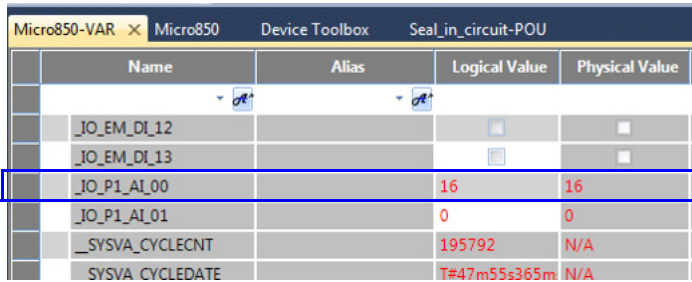
12-Pin Female Terminal Block



12. Double-click **Global Variables** in the Project Organizer.

13. Locate the variable **_IO_P1_AI_00**. This is the raw data value in relation to the voltage that is wired to Channel 0. The value should range from 0...65535 in relation to a 0...10 volt input.

14. On your simulator board, turn the red potentiometer and observe the value of `_IO_P1_AI_00` change.



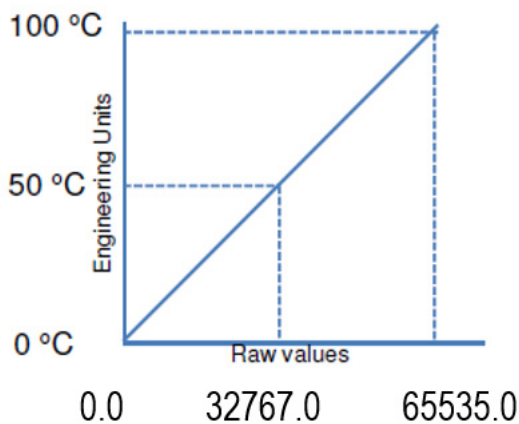
Name	Alias	Logical Value	Physical Value
_JO_EM_DI_12			
_JO_EM_DI_13			
_IO_P1_AI_00		16	16
_IO_P1_AI_01		0	0
_SYSVA_CYCLECNT		195792	N/A
SYSVA_CYCLEDATE		T#47m55s365m	N/A

You have learned how to add and configure a plug-in module using Connected Components Workbench software.

Learn About User-Defined Functions (UDF) and User-Defined Function Blocks (UDFB)

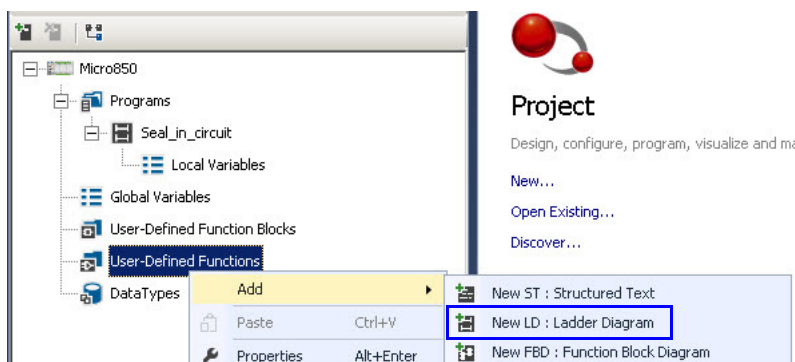
A User-Defined Function (UDF) acts like a subroutine in RSLogix 500 and Studio 5000 Logix Designer. It requires less memory but it supports only one instance and allows only one output parameter. A User-Defined Function Block (UDFB) acts like an add-on instruction in Studio 5000 Logix Designer. It supports multiple instances and allows multiple output parameters, but it takes up more memory.

In this chapter, you will be creating a UDF to convert Raw data from channel 0 of the 2080-IF2 Analog Input module to actual engineering units. The raw value will range from 0.0...65535.0 and the engineering units will range from 0...100 °C. UDF is suitable for this type of simple calculation since each call to the UDF is independent and only needs one set of local variables.

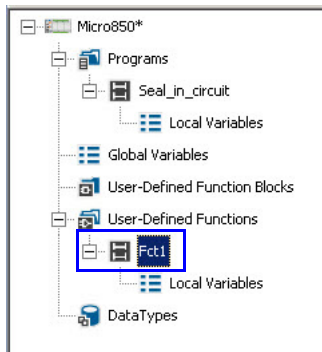


Create a UDF for your Program

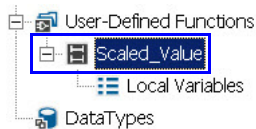
1. In your current project, right-click **User-Defined Function** and select Add -> New LD: Ladder Diagram.



A program called **Fct1** is created under User-Defined Functions.



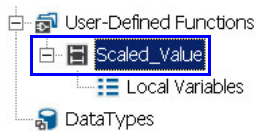
2. Right-click the UDF and rename it to “Scaled_Value”.



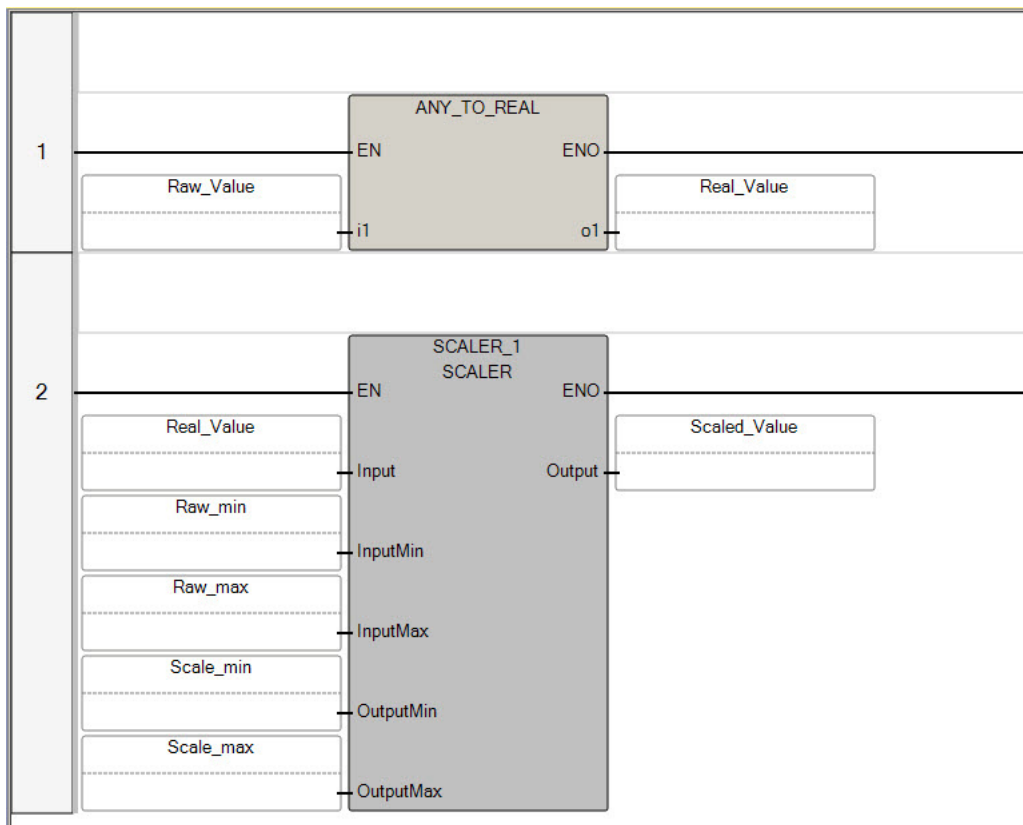
3. Double-click **Local Variables** under the Scaled_Value UDF.
4. Create the following variables. Note to carefully configure the **Direction** property. This property defines whether the variable is an Input, Output, or standard Variable.

Name	Alias	Data Type	Direction	Dimension	Initial Value
Scaled_Value		REAL	VarOutput		
Raw_min		REAL	VarInput		
Raw_max		REAL	VarInput		
Scale_min		REAL	VarInput		
Scale_max		REAL	VarInput		
Raw_Value		UINT	VarInput		
Real_Value		REAL	Var		
*					

5. Next, double-click the **Scaled_Value** UDF to launch the program editor in the main project window.



6. Create the following program.

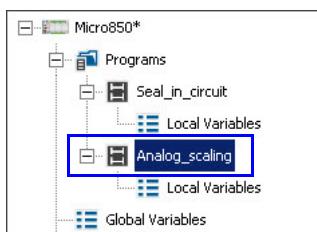


You have completed creating your UDF.

Add the UDF to your Program

To use the UDF in your program:

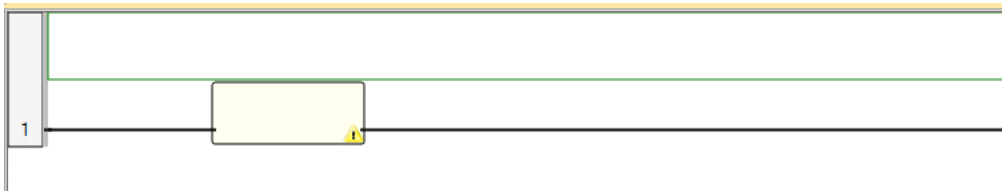
1. Create a new ladder diagram program. Right-click Programs -> Add -> New LD : Ladder Diagram.
2. Rename the new program to “Analog_scaling”.



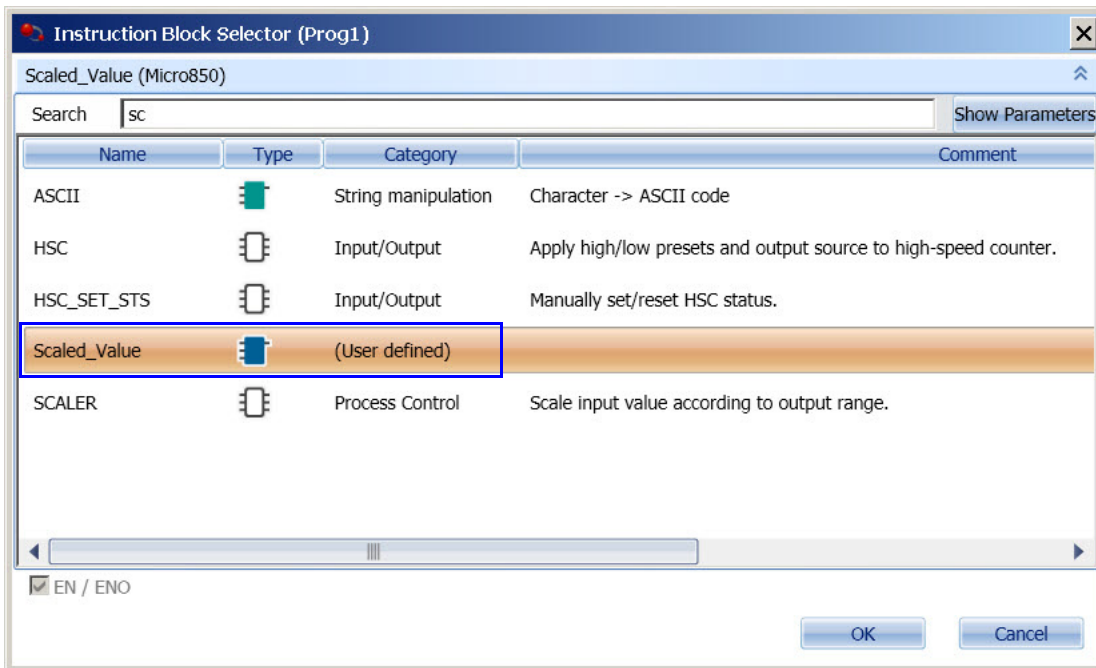
3. Open the Local Variables for the Analog_scaling program, and create the following variables.

Name	Alias	Data Type	Dimension	Project Val	Initial Value
Raw1_min		REAL			
Raw1_max		REAL			
Scale1_min		REAL			
Scale1_max		REAL			
Scale1_value		REAL			
*					

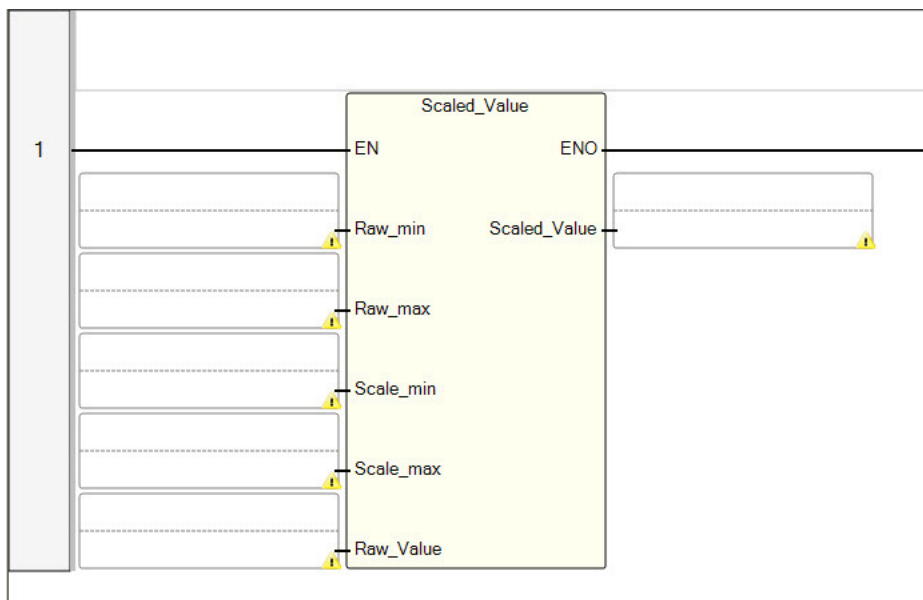
4. Next, open the Analog_scaling program, and add a **Block** Instruction to the first rung.



5. Search for your UDF, select it, and click OK to insert it into your program.



Your program should look like this.



6. Specify the following variables for each parameter of the Block instruction.

Name	Alias	Logical Value	Physical Value	Initial Value	Lock
Raw0_min		0.0	N/A		<input type="checkbox"/>
Raw0_max		65535.0	N/A		<input type="checkbox"/>
Scale0_min		0.0	N/A		<input type="checkbox"/>
Scale0_max		100.0	N/A		<input type="checkbox"/>
Scale0_value		53.29366	N/A		<input type="checkbox"/>

7. Save your project and download it to the controller.

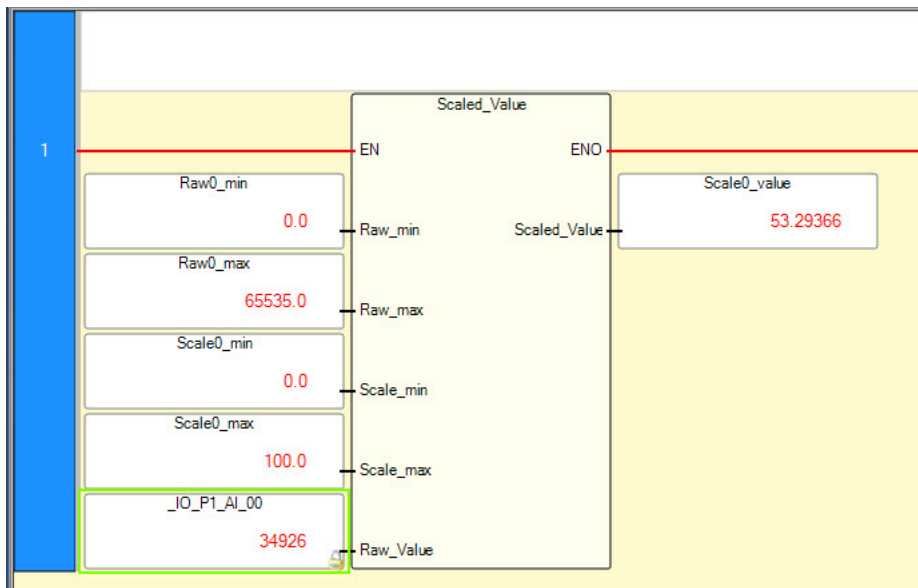
8. Once your download is complete, go to Remote Run Mode.

9. Open the Local Variables of your Analog_scaling program, and set the Logical Values as follows.

Name	Alias	Logical Value	Physical Value
Raw1_min		0.0	N/A
Raw1_max		65535.0	N/A
Scale1_min		0.0	N/A
Scale1_max		100.0	N/A
Scale1_value		49.47433	N/A

10. The Scale1_value shows the converted engineering unit for 2080-IF2 analog input 0 as you adjust the potentiometer.

Your program should look like this.

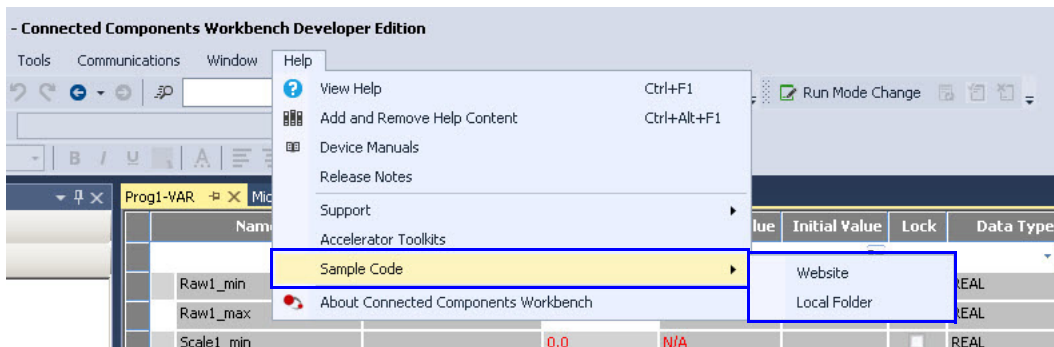


You have now learned how to create and use a UDF in your project.

Get Sample Code from the Rockwell Automation Sample Code Library

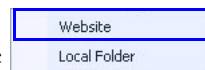
You can find UDFB applications and sample projects from the Rockwell Automation Sample Code Library website, as well as from the sample code folder in your local drive.

You can access the sample code through Help -> Sample Code -> Website or Local Folder.



Get Sample Code from Website

1. Click on Website to go to the Rockwell Automation Sample Code website
2. You can download the applicable UDFBs for your project.



Rockwell Automation LISTEN. THINK. SOLVE.[®]

Sample Code Search

Welcome to the Rockwell Automation Sample Code web site.

