L12 - Basic PLC Programming with Micro800® Controllers and PanelView™ 800 Graphic Terminals

For Classroom Use Only!
Important User Information

This documentation, whether, illustrative, printed, “online” or electronic (hereinafter “Documentation”) is intended for use only as a learning aid when using Rockwell Automation approved demonstration hardware, software and firmware. The Documentation should only be used as a learning tool by qualified professionals.

The variety of uses for the hardware, software and firmware (hereinafter “Products”) described in this Documentation, mandates that those responsible for the application and use of those Products must satisfy themselves that all necessary steps have been taken to ensure that each application and actual use meets all performance and safety requirements, including any applicable laws, regulations, codes and standards in addition to any applicable technical documents.

In no event will Rockwell Automation, Inc., or any of its affiliate or subsidiary companies (hereinafter “Rockwell Automation”) be responsible or liable for any indirect or consequential damages resulting from the use or application of the Products described in this Documentation. Rockwell Automation does not assume responsibility or liability for damages of any kind based on the alleged use of, or reliance on, this Documentation.

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

Except as specifically agreed in writing as part of a maintenance or support contract, equipment users are responsible for:
- properly using, calibrating, operating, monitoring and maintaining all Products consistent with all Rockwell Automation or third-party provided instructions, warnings, recommendations and documentation;
- ensuring that only properly trained personnel use, operate and maintain the Products at all times;
- staying informed of all Product updates and alerts and implementing all updates and fixes; and
- all other factors affecting the Products that are outside of the direct control of Rockwell Automation.

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

Throughout this manual we use the following 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.

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

**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
- recognize the consequence

**SHOCK HAZARD**
Labels may be located on or inside the drive to alert people that dangerous voltage may be present.

**BURN HAZARD**
Labels may be located on or inside the drive to alert people that surfaces may be dangerous temperatures.
Basic PLC Programming with Micro800 Controllers & PanelView 800 Graphic Terminals

Contents

Before you begin ........................................................................................................................................... 5
About this lab ................................................................................................................................................... 5
Tools & prerequisites ...................................................................................................................................... 6
Lab setup .......................................................................................................................................................... 6
Get familiar with the Connected Components Workbench design environment ........................................ 6
Create a CCW project starting with a Micro850 controller ......................................................................... 12
Program a motor seal-in circuit .................................................................................................................... 19
Build and Download your Micro850 Application ....................................................................................... 29
Debug your Micro850 program ..................................................................................................................... 33
PanelView 800 HMI Design using Connected Components Workbench .................................................... 52
About This Lab ............................................................................................................................................... 52
Add the Terminal and Setup Communications ............................................................................................. 52
Create and Use Tags ...................................................................................................................................... 54
Create the Motor Control Screen .................................................................................................................. 58
Create a Push Button .................................................................................................................................... 61
Create a Goto Config Button ....................................................................................................................... 66
Create a Multistate Indicator ........................................................................................................................ 68
Create a Numeric Entry .................................................................................................................................. 72
Generate a Report .......................................................................................................................................... 74
Validate and Transfer the Application to the Terminal .................................................................................. 76
Learn About Additional Features of the PanelView 800 (Optional Sections) ............................................ 79
Create a GoTo Screen Button ....................................................................................................................... 80
Learn how to Create and Use User Defined Objects .................................................................................. 84
Create a UDO ................................................................................................................................................. 84
Exporting a UDO ................................................................. 86
Importing a UDO ............................................................... 87
Language Switching ............................................................ 89
Learn how to Use Alarms .................................................... 100
Learn how to Create a Trend and Configure a Data Log .... 105
Create a Trend Object .......................................................... 105
Configure a Data Log .......................................................... 110
Learn how to Use Recipes .................................................. 115
Building a Recipe ................................................................ 115
Building a Recipe Display ..................................................... 118
Learn how to Add Security to the Terminal ...................... 124
Before you begin

The following should have already been verified with your demo kit by the lab instructor prior to the lab:

1. USB cable connected between PC and Micro850 controller.
2. Ethernet cable connected directly between PC and PanelView 800 terminal.
3. Micro850 controller firmware at v8.011 (or higher) with static IP address configured for 192.168.1.3.
4. PanelView 800 terminal with static IP address configured for 192.168.1.2 and subnet mask 255.255.255.0.
5. PowerFlex 4M drive parameter settings:
   - P106=2
   - P108=2