Use this free-of-charge site as a "getting started" for configuring and programming Rockwell Automation products. Common design approaches and applications are covered. [More](#)

Search: [Search Tips](#) [Return All Sample Code](#) [Send Feedback](#)

Showing 1 - 20 of 116 | Result Page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [Next](#)

Title	Information & Description	Catalog Numbers	Rating	RA Preferred Code	Downloads	Date	Reviews	Download
Micro800 UDFB: Sets the Micro800 Real Time Clock (RTC) on an Exception Basis		2080-LC*	(Not Rated)	No	34	June 2015	Submit	
Micro800 PowerFlex750 UDFB's		2080-LC50-***, 2080-LC20-***, 20G, 20F	(Not Rated)	No	728	February 2014	Submit	
Micro800 UDFB: Digital IO Plug-In UDFB suite for CCW R2/R3 only		2080-LCXX-*** (The Micro800 with Plug-In Slot), 2080-IQ4, 2080-OB4, 2080-OV4, 2080-OW4, 2080-IQ4084, 2080-IQ40V4	★★★★☆	No	3145	February 2012	Submit Read	
Starter Control & Diagnostics with the Micro800		2080-LC, 2711C, 190E, CCAT,	(Not Rated)	No	78	May 2015	Submit	
Astronomical Clock for Micro800		2080-LCxx	★★☆☆☆	No	1167	August 2014	Submit Read	

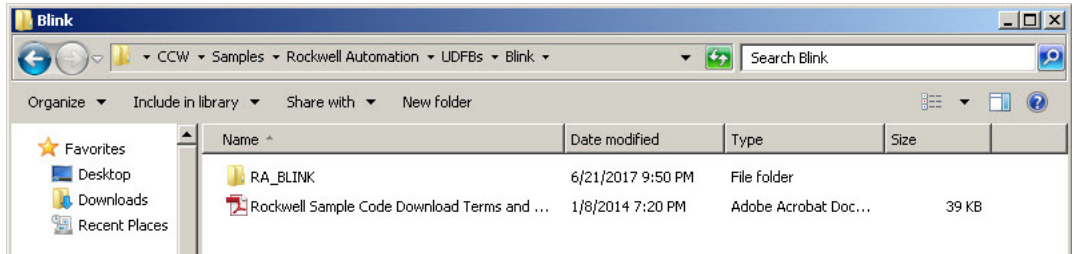
Filter Publication Results:

- RA Preferred Code
 - No (114)
 - Yes (2)
- Product Family
 - Motion Control, Integrated, Programmable Controllers (2)
 - Programmable Controllers (111)
 - Programmable Controllers, Drives (2)
 - Programmable Controllers, Partner Products (1)
- Products
 - Micro800 System (116)

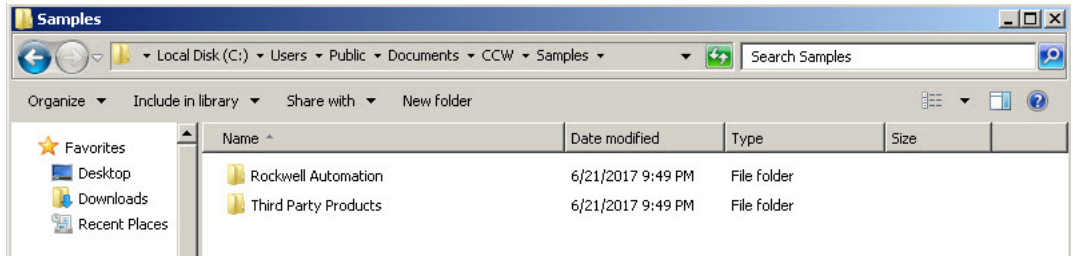
What's New
[Last 30 days](#)

Get Sample Code from Local Folder

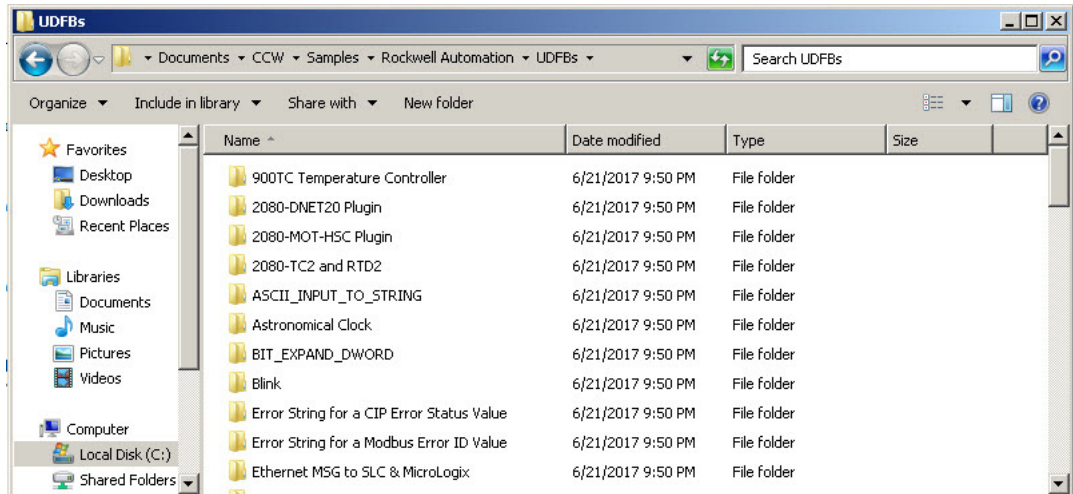
1. Click on Local Folder to bring you to the directory



2. Double-click the Rockwell Automation folder to view the UDFBs or Sample Projects sub folders.

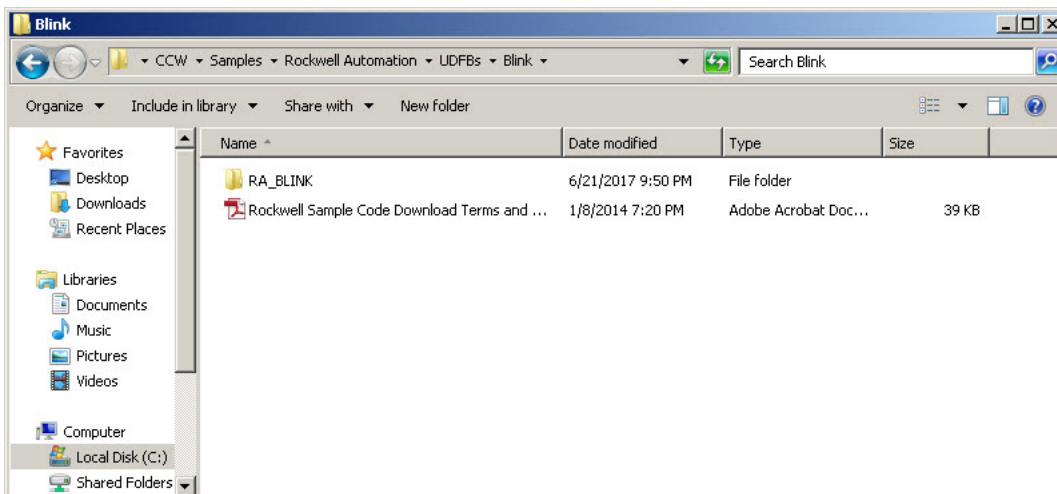


For example, if you double-click the UDFB's folder, you are presented with the list of UDFBs available for your use.



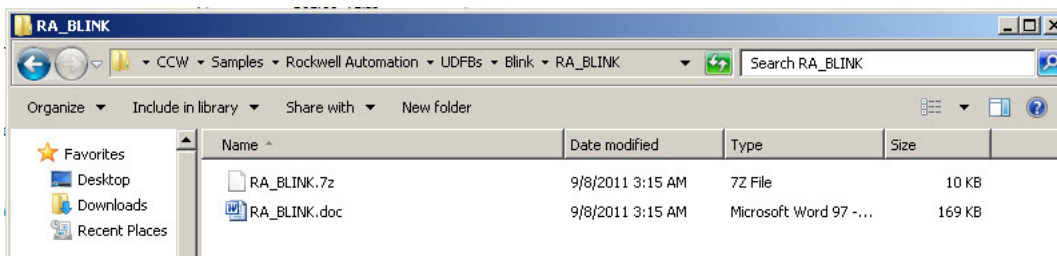
Import Sample Code into Your Project

1. For this example, select the UDFB located under Local Folder -> CCW -> Samples -> Rockwell Automation -> UDFBs -> Blink.

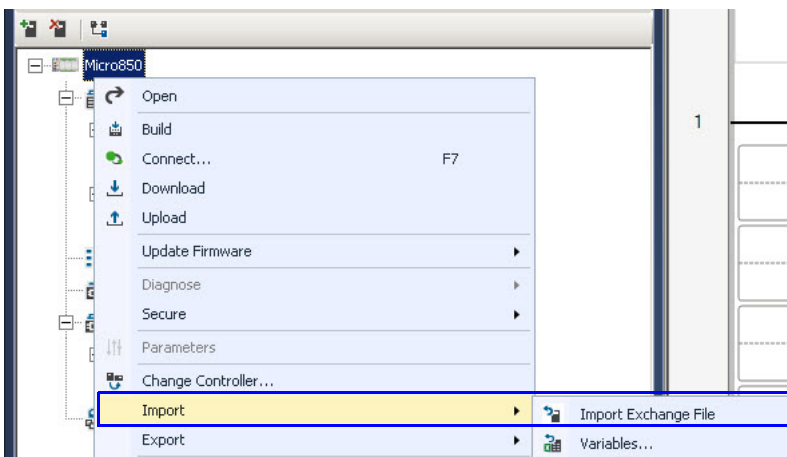


2. Double-click the RA_Blink folder to see the contents.

The contents inside are the import exchange file in 7zip / zip format and a word document to describe the operation of the UDFB.

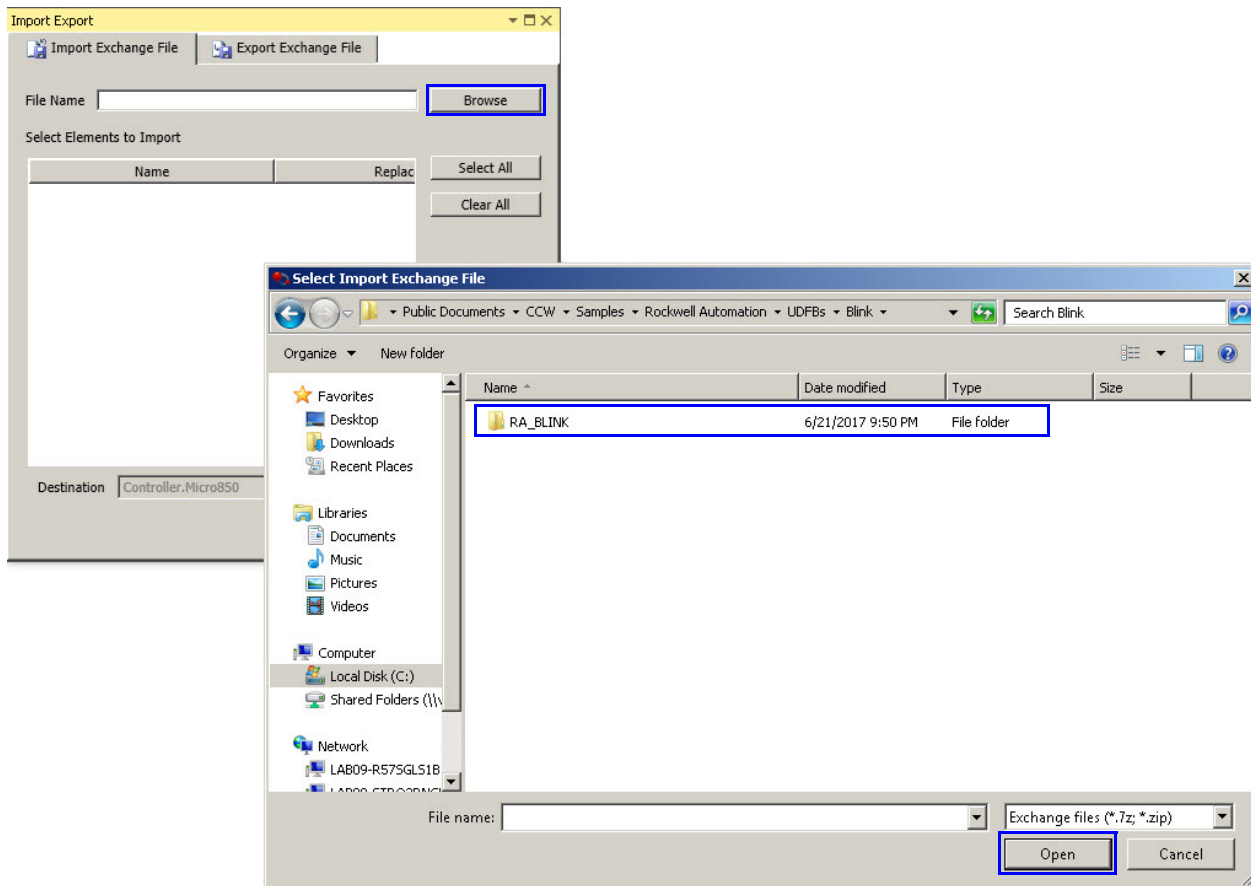


3. Right-click the Micro800 controller and select Import -> Import Exchange File.



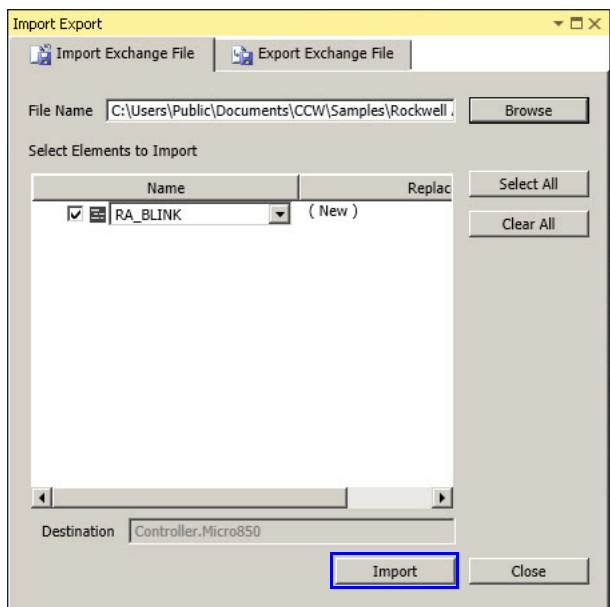
4. Click Browse to locate the directory for the selected UDFB RA_BLINK.

5. Select RA_BLINK.7z and click Open.

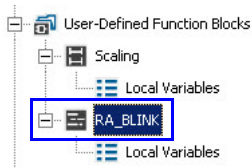


You see the RA_BLINK appearing under the Import window.

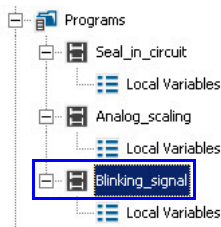
6. Click Import.



The RA_BLINK UDFB is added under the User-Defined Function Blocks.



7. Create a new ladder diagram program called Blinking_signal.



8. Open the Local Variables for the Blinking_signal program, and create the following variables. Observe that the Data Type for the variable is the RA_BLINK UDFB you imported.

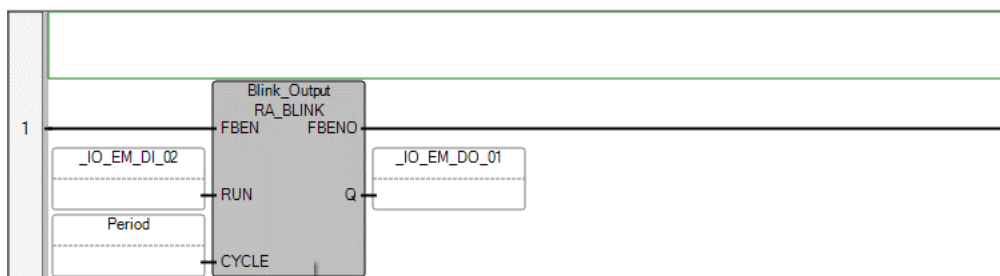
9. Create the following variables.

Name	Alias	Data Type
Blink_Output		RA_BLINK
Period		TIME

10. Open the Blinking_signal program, add a Block instruction to the first rung. Select the RA_BLINK UDFB, specify the Instance Blink_Output, and click OK.

11. Specify the following variables for each parameter of the Block.

Your program should look like this.



12. Save your project and download it to the controller.

13. Refer to the word document of the RA_BLINK UDFB and see the Arguments information for the UDFB.

Arguments:

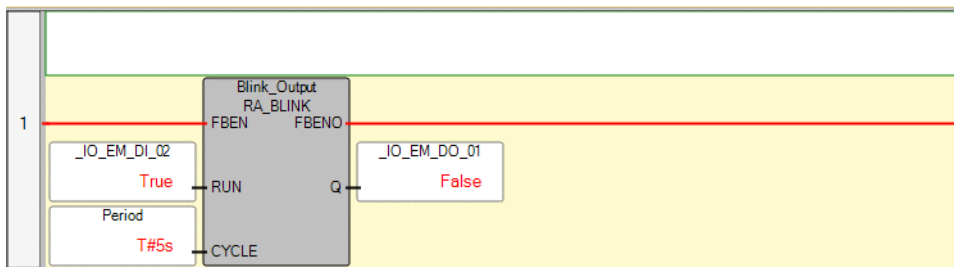
Parameter	Parameter Type	Data Type	Description
FBEN	Input	BOOL	Function block enable When FBEN = TRUE, execute function. When FBEN = FALSE, do not execute function.
RUN	Input	BOOL	Mode: TRUE = blinking / FALSE = reset the output to false.
CYCLE	Input	TIME	Blinking period, must be 500 ms and above.
FBENO	Output	BOOL	Function block enable out.
Q	Output	BOOL	Output blinking signal.

14. Double-click Local Variables under the Blinking_signal program and set the **Period** to 5 seconds.

Name	Alias	Logical Value	Physical Value
Period		T#5s	N/A
Blink_Output	

15. Toggle the simulator board switch **SW13** ON to run the instruction.

16. Observe the **_IO_EM_DO_01** output indicator on the controller turning on and off at five seconds intervals.



You have now learned how to import a readily available UDFB from the Rockwell Automation Sample Code Library from your Local Folder.

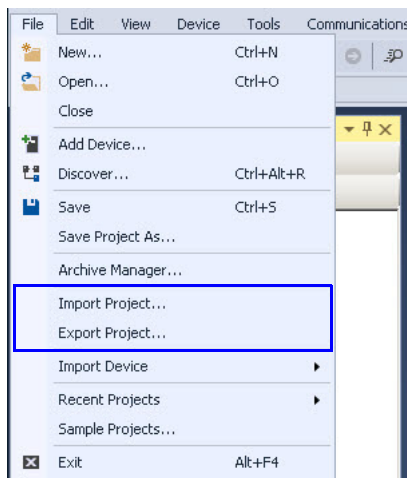
If another output (such as `_IO_EM_DO_02`) is required to independently blink with a different cycle time (for example `T#2s`), then another instance of the UDFB should be created by adding another variable of type `RA_BLINK` (for example `Blink_Output_2`) and calling the instance `Blink_Output_2` on another rung. A UDF cannot easily be used for more than one output since a UDF has only a single set of local variables.

Import and Export Project

Importing and Exporting a project requires Connected Components Workbench software, version 8.00 or later.

You can export the entire project to a single file for easy transfer to another computer. The exported project file (.ccware) is saved in the “My Documents\CCW\Import_Export” folder. The exported file can be copied to another computer and imported back into Connected Components Workbench software.

To access the Import Project or Export Project window, click File -> Import Project or Export Project.



Notes:

Quick Tips

Quick tips when working with Connected Components Workbench software

Keyboard Shortcuts

Shortcut	Description
Working with Rung	
Crtl + 0	Inserts a rung after a selected rung ⁽¹⁾
Crtl + Alt + 0	Inserts a rung before a selected rung ⁽¹⁾
Working with Branch	
Crtl + 1	Inserts a branch after a selected element
Crtl + Alt + 1	Inserts a branch before a selected element
Working with Instructions	
Crtl + 2	Inserts an instruction block after a selected element
Crtl + Alt + 2	Inserts an instruction block before a selected element
Working with Contacts	
Crtl + 3	Inserts a contact after a selected element ⁽²⁾
Crtl + Alt + 3	Inserts a contact before a selected element ⁽²⁾
Working with Coils	
Crtl + 4	Inserts a coil after a selected element ⁽²⁾
Crtl + Alt + 4	Inserts a coil before a selected element ⁽²⁾
Spacebar	For coils or contacts, toggles between the available types
F1	Invokes Context Sensitive Help

⁽¹⁾ When no rung is selected, a rung is added at the end of the rung list.


⁽²⁾ When a branch is selected, an element is inserted at the end of the branch.

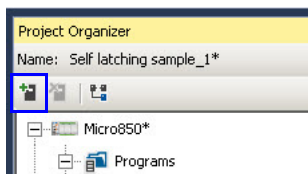
Notes:

PanelView 800 HMI Design Using Connected Components Workbench

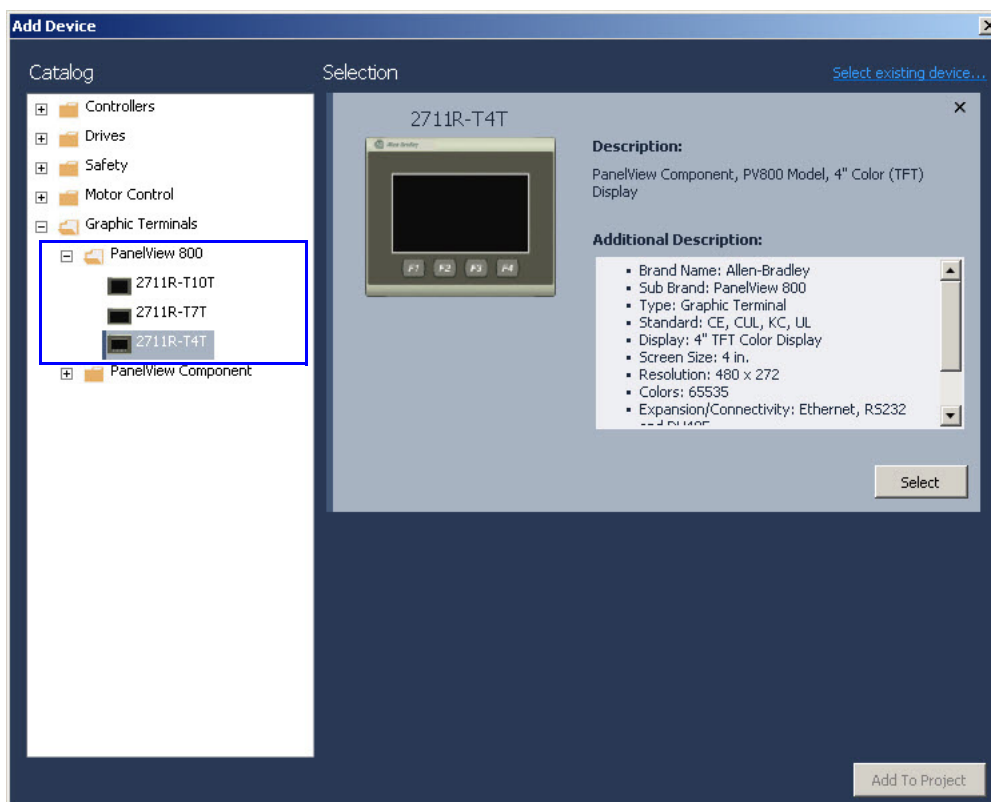
Add a PanelView 800 Terminal to Your Project

You can add a PanelView 800 terminal to an existing Connected Components Workbench (version 8.00 or later) project. For this example, we continue to work on the same project.

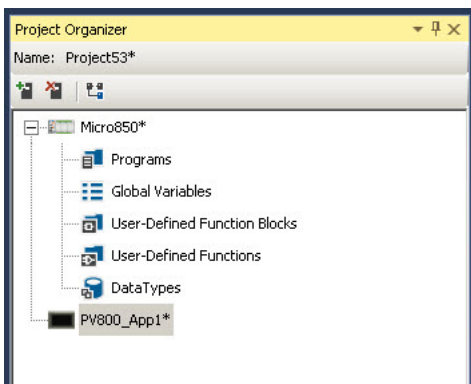
1. If you have not added a PanelView 800 terminal, click the **Add Device**  icon located in Project Organizer.



2. In the Add Device dialog box, select a PanelView 800 terminal under Graphic Terminals.



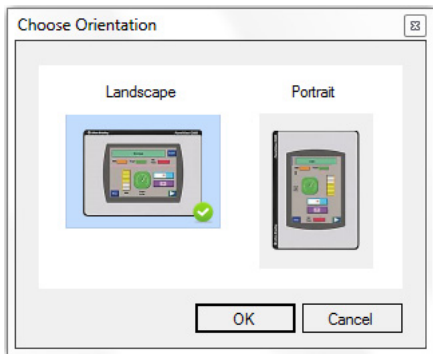
3. Click Select to add the terminal to the shopping cart of devices and then click Add To Project to add the PanelView 800 terminal to your project.



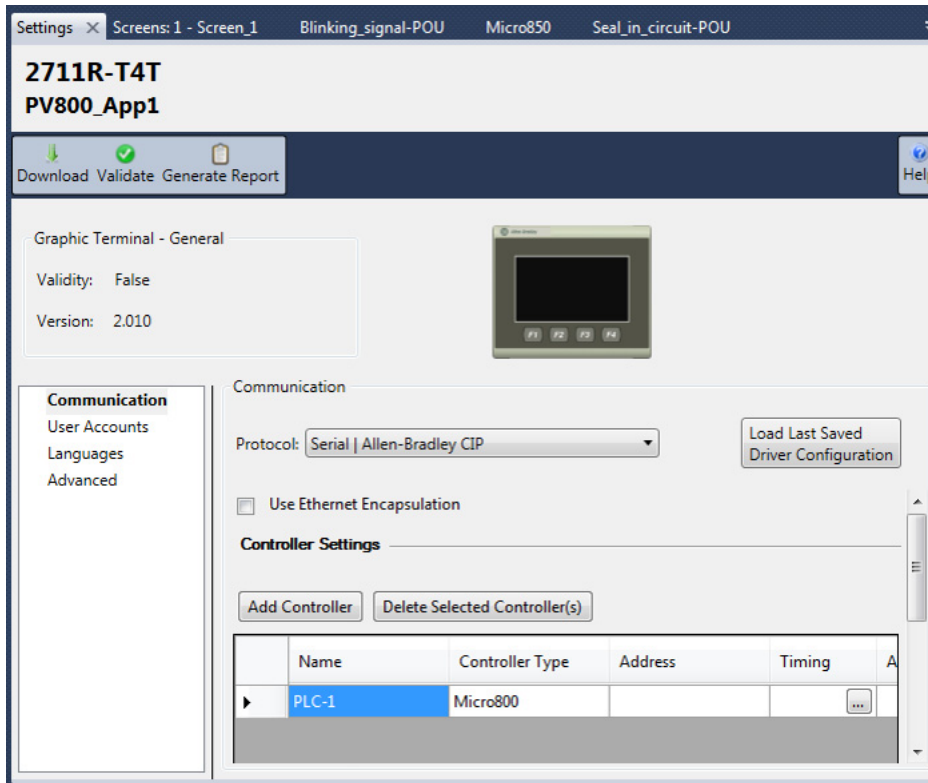
Configure Your PanelView 800 Terminal Communication Settings

1. Double-click **PV800_App1** to launch the HMI Settings tab.
2. For PanelView 800 terminals, you can choose either Landscape or Portrait orientation.

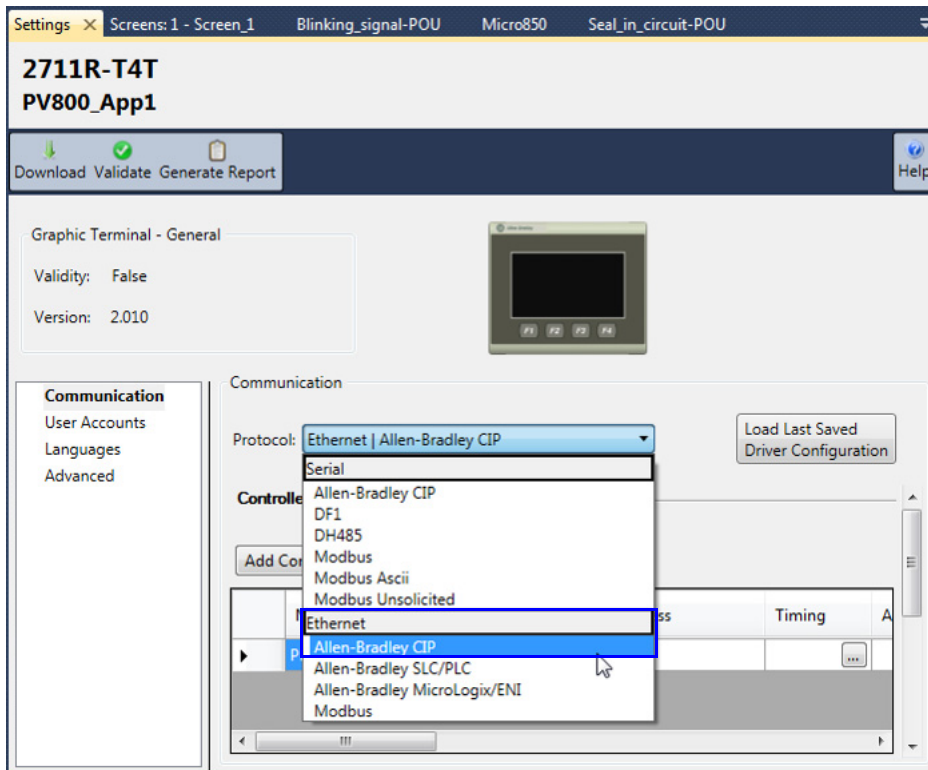
For this example, choose Landscape orientation and click OK.



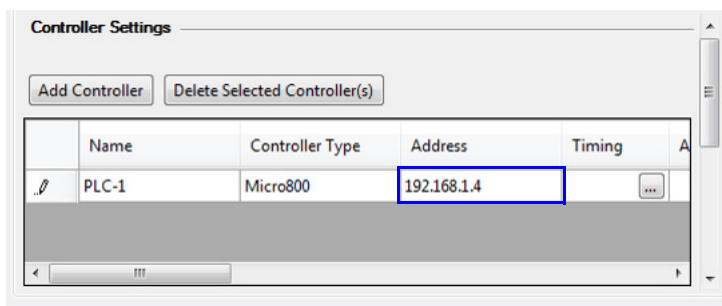
3. The Communication settings tab appears.



4. For the **Protocol** configuration, select “Allen-Bradley CIP” under the Ethernet category from the drop-down box.



5. By default, a controller node has already been created and added to your application, but it is necessary to specify the IP Address of the controller. For this example, set the IP Address to “192.168.1.4”.



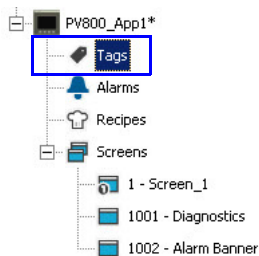
You have now learned how to add and configure a PanelView 800 terminal to your project.

Using HMI Tags

HMI tags bring back data from a Micro800 controller to the terminal. The Tag Editor tab is where you create, view, and modify tags used by your HMI application. The tag types are external, memory, system, and global. Each type has a different data source.

In this chapter you will learn how create a few individual HMI tags. These tags can either be typed in manually, or they can be copied and pasted from an external location, such as Microsoft Office Excel.

- Double-click Tags in your Project Organizer to launch the HMI Tag Editor tab.



Note that the tags from the controller that are to be added to the Tag Editor have to be created in the Global Variables scope.

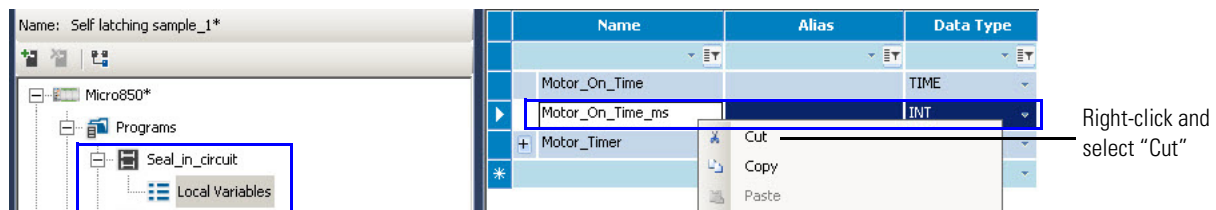
Use the project that has at least the following programs created earlier:

- Seal_in_circuit
- Analog_scaling

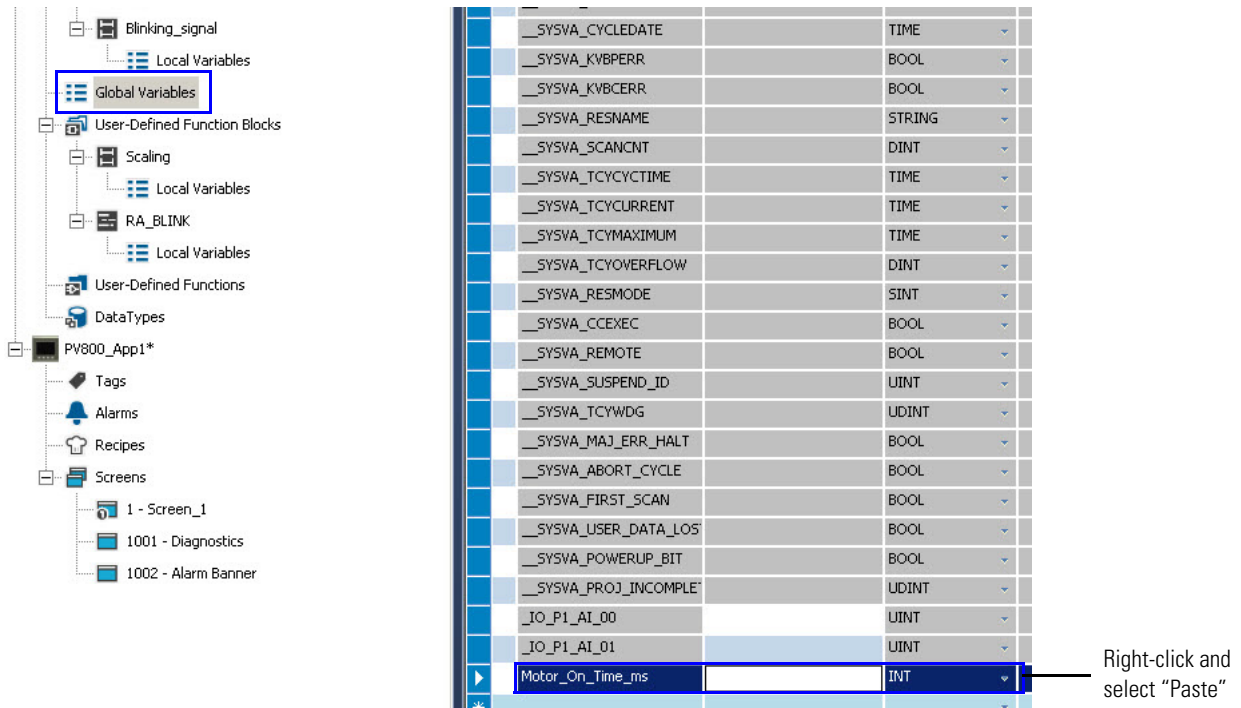
Create Global Variables

Create the following Global Variables to be used as HMI tags for your PanelView 800 terminal.

1. Go to the Local Variables for the Seal_in_circuit program. Right-click the Motor_On_Time_ms variable and select **Cut**.

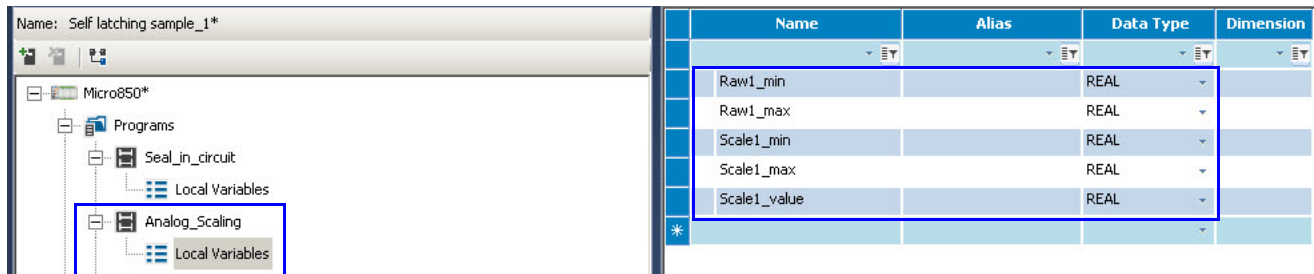


2. Double-click Global Variables, right-click and select **Paste**.



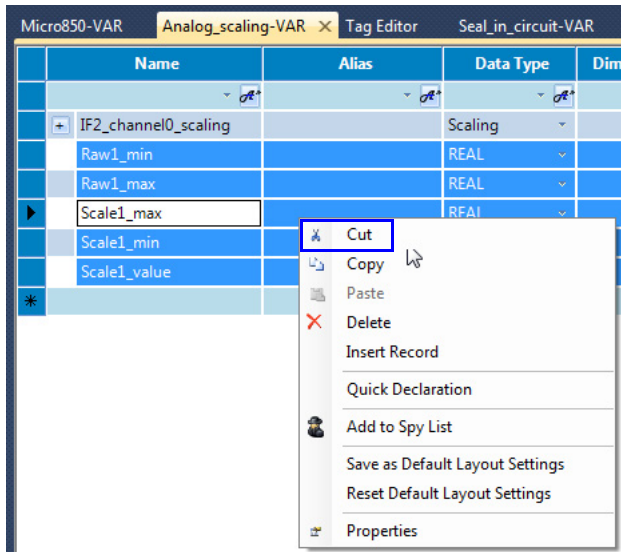
This tag is used to set the motor on time.

3. Go to the Local Variables for the Analog_scaling program, Hold down the “Shift key + arrow down key” to select the five variables shown below.