   To verify:
   1. Press Esc on the PowerFlex 4M drive keypad multiple times until 0.0 is displayed.
   2. Press Sel until the leftmost alphanumeric character is flashing.
   3. Press the up or down arrow until the leftmost alphanumeric character being displayed is a flashing P.
   4. Press the enter key. The rightmost digit will be flashing.
   5. Press the up or down arrow until P106 is displayed.
   6. Press the enter key. The value of P106 is displayed.
   7. If the value displayed for P106 is 2, jump to step 10.
   8. Press the enter key. The value displayed for P106 will be flashing.
   9. Press the up or down arrow until a flashing 2 is displayed. Press the enter key to accept this new value.
   10. Press Esc – P106 is displayed and the 6 will be flashing.
   11. Press the up arrow two times so that P108 is displayed.
   12. Press the enter key. The value of P108 is displayed.
   13. If the value displayed for P108 is 2, jump to step 16.
   14. Press the enter key. The value displayed for P108 will be flashing.
   15. Press the up or down arrow until a flashing 2 is displayed. Press the enter key to accept this new value.
   16. Press Esc multiple times until 0.0 is displayed.

6. Demo kit power cycled off and back on.

About this lab

Connected Components Workbench (CCW) is the integrated design environment software package that is used to program, design, and configure your Rockwell Automation Connected Components devices such as Micro800 programmable logic controllers, PowerFlex variable frequency drives, Kinetix 3 servo drives, SMC soft-starters, Guardmaster software configurable safety relays and PanelView 800 operator interface terminals.

This lab will guide you in configuring a PowerFlex 4M drive, programming a Micro850 controller, and creating and downloading a PanelView 800 terminal application, all using Connected Components Workbench software.
This lab takes approximately 75 minutes to complete (not including optional sections).

**Tools & prerequisites**
- Software: Connected Components Workbench (CCW) Version 8.00 software – Developer Edition
- Hardware: Connected Components demo kit – DEMO-CCMICRO1

**Lab setup**

Get familiar with the Connected Components Workbench design environment

This section will help get you familiar with the Connected Components Workbench design environment. As our goal to help simplify your engineering efforts, we’ve developed CCW using the Microsoft Visual Studio Shell. This common and popular software shell provides you the benefits of a common look, feel, and design environment when transitioning from other similar software packages. Let’s take a couple minutes to get familiar with the CCW design environment.

(If Connected Components Workbench software is already open, jump directly to Step 2.)

1. Start the Connected Components Workbench (CCW) software.

   ![Connected Components Workbench](image)

   Double-click the Connected Components Workbench shortcut icon on your desktop.
2. Get familiar with the CCW design environment.

This is the default project layout. Below are descriptions of each of the panels’ contents and the general task the pane is used for.
3. Project Organizer

The 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, as well as double-click them to display their contents.

If your project contains a Micro800 controller, the Project Organizer also displays the logic programs, variables, and user-defined function blocks associated with that Micro800 controller.
4. Device Toolbox

The Device Toolbox displays all of the devices that you can add to your Connected Components Workbench™ project. From the Device Toolbox, you can select devices for your project from the following two tabs:

- Discover - discovers devices that are connected to your computer and recognized by Connected Components Workbench.
- Catalog - browses a list of devices that are included with Connected Components Workbench.
5. Toolbox

The Toolbox displays icons for items that you can add to programs. From the Toolbox, you can drag and drop individual Toolbox elements onto a design view surface or copy and paste these into a code editor. Above is an example of what is available in the Toolbox when editing a Ladder Diagram program.

There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
6. Output

Click on the Output tab. In the Output window, you can view and manage general purpose and debug messages generated by the various features of Connected Components Workbench. From the Output window, you can do the following:

• Review status messages
• Locate errors within programs
Create a CCW project starting with a Micro850 controller

In this section, you will create a new CCW project and learn how to:

- Add a Micro850 controller to your project
- Add and configure plug-in modules for the Micro850 controller
1. Add a Micro850 to your project.

Locate the Device Toolbox (upper right-hand corner). Expand Catalog and locate the Controllers folder. Expand the Controllers folder and locate the Micro850 folder. Expand the folder and double click on catalog number 2080-LC50-24QBB. This will add a Micro850 controller to your Project.

If the following window appears for you to select Major revision, select ‘8’ and click OK.
2. Notice that the Micro850 shows up both in the Project Organizer on the left-hand side as well as a large graphic in the middle.