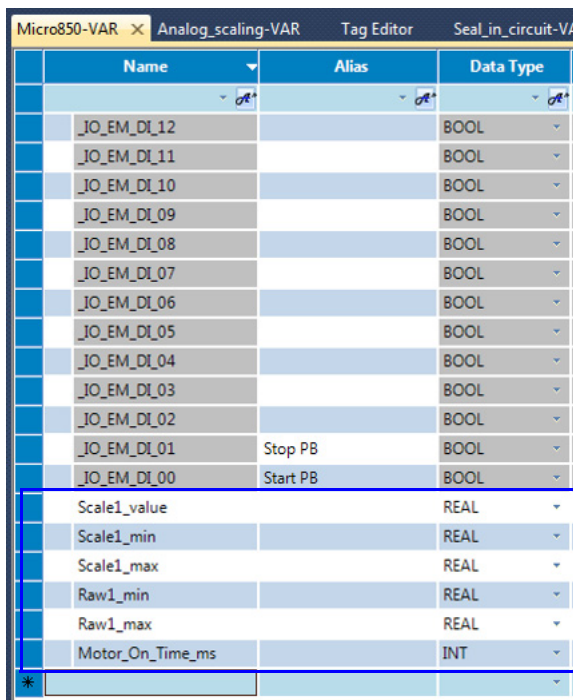


These tags are used to input the analog input raw value range for the 2080-IF2 channel 0 and the engineering units range.

4. Right-click and select **Cut**. Then paste these tags under Global Variables.



The Global Variables should have the following tags.



5. Add two global variable tags as shown.

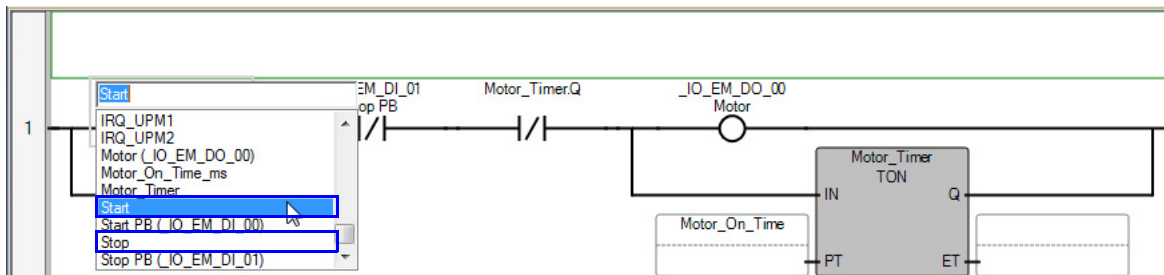
Name	Alias	Data Type
_IO_EM_DI_10		BOOL
_IO_EM_DI_09		BOOL
_IO_EM_DI_08		BOOL
_IO_EM_DI_07		BOOL
_IO_EM_DI_06		BOOL
_IO_EM_DI_05		BOOL
_IO_EM_DI_04		BOOL
_IO_EM_DI_03		BOOL
_IO_EM_DI_02		BOOL
_IO_EM_DI_01	Stop PB	BOOL
_IO_EM_DI_00	Start PB	BOOL
Stop		BOOL
Start		BOOL
Scale1_value		REAL
Scale1_min		REAL
Scale1_max		REAL
Raw1_min		REAL
Raw1_max		REAL
Motor_On_Time_ms		INT
*		

These tags are used to start/stop the motor for the Seal_in_circuit.

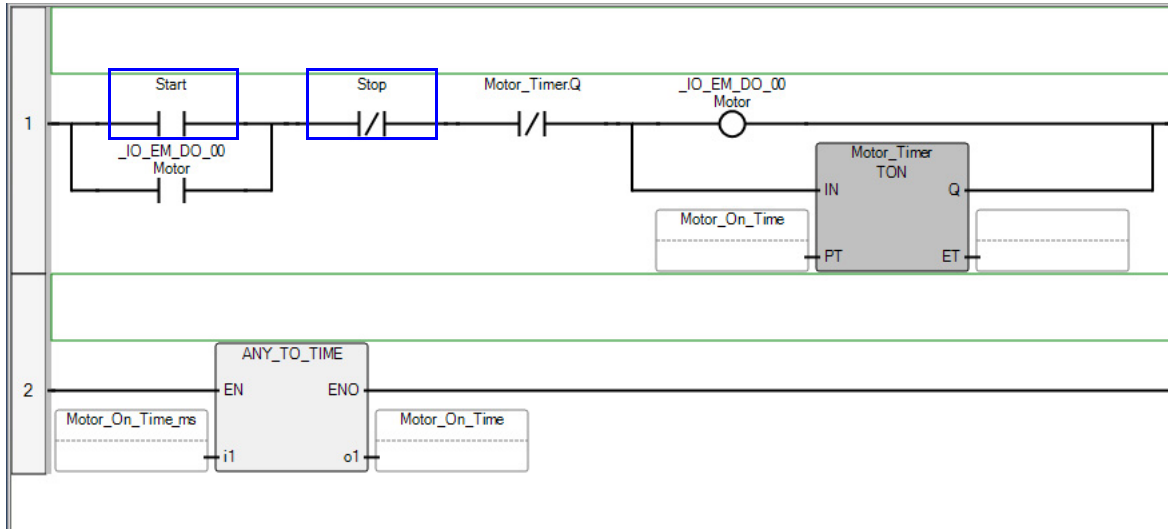
Edit Ladder Diagram Program

Make the following changes to rung 1 of the Seal_in_circuit program as we execute the start/stop control from the PanelView 800 terminal

1. Replace “Start PB (_IO_EM_DI_00)” with the **Start** Global variable and replace “Stop PB (_IO_EM_DI_01)” with the **Stop** Global variable.



Your program should look like this.



2. You should have the following additional variables in the Global Variables list.

Stop		BOOL	▼
Start		BOOL	▼
Scale1_value		REAL	▼
Scale1_min		REAL	▼
Scale1_max		REAL	▼
Raw1_min		REAL	▼
Raw1_max		REAL	▼
Motor_On_Time_ms		INT	▼
*			▼

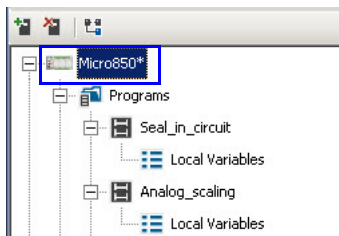
You have now learned how to create HMI tags for your PanelView 800 terminal.

Notes:

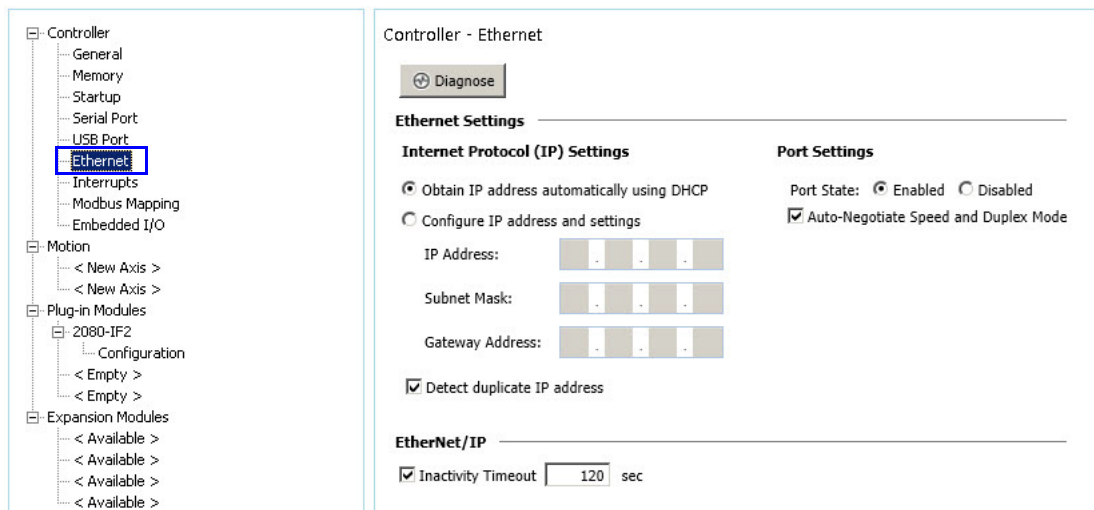
Configure Micro800 Controller Ethernet Settings

In this chapter you will learn how to configure the IP address for the Micro800 controller to prepare for communicating with the PanelView 800 terminal.

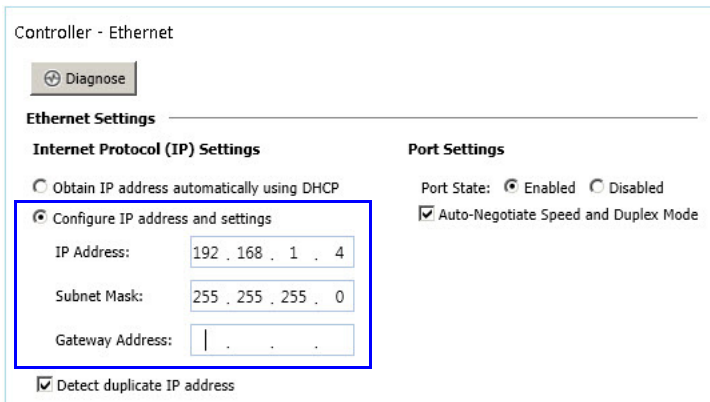
1. Double-click your Micro800 controller under the Project Organizer to bring up the General Controller Properties in the main project window.



2. Select **Ethernet** to bring up the Ethernet port configuration.



3. Set the Ethernet configuration as shown.



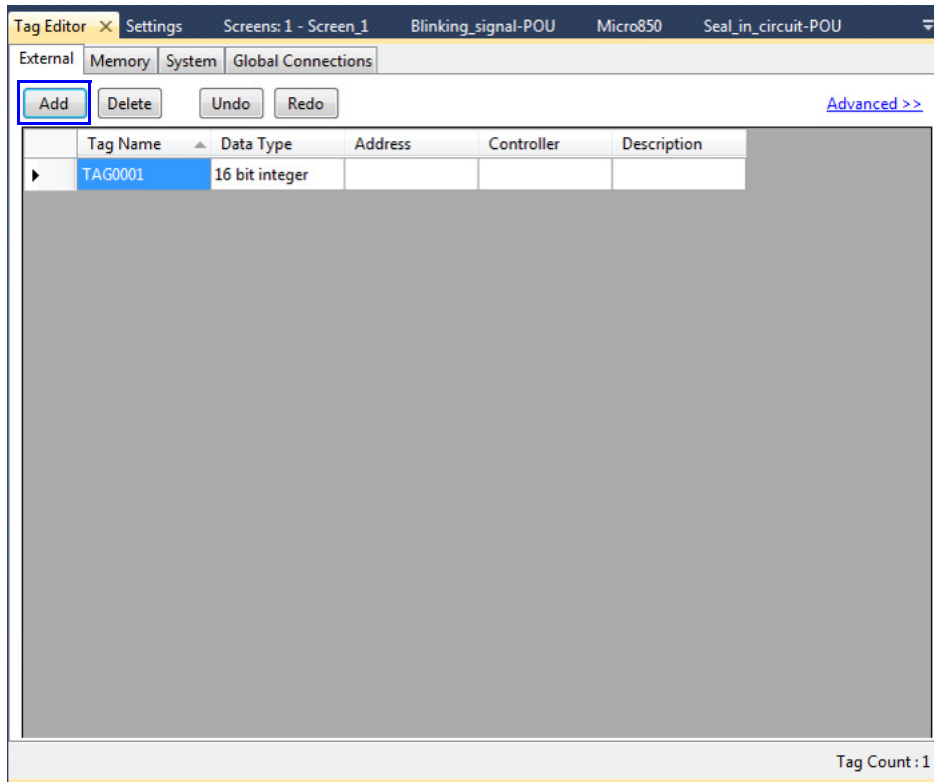
Save the project as “Self Latching Sample with PV800” using the menu option File -> Save Project As, then download the changes to the controller.

You have learned to configure the Ethernet port settings for your Micro800 controller.

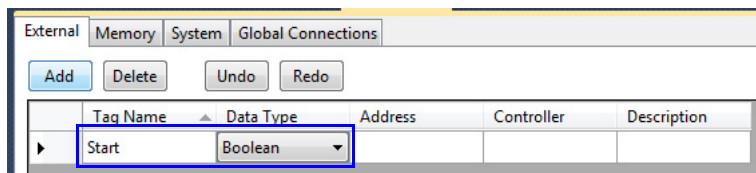
HMI Tag Editor

In this chapter you will learn how to add tags from the Micro800 controller to your PanelView 800 terminal.

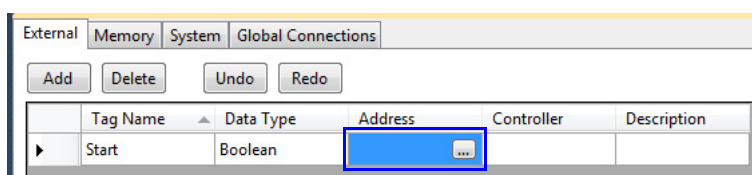
1. In the Tag Editor tab, Click Add to add a tag to your PanelView 800 application.



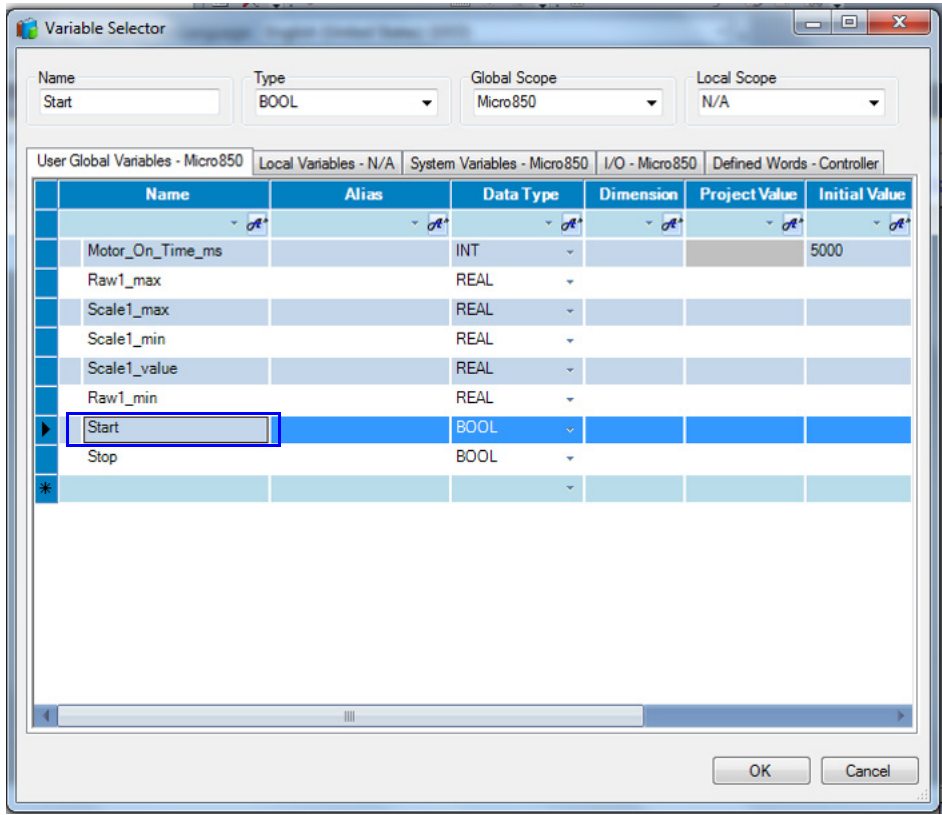
2. Rename the tag to “Start” and change the Data Type to Boolean.



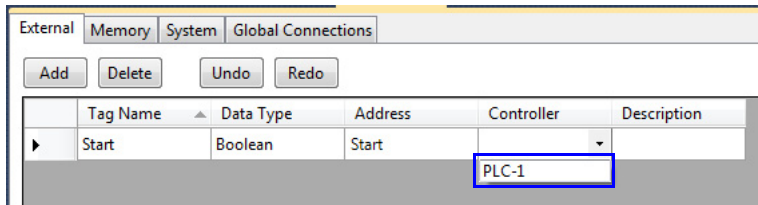
3. Select the Address field and then click the ellipses to browse the Micro800 controller program for the controller variable.



4. Select variable **Start** and click OK.



5. Select the Controller field and select controller PLC-1 from the drop-down list.



You have completed adding a tag.

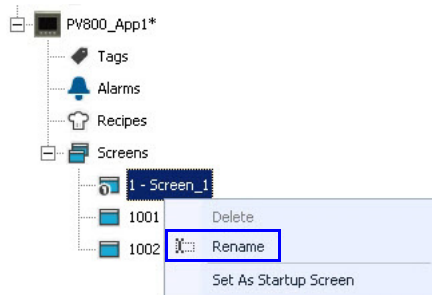
6. Repeat steps 1...5 to add the following tags.

Tag Name	Data Type	Address	Controller	Description
Start	Boolean	Start	PLC-1	
Stop	Boolean	Stop	PLC-1	
Raw1_min	Real	Raw1_min	PLC-1	
Raw1_max	Real	Raw1_max	PLC-1	
Scale1_min	Real	Scale1_min	PLC-1	
Scale1_max	Real	Scale1_max	PLC-1	
Scale1_value	Real	Scale1_value	PLC-1	
Motor_Run	Boolean	_IO_EM_DO_00	PLC-1	

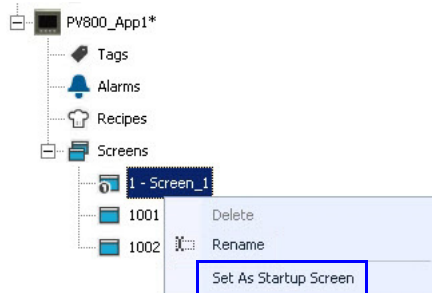
HMI Screen

Create a Screen for Your PanelView 800 Application

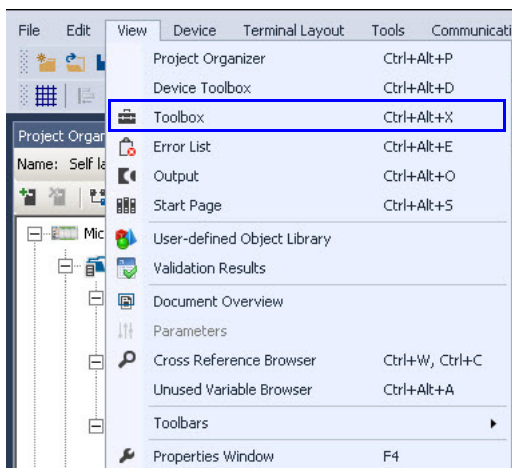
1. By default, a screen is already added to your application called Screen_1. Rename this screen to “Main” by right-clicking it, and select **Rename**.



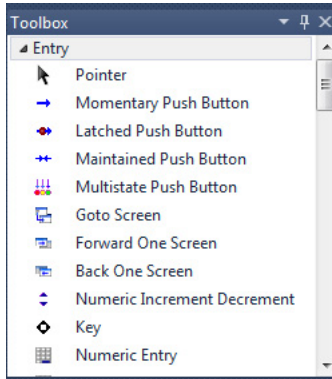
2. Right-click Main and select **Set As Startup Screen**.



3. Double-click Main to open the screen for editing.
4. Open the Toolbox window to access objects to use for designing your screen. Click View -> Toolbox from the menu bar.



The Toolbox window appears in the lower right hand corner of your workspace.

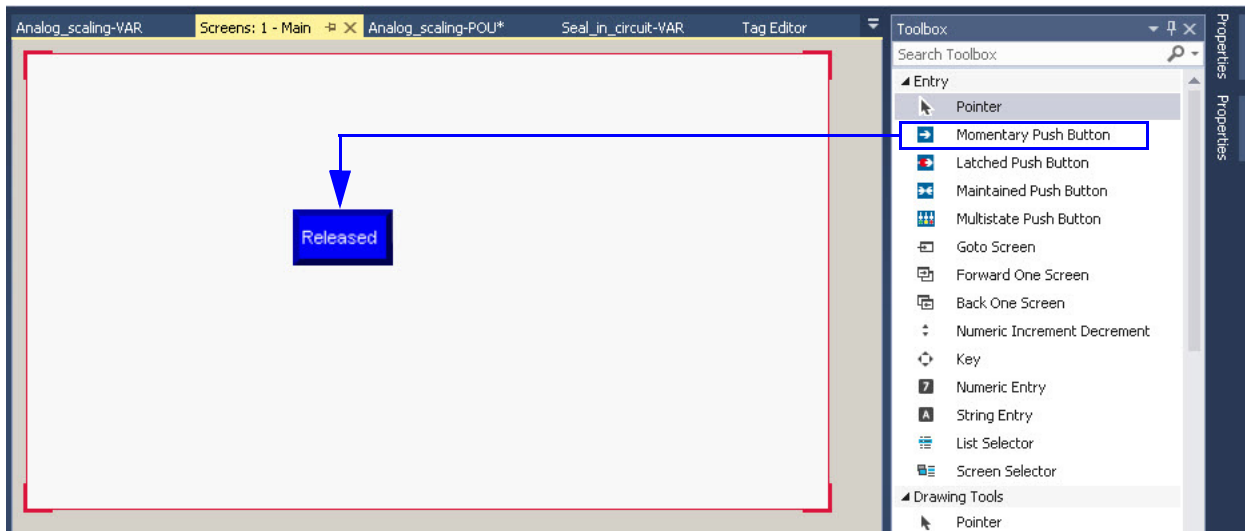


Create Objects for Your Screen

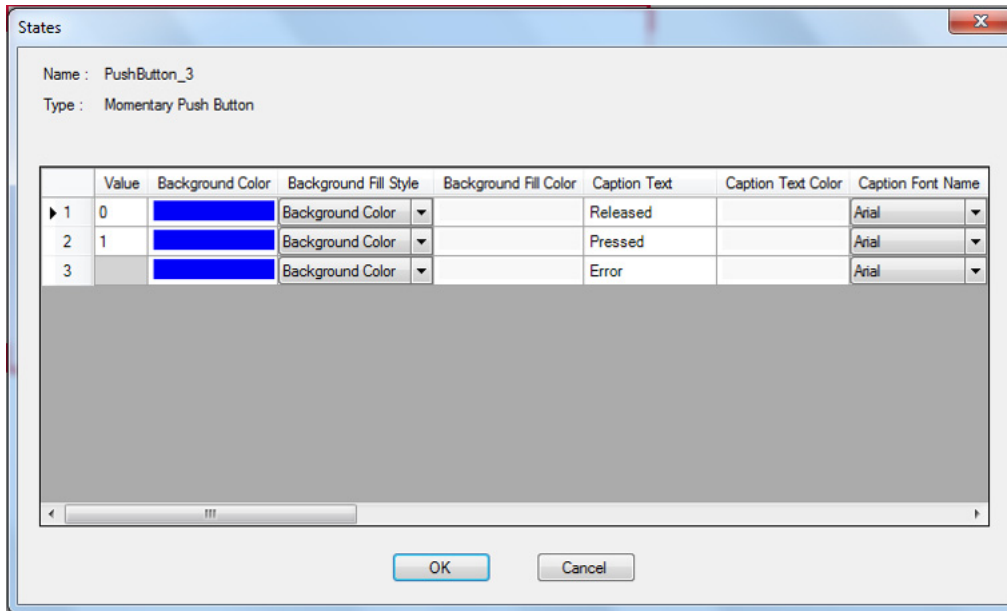
Now we will create objects for your PanelView 800 application screen.

Create Push Button Objects

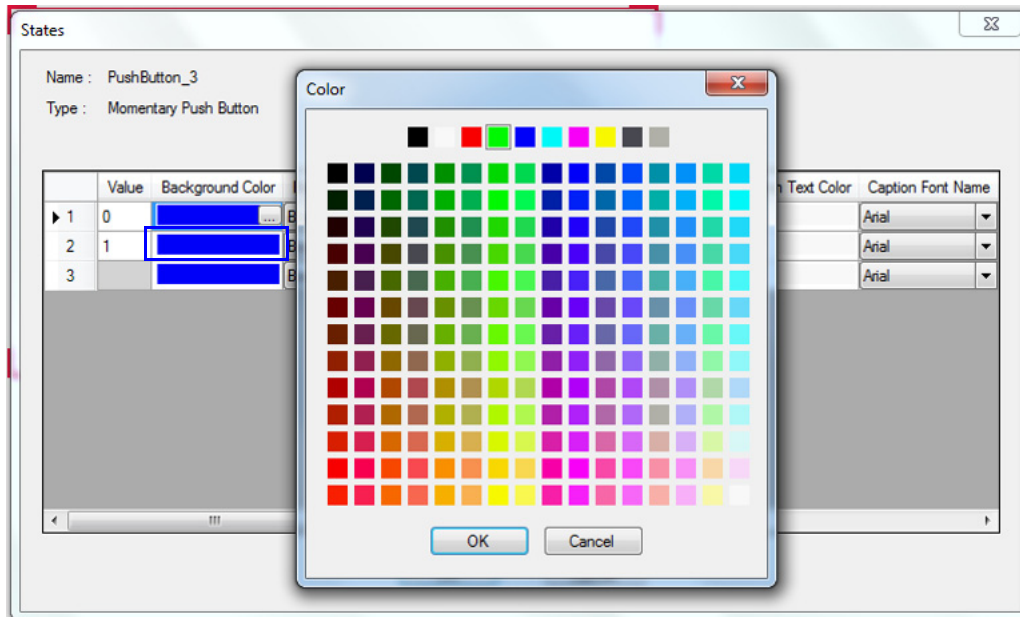
1. Locate the **Momentary Push Button** in your Toolbox and drag-and-drop it onto your screen.



- Once the push button appears on the screen, double-click it to change the appearance and properties of the push button states.






- Under the Background Color column, click the **ellipses** to bring up the Color Selector, and change the color of State 1 to green.



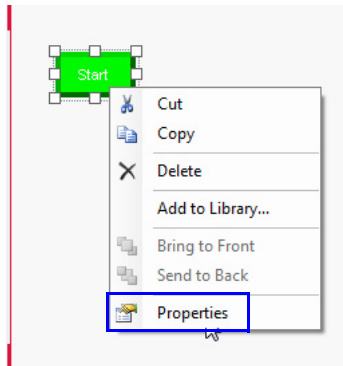
- Repeat step 3 to change the background color of State 2 to **green**.

5. Change the Caption Text of State 1 and State 2 to “Start”.

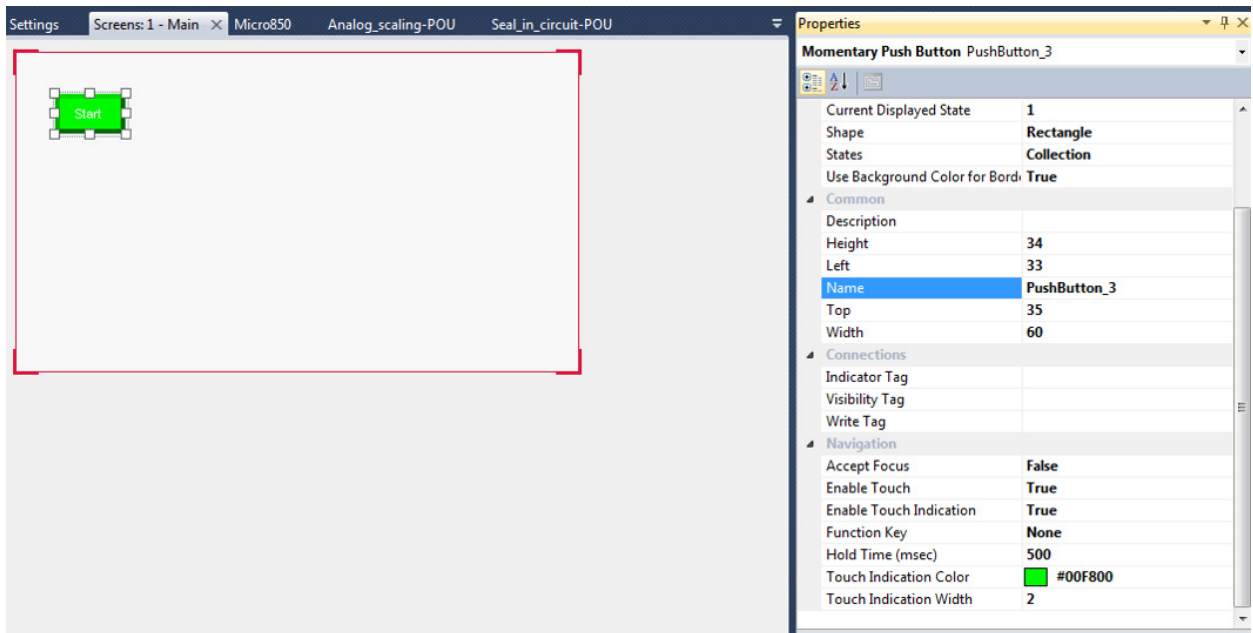
	Value	Background Color	Background Fill Style	Background Fill Color	Caption Text	Caption Text Color	Caption Font Name
▶ 1	0		Background Color ▼		Start		Arial ▼
2	1		Background Color ▼		Start		Arial ▼
3			Background Color ▼		Error		Arial ▼

6. Click OK to close the State Editor and apply the changes.

7. Right-click the momentary push button and select Properties to display the object’s properties window.



8. The object’s properties window opens on the right-hand side.

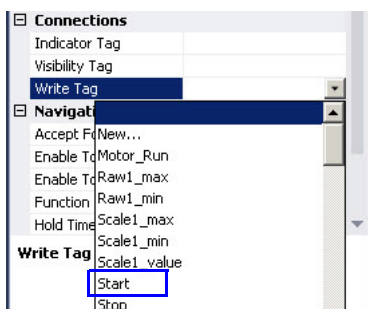


9. Set the following properties to the values specified below.

- Height: 50
- Width: 70
- Left: 15
- Top: 20

Common	
Description	
Height	50
Left	15
Name	PushButton_3
Top	20
Width	70

10. Click the Write Tag drop-down arrow in the properties window, and select **Start** from the list of tags.



You have completed creating your first push button. If you hover your mouse cursor over the momentary push button you can see its tag connections.



Create Maintained Push Button Object

1. Add a maintained push button next to the Start push button and configure the states of this object as shown.

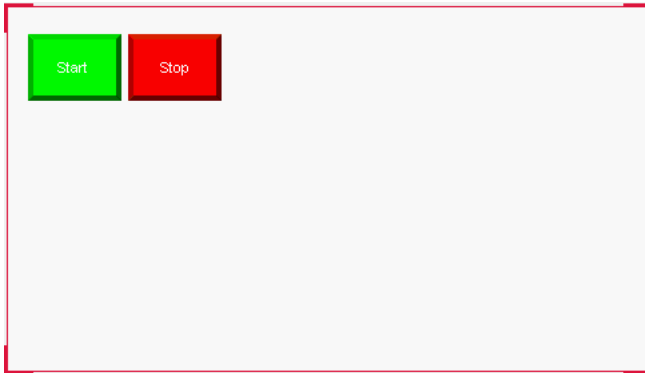
	Value	Background Color	Background Fill Style	Background Fill Color	Caption Text	Caption Text Color	Caption Font Name
1	0	 	Background Color		Stop		Arial
2	1	 	Background Color		Stop		Arial
3		 	Background Color		Error		Arial

2. Open the properties window for the object and set the following properties.

- Height: 50
- Width: 70
- Left: 90
- Top: 20

3. Set the following tag connections for the object.
 - Indicator Tag: None
 - Visibility Tag: None
 - Write Tag: Stop

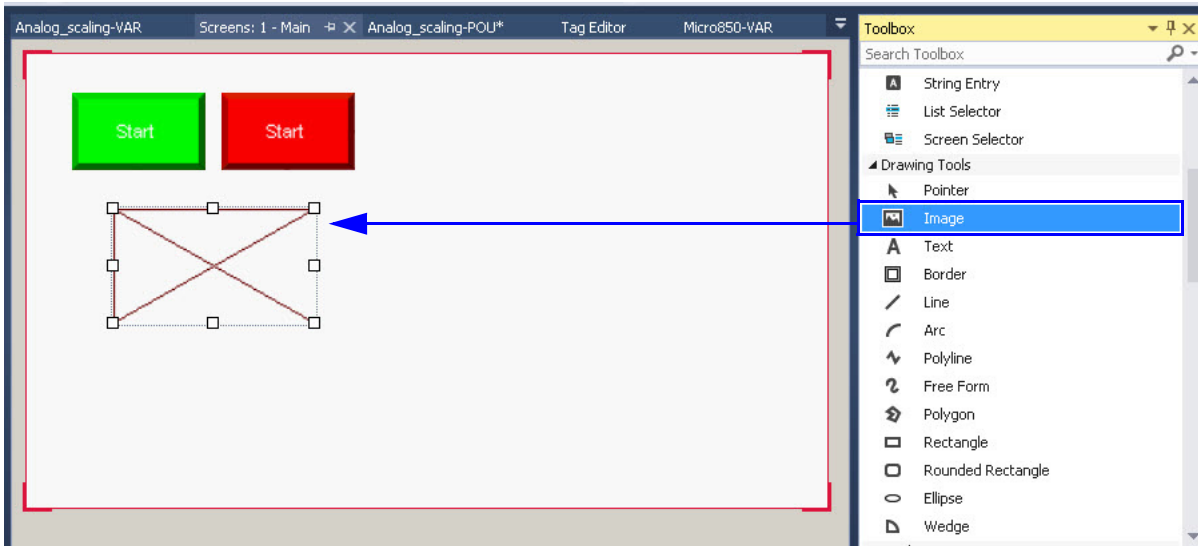
Your screen should look like this.



Next we will add an image to represent the motor operation.

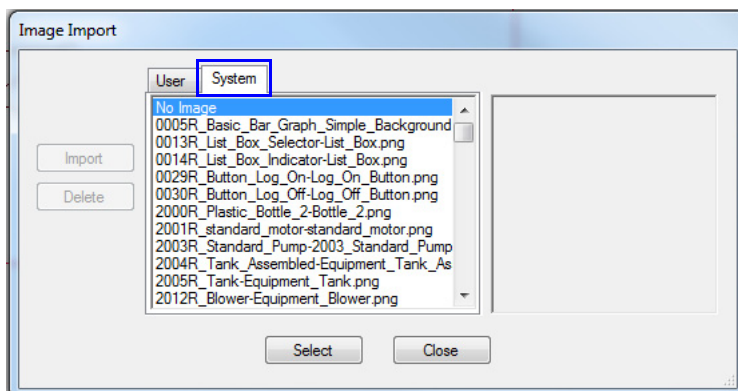
Add an Image to Your Screen

1. Locate the **Image** tool in the Toolbox and drag-and-drop it onto the screen in the lower left corner.

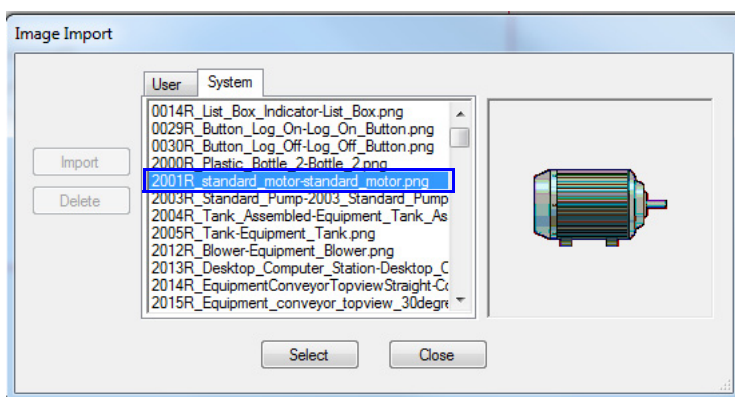


2. Double-click Image object you just added. This launches the Image Import dialog box.

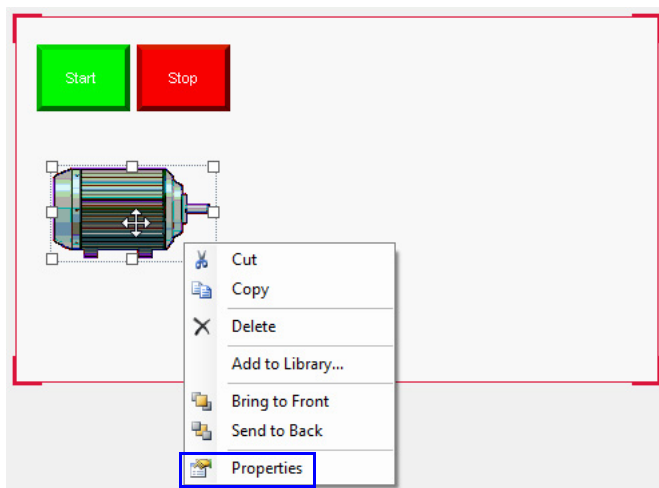
3. Select the System tab.



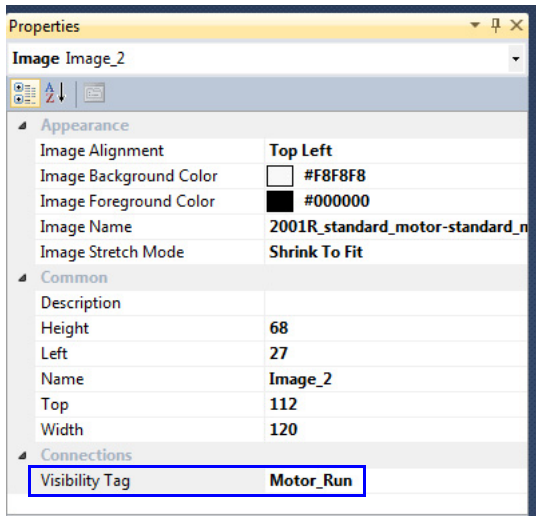
4. Select the image “2001R_standard_motor_standard_motor.png” and click Select.



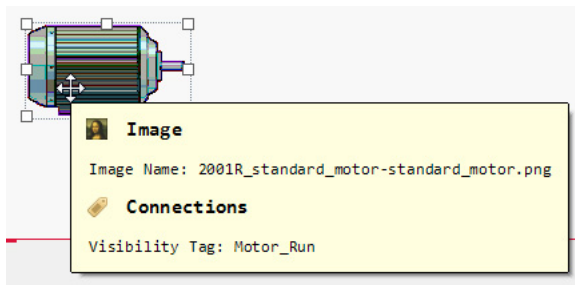
5. Right-click the image and select Properties to display its properties window.