3. Next, the plug-in modules used by this lab need to be added to the controller and configured. This model of the Micro850 controller has three plug-in slots available on the front of the controller. Plug-ins can be added to provide analog inputs and/or outputs, additional digital inputs and/or outputs, additional communication ports and other specialty functions.
4. Right click on the middle plug-in slot on the large Micro850 graphic and select a **2080-IF4** analog input plug-in module.
5. Using the pull-down menu, change the 2080-IF4 Channel 0 Input Type from Current to Voltage.
6. Similarly, right click on the third plug-in slot, select a **2080-OF2** analog output plug-in module, and change the 2080-OF2 Channel 1 Output Type to **Voltage** and the Output State to **Enabled**.
7. Note that the physical Micro850 controller in the demo box also has a 2080-SERIALISOL plug-in module installed in the first plug-in slot, but because we are not going to use this serial communications plug-in for this lab, we don't need to include it in this project.

8. Now you are ready to program the Micro850 controller to operate the drive, which controls the motor.
Program a motor seal-in circuit

In this section, you will add a ladder diagram program to the Micro850 controller and program a motor seal-in circuit.

1. Add a Ladder Diagram program.

   Right-click Programs under the Micro850 in your Project Organizer, and select Add → New LD : Ladder Diagram.
2. Notice a new Ladder Diagram program called **Prog1** has been added under Programs.

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

Since we'll be creating a Motor Circuit in this program, let's rename it **Motor_Circuit**.
3. Right-click the **Prog1** program icon and select **Rename**.
4. Rename the program 'Motor_Circuit'.

5. Create a motor seal-in circuit in your Motor_Circuit Ladder Diagram program.

This circuit will use the DI0 (Digital Input 0) switch on the Demo box as your Start switch, and the DI1 (Digital Input 1) switch as your Stop switch. The start motor control is wired to DO9 (Digital Output 9) so that when DO9 is turned on, the motor accelerates and runs, and when DO9 is turned off, the motor decelerates to a stop.
6. Double-click the **Motor_Circuit** program icon. A ladder diagram editor will appear in the main project workspace with one empty rung.

![Ladder Diagram Editor](image)

7. You do not need the Device Toolbox window for the time being. Toggle 📦 at the top right-hand corner of the window to hide it.

![Device Toolbox](image)
8. The Toolbox provides all the elements needed for programming a Ladder Diagram. The elements include rung, instruction block, branch, coil, contact, return and jump as defined below.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rung</td>
<td>Represents a group of circuit elements that lead to the activation of a coil.</td>
</tr>
<tr>
<td>LD instruction block</td>
<td>A block element can be a function block, a function, a user-defined function block, or an operator.</td>
</tr>
<tr>
<td>Branch</td>
<td>Two or more instructions in parallel.</td>
</tr>
<tr>
<td>Coil</td>
<td>Represents the assignment of outputs or internal variables. In an LD program, a coil represents an action.</td>
</tr>
<tr>
<td>Contact</td>
<td>Represents the value or function of an input or internal variable.</td>
</tr>
<tr>
<td>Return</td>
<td>Represents the conditional end of a diagram output.</td>
</tr>
<tr>
<td>Jump</td>
<td>Represents the conditional and unconditional logic in the LD program that controls the execution of diagrams.</td>
</tr>
</tbody>
</table>

9. Locate the Direct Contact instruction in the Toolbox pane (right-hand side), and drag-and-drop it onto the left side of the first ladder rung.

After inserting the Direct Contact instruction, the Variable Selector window will automatically pop-up, allowing you to select the variable or input/output point to assign to this instruction.
10. In the Variable Selector window, select the I/O – Micro850 tab.
11. We will be assigning an embedded input point to this instruction. Embedded Input/Output (I/O) variable names start with '_IO_EM_', followed by DI for digital input or DO for digital output, concluding with the input or output point number, starting with 0. Select _IO_EM_DI_00 for embedded input 0 and then, in the ‘Alias’ column of _IO_EM_DI_00, type ‘Start Motor switch’ and click OK.

12. Your rung should look like the following.
13. Locate the **Direct Coil** instruction in the Toolbox, and drag-and-drop it onto the far right side of the rung, and assign it to the embedded output point _IO_EM_DO_09 with alias description ‘Motor ON/OFF’. Your rung should look like the following.