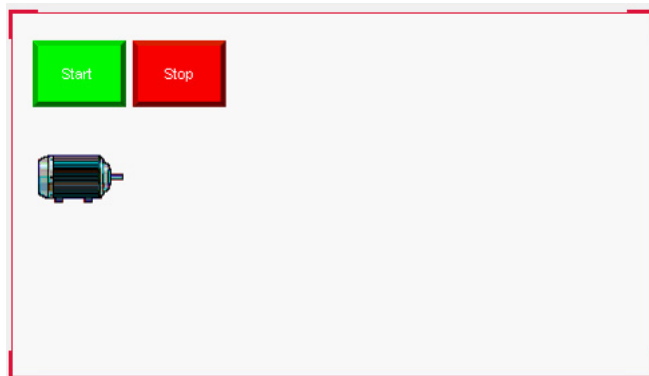
6. Fill in the tag information for the image object as shown below.



If you hover your mouse cursor over the image you can see its tag connections.



Your screen should look like this.



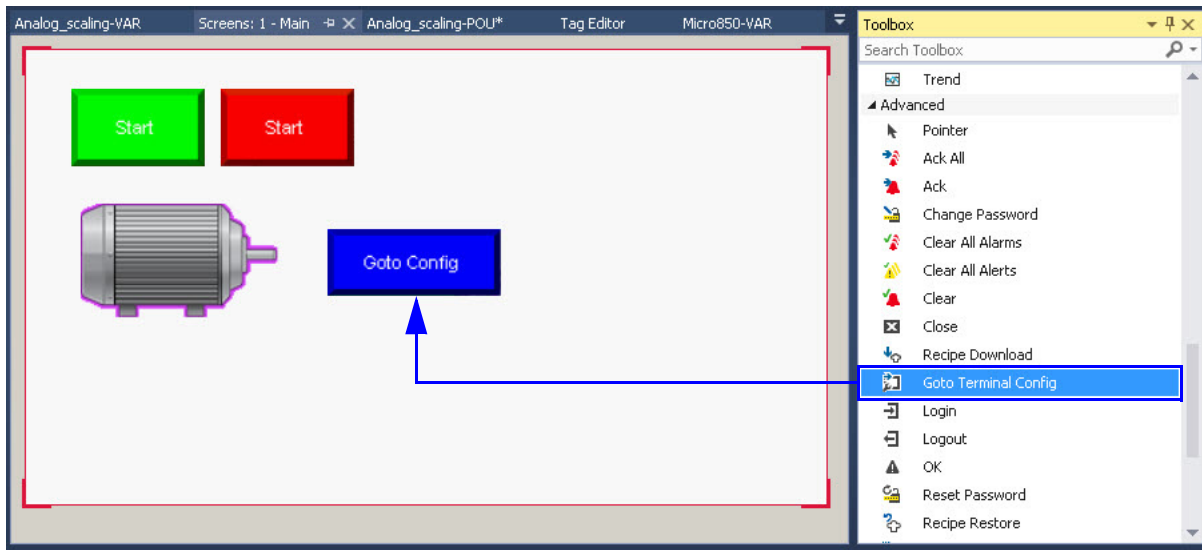
Next, we will create a Goto Config push button.

Create Goto Config Button Object

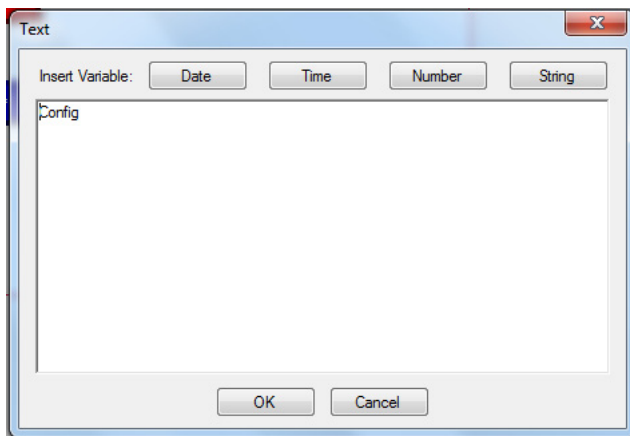
Returning to the configuration screen of the PanelView 800 terminal is very important for all applications. The configuration screen allows users to change terminal settings, and observe its communication, set up and memory usage.

To create a Goto Config push button, do the following:

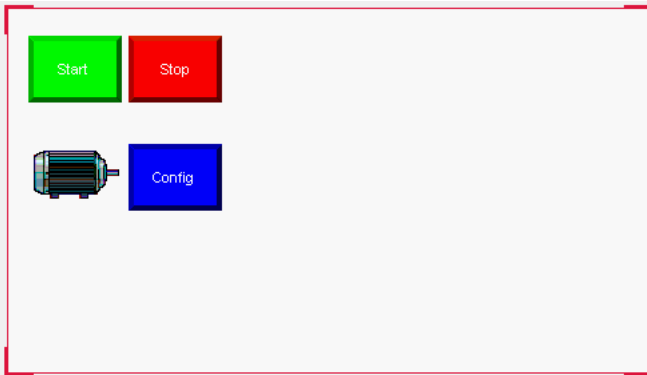
1. Locate the **Goto Config** push button in your Toolbox. Drag-and-drop it onto your screen, just below your Stop push button.



2. Double-click the Goto Config push button and change the text to “Config”, then click OK.



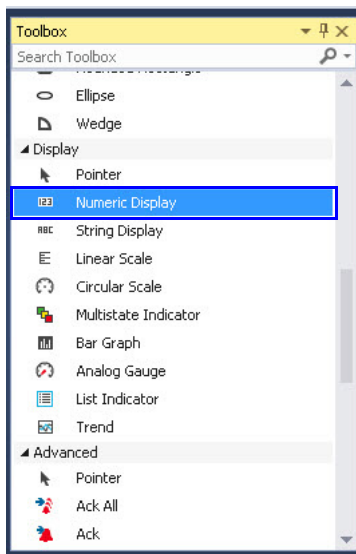
3. Change the button's width to "70" and height to "50". Your screen should look like this.



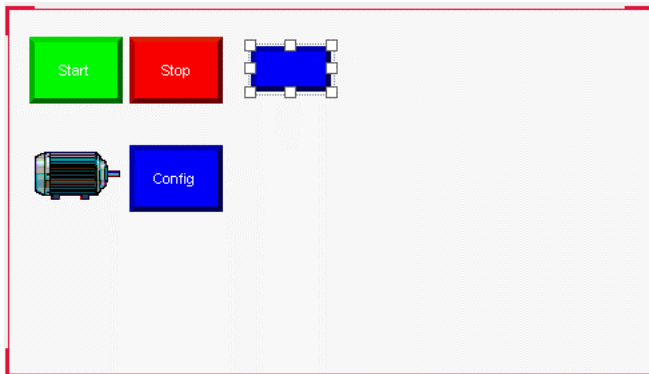
Create a Numeric Display Object

A numeric display shows the value of the tag that it is connected to. For this application, we displays the 2080-IF2 analog module channel 0 scaled value.

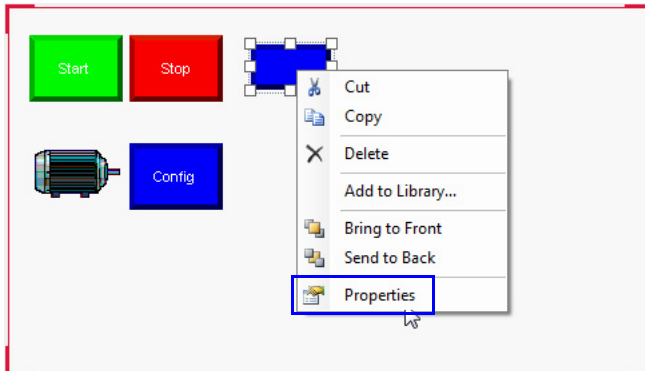
1. Locate the **Numeric Display** object in your Toolbox.



2. Drag-and-drop the Numeric Display object onto your screen, next to the Stop push button.



3. Right-click the object and select Properties to display its properties window.



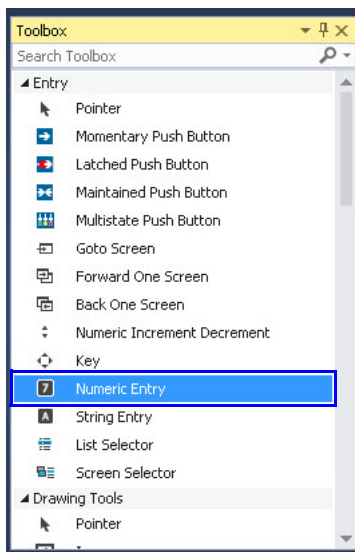
4. Set the following property values.
 - Width: 81
 - Height: 33
5. Set the Read Tag to **Scale1_value**.
6. Set the Number of Decimal places to “0” and Number of Digits to “6”.

Next, we will create a Numeric Entry object.

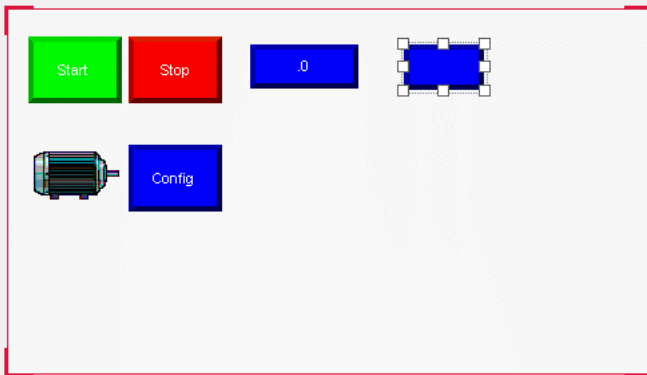
Create a Numeric Entry Object

You are to set the minimum and maximum range for the Raw and Scaling value for the analog input scaling.

1. Locate the **Numeric Entry** object in your Toolbox.

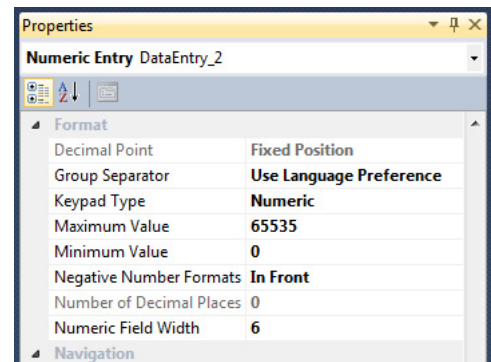


2. Drag-and-drop the Numeric Entry object onto your screen, next to the Numeric Display object.



3. Right-click the object and select Properties to display its properties window.
4. Set the following property values:
 - Width: 81
 - Height: 33

5. Set the following values:
 - Keypad Type: Numeric
 - Maximum Value: 65535
 - Minimum Value: 0
 - Number of Decimal Places: 0
 - Numeric Field Width: 6



6. Set the Write Tag to **Raw1_min**.
7. Set the Indicator Tag to **Raw1_min**.

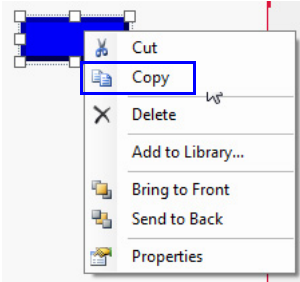
You have created a Numeric Entry object for setting the Raw minimum value

8. Hover your mouse cursor over the object to see the tag values.



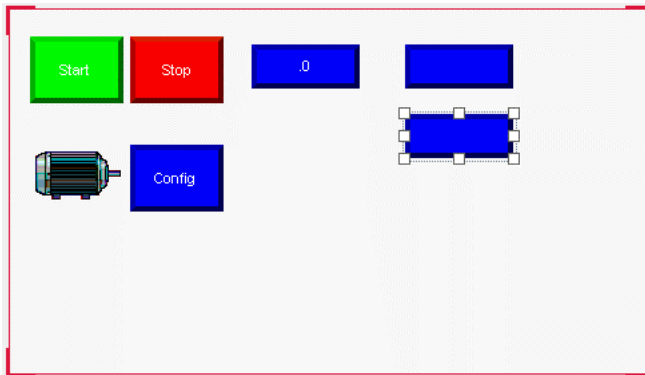
Let us duplicate this numeric entry object for the **Raw1_max** setting.

1. Right-click the object and select **Copy**.



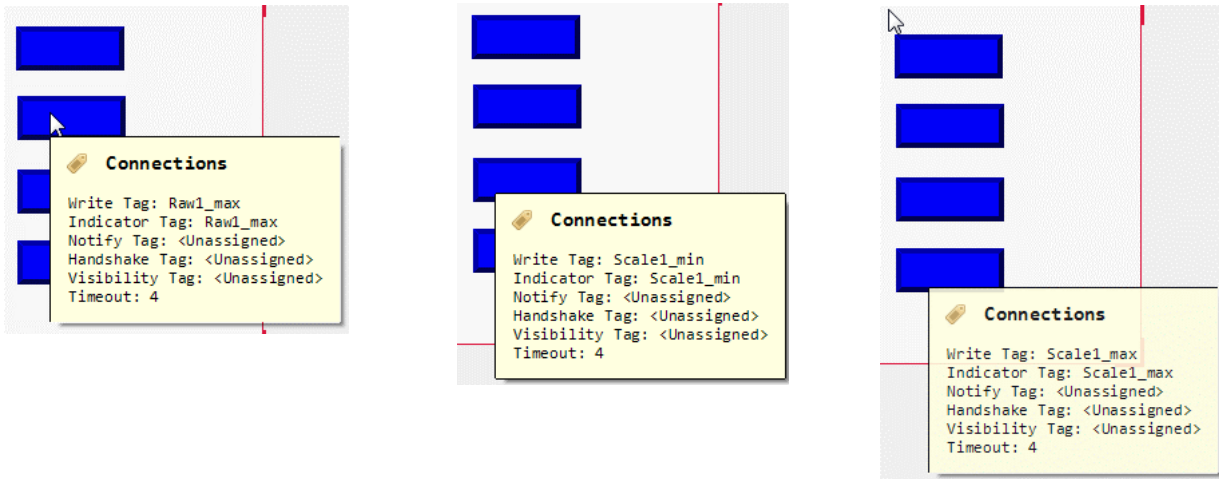
2. Right-click on the screen below the existing numeric entry object and select **Paste**.

Your screen should look like this.

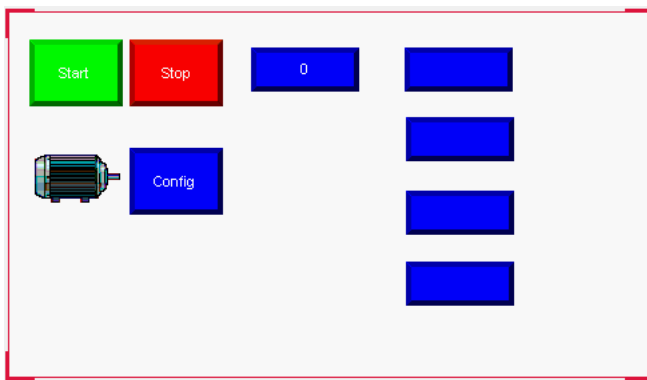


3. Set the Write Tag to **Raw1_max** and Indicator Tag to **Raw1_max**.
4. Duplicate two more numeric entry objects for the **Scale1_min** and **Scale1_max** display and entry.
5. Set the following values for the **Scale1_min** and **Scale1_max** numeric entry object:
 - Maximum Value: 100
 - Minimum Value: 0
 - Number of Decimal Places: 0
 - Numeric Field Width: 6

6. Hover your mouse cursor over the numeric entry objects to see the Write and Indicator tags.



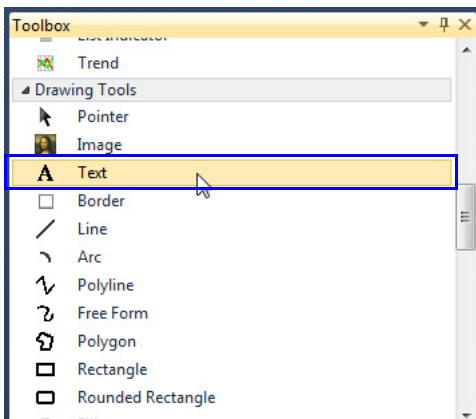
Your screen should look like this.



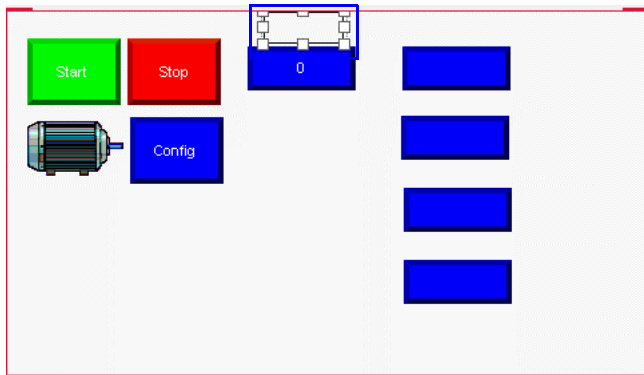
Next, we will add text to the numeric display and entry objects.

Create Text Labels for Your Objects

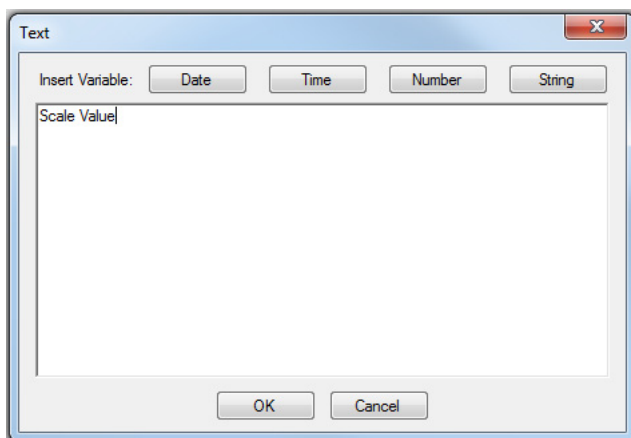
1. Locate the **Text** object in your Toolbox.



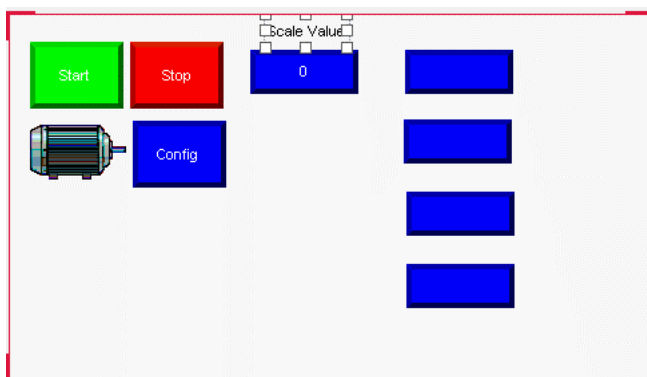
2. Drag-and-drop the text object onto your screen, above the numeric display object.



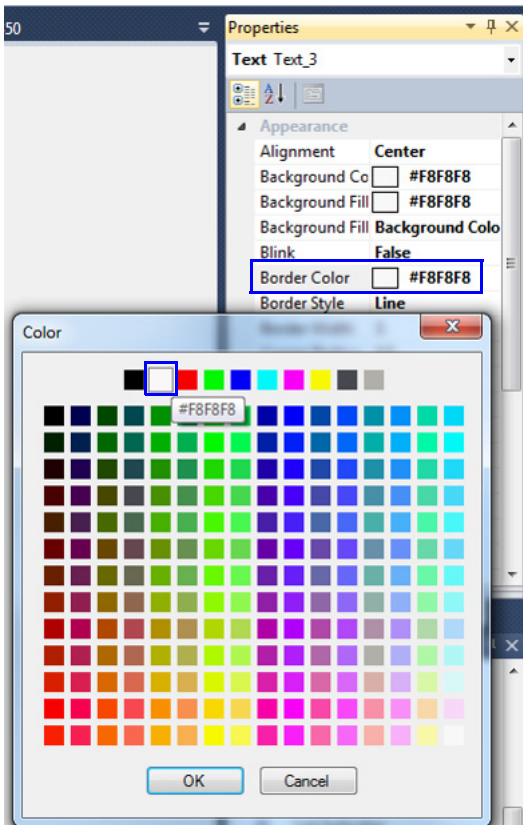
3. Double-click text object and type in "Scale Value", then click OK.



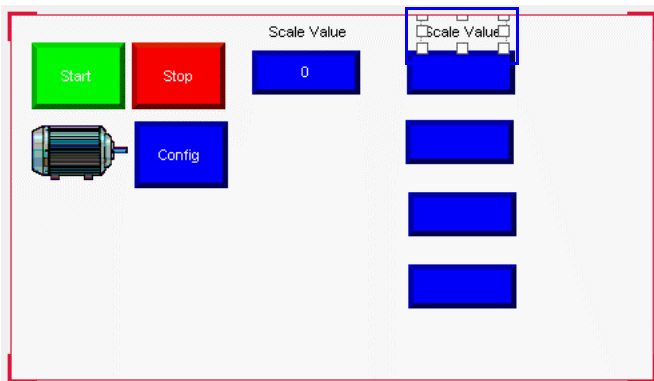
4. Access the properties of the text object



- 5. Set the Border color to White and click OK.

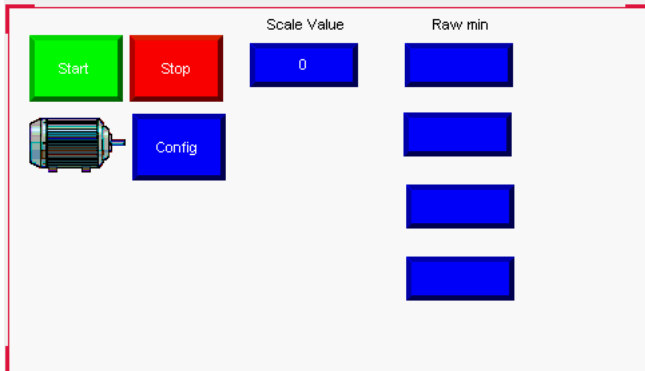


- 6. Copy the text object and paste it above the numeric entry object.

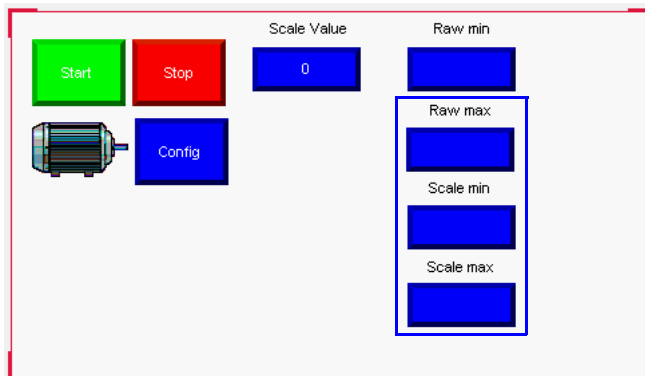


- 7. Double-click the text object and type in "Raw min", then click OK.

Your screen should look like this.



8. Duplicate the text object for the remaining three numeric entry objects.
9. Replace the text as shown.



10. Adjust the spacing for the numeric entry objects and text objects.

You have completed the screen required for the application. Next, let us configure the IP address of the PanelView 800 terminal.

Notes:

Configure PanelView 800 Terminal Ethernet Settings

1. From the PanelView 800 terminal, select **Terminal Setting** -> **Communication**.
2. If the IP Mode is “DHCP”, select **Disable DHCP**.
3. Select **Set Static IP Address**.
4. Set IP Address as **192.168.1.2**.
5. Set Mask as **255.255.255.0**.
6. Leave the Gateway as default **0.0.0.0**.
7. Select **Back** to go back to the Communication screen.
8. Select **Main** to go back to the main configuration screen.

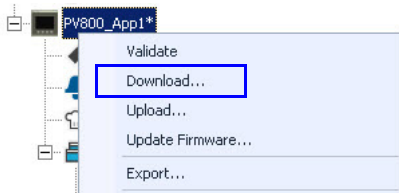
Notes:

Download HMI Application

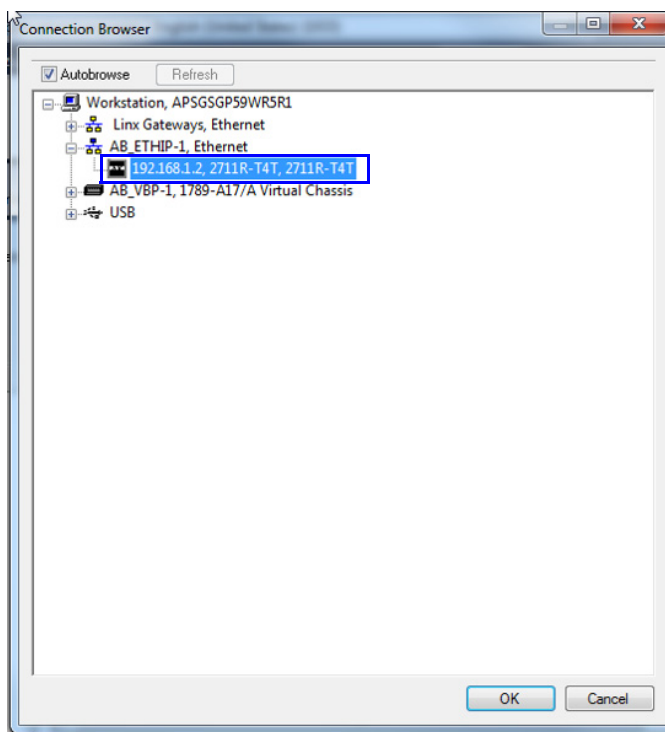
Save and Download Project

Save the project and download your application to the PanelView 800 terminal.

1. Right-click PV800_App1 and select **Download**.

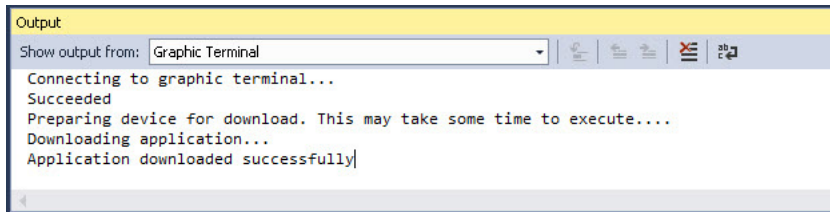


2. Browse and select **2711R-T4T** as the target to download and click OK.



By default the EtherNet/IP drive will scan 256 addresses. In order to decrease the time to detect the terminal, it may be necessary to right-click on the drive and use Properties to limit the range of address from zero to four.

3. The Output window shows the download status to the PanelView 800 terminal.



Test the HMI Application

Run the HMI Application

1. From the PanelView 800 terminal main screen, press **File Manager** to select the downloaded application.
2. Select **PV800_App1** and press **Set As Startup**.
3. Press **Run**.

Connect an Ethernet cable between the PanelView 800 terminal and the Micro800 controller.

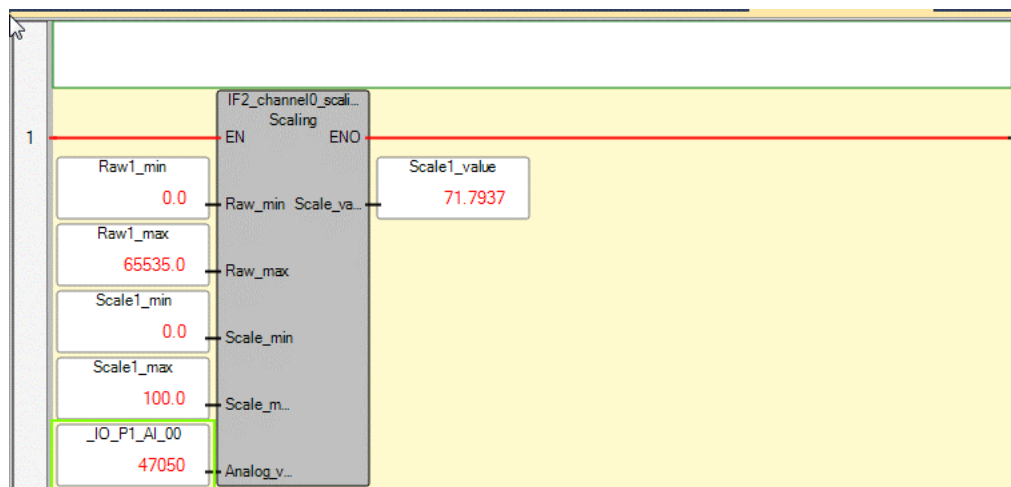
Test the HMI application.

Key in the following values for the analog scaling:

- Key in **0** for the Raw_min
- Key in **65535** for the Raw_max
- Key in **0** for the Scale_min
- Key in **100** for the Scale_max

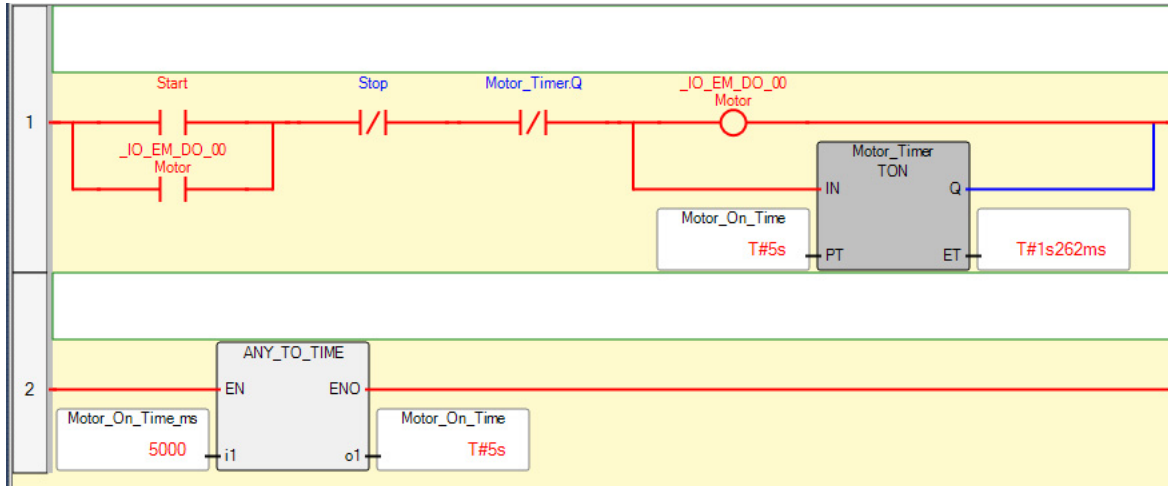
You should see the Scale value display being updated on the HMI screen as you turn the potentiometer on the simulator board.

Connect to the controller and you should see the following values from your Analog_scaling program.



Testing the Seal_in Circuit program:

1. Press the **Start** pushbutton to turn on output 0 and you should see the motor image appearing on the screen for five seconds.
2. Press the **Stop** pushbutton to turn off output 0.



You have now learned how to setup and program a PanelView 800 terminal using the Connected Components Workbench software.

Set Up Serial Communication Between PC and Micro820 Controller

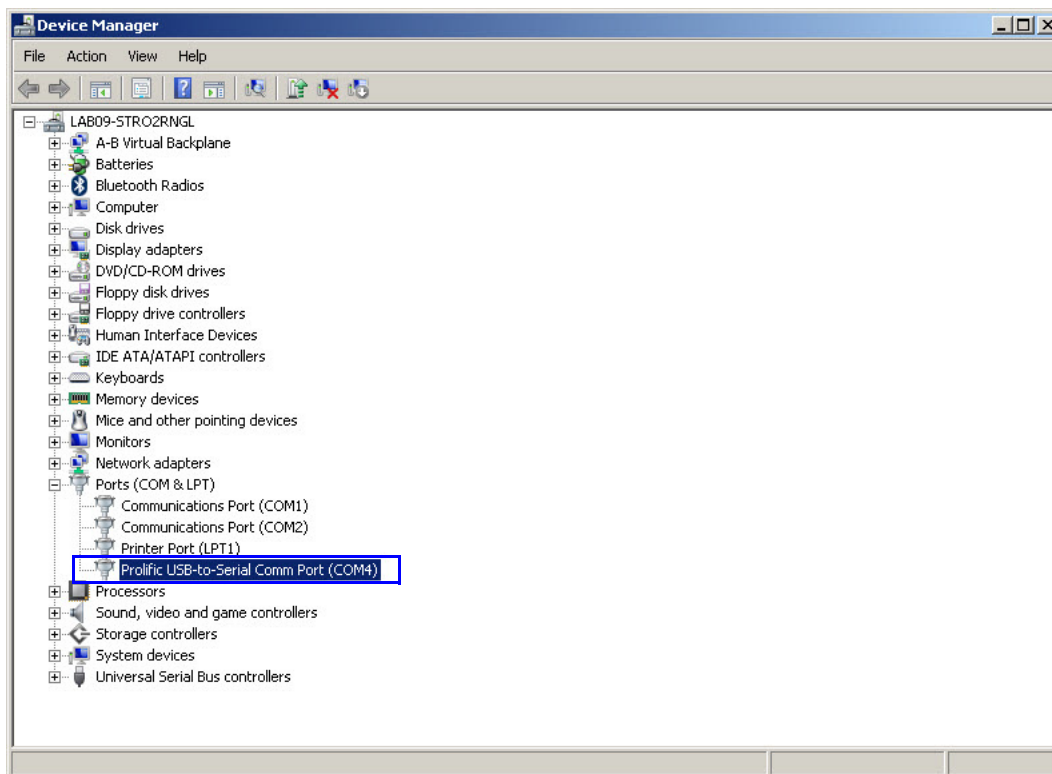
The embedded serial port on the Micro820 controller can be used as a programming port. This section describes how to set up RSLinx to allow Connected Components Workbench software to communicate with the Micro820 controller.

Things you need:

- A PC with a serial port, a 9300-USBS USB-Serial adapter, or a 2080-REMLCD.
- A 1761-CBL-PM02 cable that has been modified to connect between PC and Micro820 controller.
- A Micro820 controller.

Before You Begin

1. If you are using a USB-Serial adapter, verify that the driver is properly installed. You can check the communication port number through the Windows Device Manager if the adapter works properly in your computer.



If your PC comes with an RS232 serial port, simply note down the communication port number of the serial port you intend to use. In this example shown above, the communication port used is **COM4**.

Connect the PC to the Micro820 Controller

To connect the PC to the Micro820 controller using the 2080-REMLCD, see the Micro820 Programmable Controllers User Manual, publication [2080-UM005](#). A USB type A/B cable is required.

To connect the PC to the Micro820 controller using the serial port on the PC, or the USB-Serial adapter, you need to modify a cable so that it has a 9-pin D sub connector on one end and exposed wires on the other end. Here we will use the Allen Bradley MicroLogix programming cable (1761-CBL-PM02).

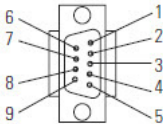
Modifying the 1761-CBL-PM02 Cable



The 1761-CBL-PM02 cable has an 8-pin mini DIN connector on one end and a 9-pin D sub connector on the other end. Cut the cable at any point near the end with the 8-pin mini Din connector to expose the wires.

Alternatively, you can customize your own cable with a DB9 female connector on one end. Note that the cable length should not exceed three meters (10 feet).

1761-CBL-PM02 Pinout Diagram



Pin	DB-9 RS-232	Micro820
2	Received data (RxD)	Tx terminal
3	Transmitted data (TxD)	Rx terminal
5	Signal common (GRD)	G terminal

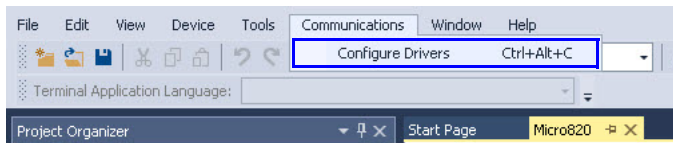
The rest of the pins and terminal connections are not required.

1. Connect the exposed wires of the cable to the Micro820 controller.
2. Connect the other end of the cable to your computer serial port or a USB-Serial adapter.

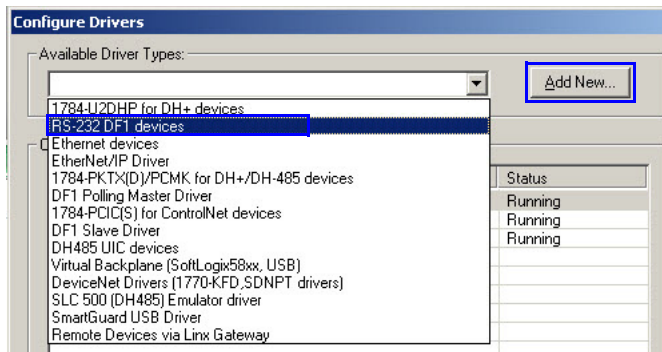
Configure RSLinx

When using the 2080-REMLCD, the DF1 driver is automatically added when the USB cable is connected to the PC and no RSLinx configuration is needed. When using the embedded serial port on the PC, or USB-Serial adapter, you must manually add the DF1 driver.

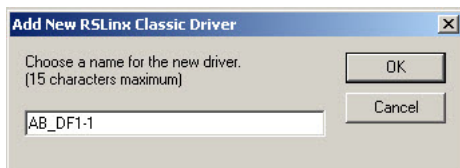
1. In Connected Components Workbench software, click Communications -> Configure Drivers.



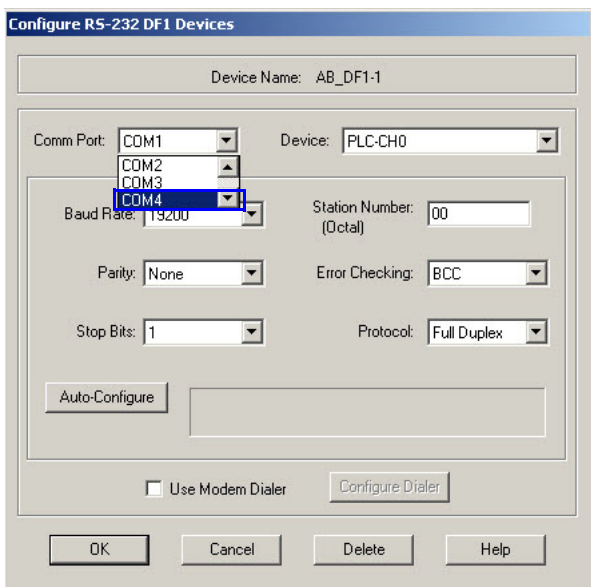
2. Select **RS-232 DF1 devices** from the drop-down menu and click Add New.



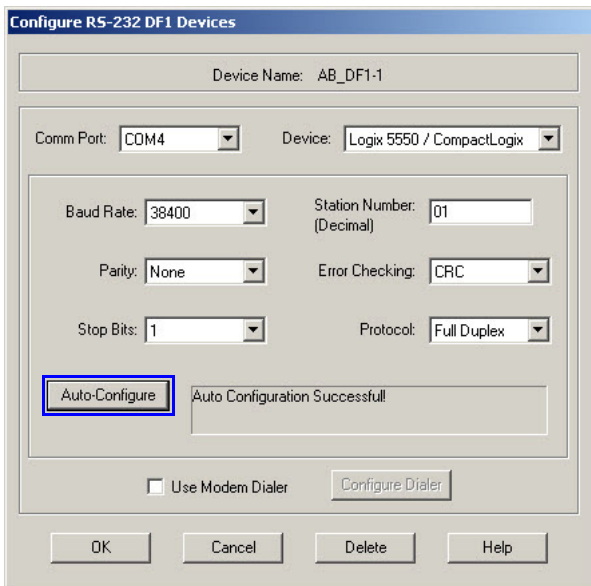
3. Name the new driver and click OK.



- 4. Select the communication port number where the Micro820 controller is connected to. For this example, select COM4.



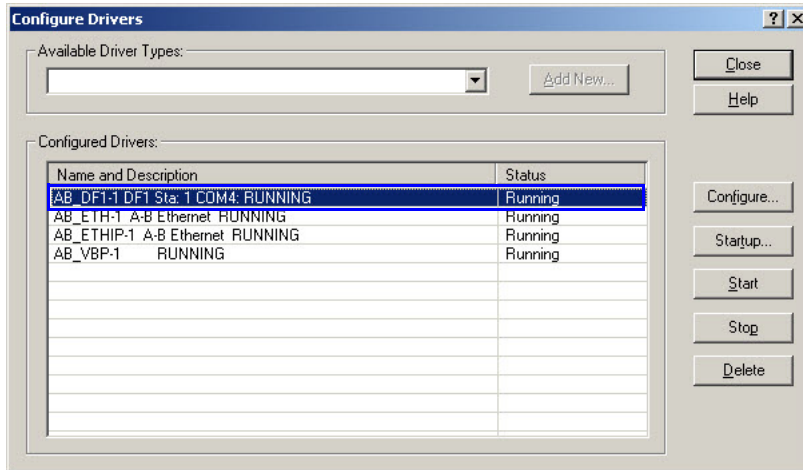
- 5. Verify that your PC is connected to the Micro820 controller, then click Auto-Configure.



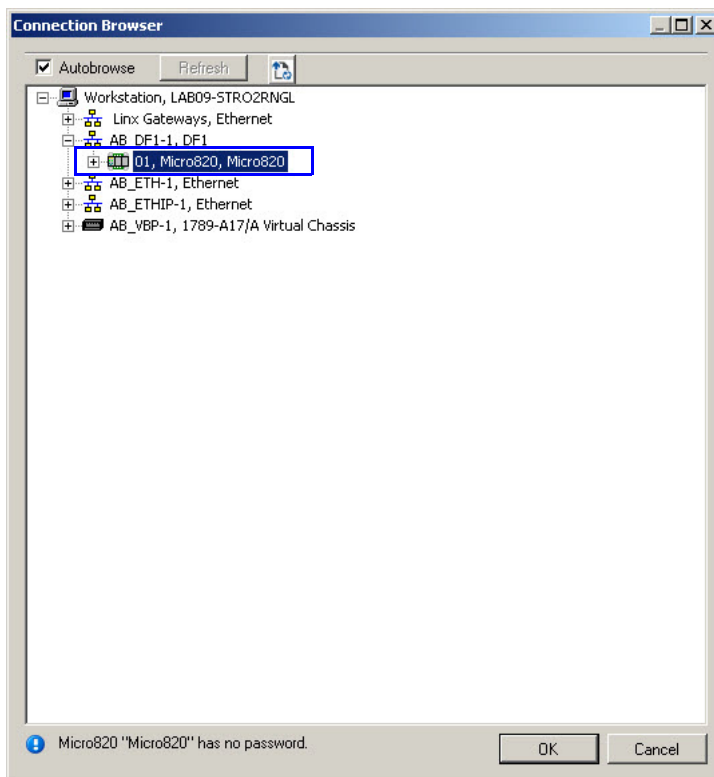
Alternatively, you can manually enter the configuration based on the image shown above.

- 6. Click OK after you have finished configuring the driver for the Micro820 controller.

7. Verify that the newly configured driver is “RUNNING”, then close the window.



8. Connected Components Workbench software should now be able to establish a connection with the Micro820 controller if the serial port configuration has not changed.

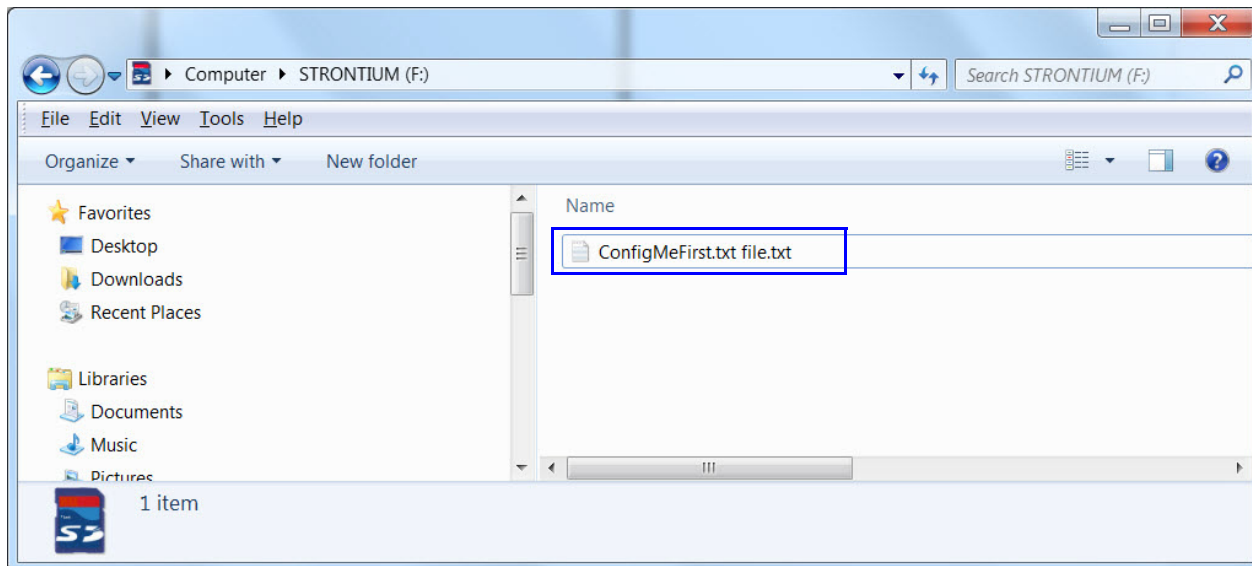


If Connected Components Workbench software is not able to establish communication with the Micro820 controller. See the next section to learn how to restore the embedded serial port back to the factory default setting.

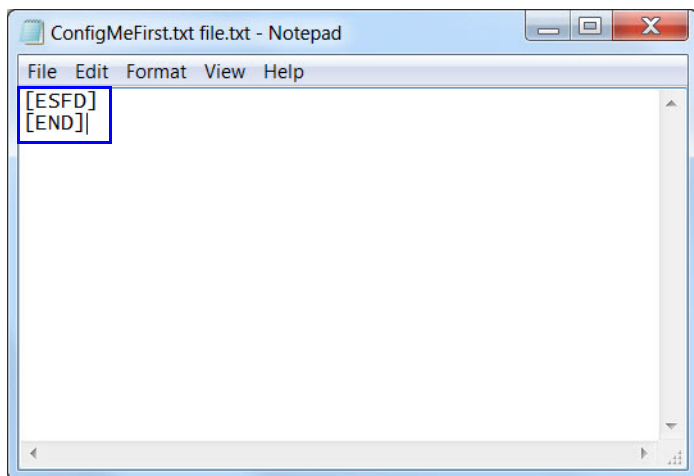
Restore Serial Port to Factory Default Setting Using MicroSD Card

This section describes how to configure the embedded serial port to work with Connected Components Workbench software. A microSD card is required to perform the following steps.

1. Using your PC, create a “ConfigMeFirst.txt” file in the microSD card.



2. Open the file and type in the commands as shown below.




These commands will restore the embedded serial port back to the factory default setting upon power cycling the Micro820 controller.

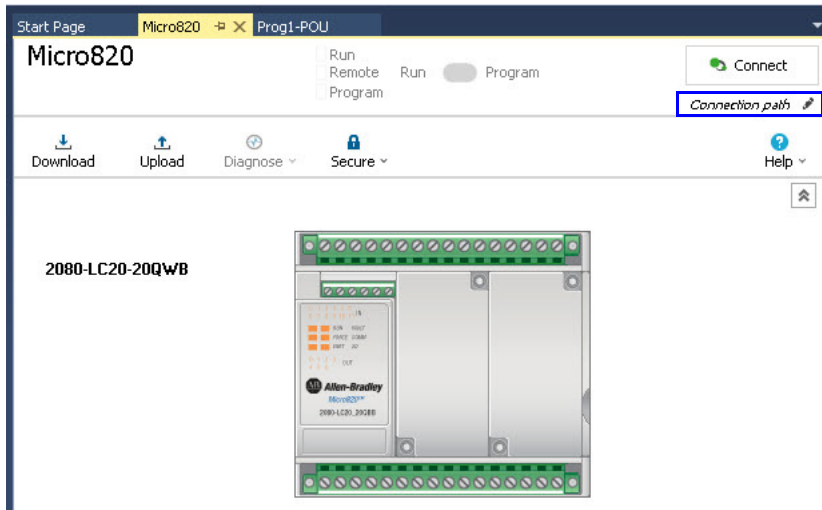
3. Save and close the text file.
4. Remove the microSD card from the PC and insert it into the microSD card slot located at the side of the Micro820 controller.

When the microSD card is detected, the SD card LED on the controller lights up.

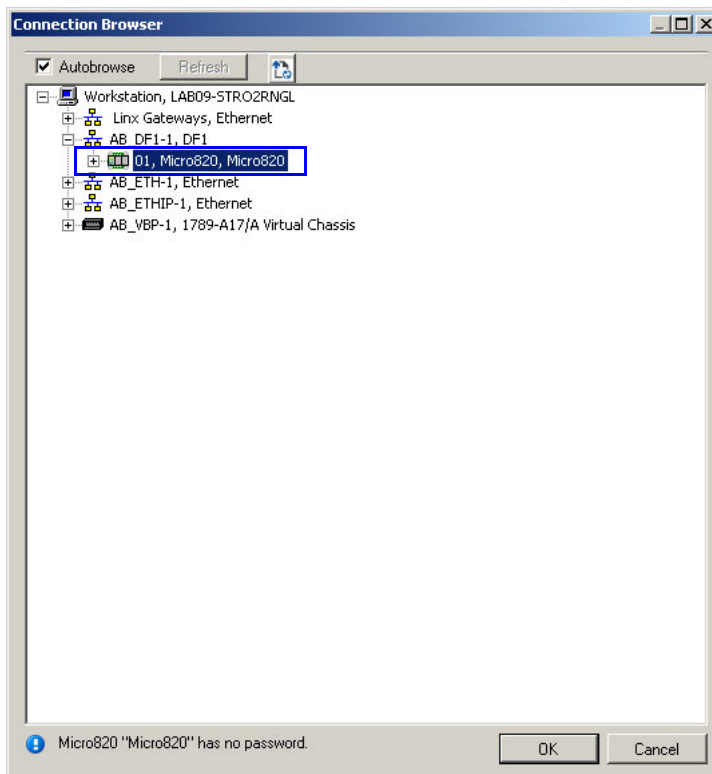
5. Power cycle the controller.

The embedded serial port will be restored to the factory default setting. When the SD card LED becomes stable, you may remove the microSD card.

6. In Connected Components Workbench software, click the Connection Path  icon to set up the communication path.

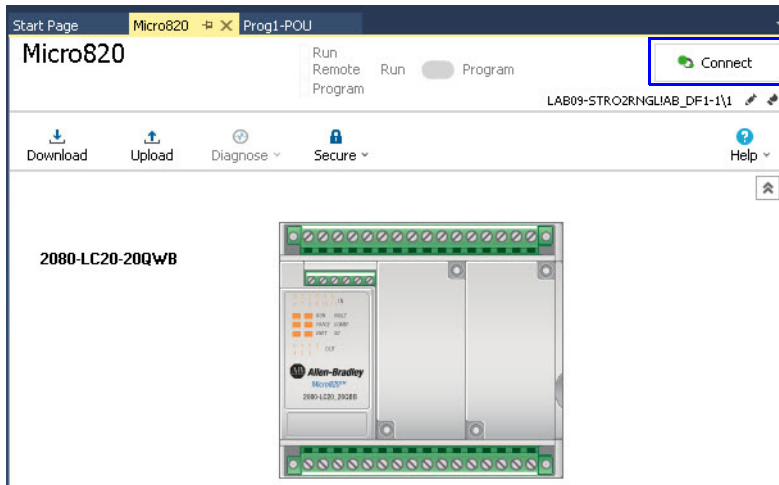


7. Browse for the Micro820 controller under the newly added DF1 driver and click OK.
The 2080-REMLCD driver is identified as "2080-REMLCD, DF1" instead of "AB_DF1, DF1" in RSLinx.



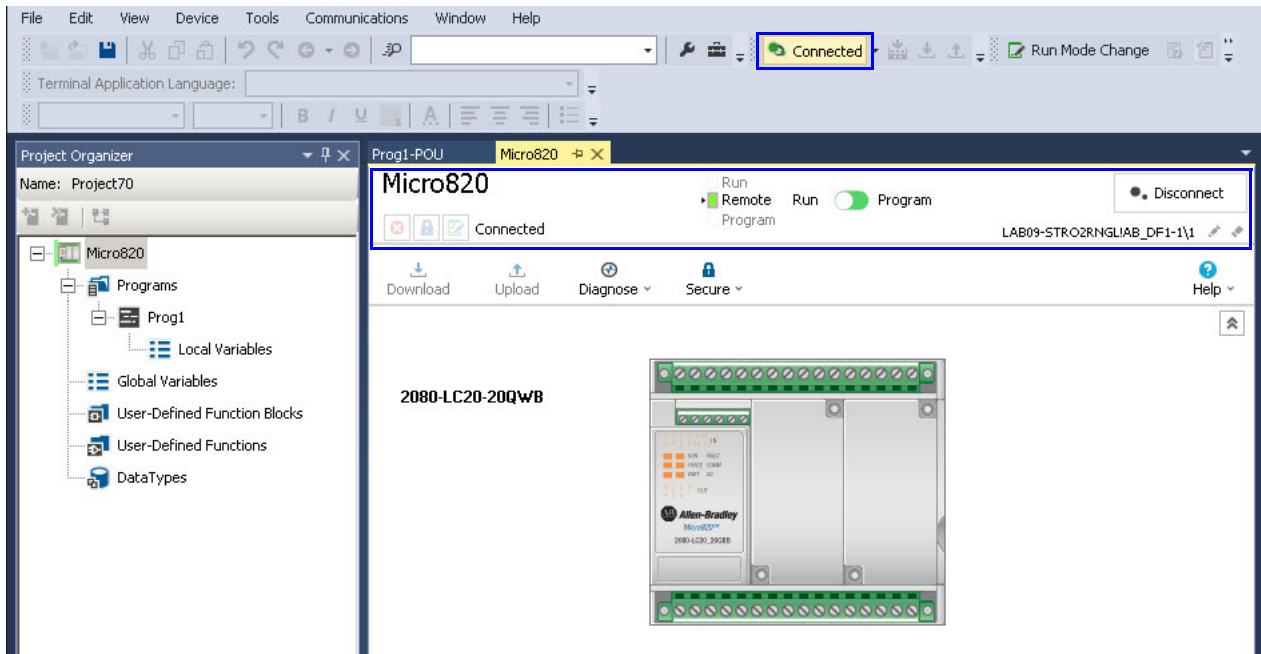
If Connected Components Workbench prompts an error when you select the Micro820 controller, or the controller does not seem available, try to restart the Connected Components Workbench software and set up the communication path again.

- 8. Click **Connect** to establish communication between Connected Components Workbench software and the Micro820 controller.



If Connected Components Workbench software is unable to establish connection with the Micro820 controller, try to restart the Connected Components Workbench software and connect again.

- 9. The following image shows a successful connection between Connected Components Workbench software and the Micro820 controller.



Rockwell Automation Support

Use the following resources to access support information.

Technical Support Center	Knowledgebase Articles, How-to Videos, FAQs, Chat, User Forums, and Product Notification Updates.	https://rockwellautomation.custhelp.com/
Local Technical Support Phone Numbers	Locate the phone number for your country.	http://www.rockwellautomation.com/global/support/get-support-now.page
Direct Dial Codes	Find the Direct Dial Code for your product. Use the code to route your call directly to a technical support engineer.	http://www.rockwellautomation.com/global/support/direct-dial.page
Literature Library	Installation Instructions, Manuals, Brochures, and Technical Data.	http://www.rockwellautomation.com/global/literature-library/overview.page
Product Compatibility and Download Center (PCDC)	Get help determining how products interact, check features and capabilities, and find associated firmware.	http://www.rockwellautomation.com/global/support/pcdc.page

Documentation Feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete the How Are We Doing? form at http://literature.rockwellautomation.com/idc/groups/literature/documents/du/ra-du002_-en-e.pdf.

Rockwell Automation maintains current product environmental information on its website at <http://www.rockwellautomation.com/rockwellautomation/about-us/sustainability-ethics/product-environmental-compliance.page>.

Allen-Bradley, Rockwell Software, Connected Components Workbench, ControlFLASH, Micro800, Micro820, Micro830, Micro850, PanelView, and Rockwell Automation are trademarks of Rockwell Automation, Inc. Trademarks not belonging to Rockwell Automation are property of their respective companies.

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444

Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640

Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Publication 2080-QS004B-EN-E - October 2017

Supersedes Publication 2080-QS004A-EN-E - December 2015

Copyright © 2017 Rockwell Automation, Inc. All rights reserved.