14. Locate the **Reverse Contact** instruction in the Toolbox and drag-and-drop it onto your rung, just to the right of the Direct Contact and assign it to embedded input point _IO_EM_DI_01 with alias description ‘Stop Motor switch’. Your rung should look like the following.

15. Locate the **Branch** instruction in the Toolbox and drag-and-drop it on top of the Direct Contact on the far left of the rung. Your rung should look like the following.

16. Drag-and-drop a **Direct Contact** onto the Branch that you just added, and assign it to the embedded output point _IO_EM_DO_09. Notice that output points can be assigned to contacts as well as to coils. Your rung should look like the following.
17. It is always a good programming practice to document your program rungs so that when you or someone else looks at the rung in the future, it will be clear what the function of the rung is. Double click within the green rung comment box just above the rung and type in “Switch DI0 turns the motor ON and switch DI1 turns the motor off.”

18. Congratulations - you’ve completed creating your motor seal-in circuit! When the Start Motor switch is momentarily turned on (while the Stop Motor switch remains off), you complete the rung circuit power flow from left to right to the coil so that the Motor output turns ON. Once the motor is running, you can release the Start Motor switch because the branch circuit around the switch contact “seals” it in and the only way to interrupt the circuit is to momentarily turn on the Stop Motor switch. This breaks the circuit power flow, which turns the Motor output OFF and drops out the seal-in branch circuit. The Stop Motor switch can then be released and the Motor output remains OFF until the Start Motor switch is turned on again.
Build and Download your Micro850 Application

In this section, you will build and download your Micro850 application to the controller.

Before you can download a project to the controller, you must build it to verify that there are no errors with the programming.

1. Build your application by right-clicking the Micro850 in your Project Organizer, and selecting Build.

When the build is complete, you will see a message in the lower left-hand corner that the build has succeeded. If there were errors in your programming, then they would be listed in the Error List panel and clicking on the error would direct you to the error in your project.

Now that your build has completed, you can download the project to your controller.
2. Download the project to your Micro850 by clicking the **Download** button.

3. The **Connection Browser** window will appear. Browse for your controller by expanding **USB** and selecting **16, Micro850, Micro850**, then clicking **OK**.
4. The **Download Confirmation** window will appear. You will be prompted to overwrite the project in the controller. The two choices are **Download** and **Download with Project Values**. The second choice is a new feature in Release 7 (and later) of Connected Components Workbench. Now, when a project is uploaded from the controller, a snapshot of the values of all of the user variables is uploaded and saved. These values can be downloaded back into the controller along with the project by selecting **Download with Project Values**. If you clicked the Help button, you would find that the following conditions must be met in order for a Project Value to be downloaded to a variable:
   - Variables with both a Project Value and an Initial Value will be initialized with their Initial Value.
   - Variables with an Attribute of "Read" or "Write" cannot be downloaded. Variables must have an Attribute of "Read/Write".
   - Variables with a Direction of "VarInput" cannot be downloaded.
   - Complex variables (structures, arrays) that contain members with Initial Values cannot be downloaded.

5. In this lab, we have a new project and none of the variables have a Project Value assigned to them yet. Therefore, just click **Download**.

6. The download will proceed. When the download is complete, you will be prompted to put the controller back in Remote Run mode. Click **Yes**.
7. You have completed downloading to your Micro850 controller. Proceed to the next section to test your project.
Debug your Micro850 program

In this section, you will learn how to debug your Micro850 program. By debugging your program, you can view your program visually in real time and watch values change in the program, as well as visually debug your Ladder Logic or Function Block Diagram.

1. Notice that at the conclusion of the download, the software went directly into Connected mode, as highlighted below, and is now displaying the real-time status of rung 1. Red contacts indicate that power can flow through them, while blue contacts indicate that the power flow is blocked. Red coils indicate that they are turned on, while blue coils are turned off. Currently the Motor coil is off (blue) because power is blocked by both the blue Start Motor switch contact and the blue Motor ON/OFF contact.

2. Turn and release the DI0 switch on the Demo box. Notice the _IO_EM_DI_00 Direct Contact instruction turn red as you turn on the switch, and then turn blue as you release it (if you turn and release the switch too fast, you may not see the color update in the ladder diagram). Then notice the _IO_EM_DO_09 Direct Contact and Direct Coil instructions turn red. Verify that the RUN LED on the drive display is on, although the motor will not be spinning.

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 is turned on.
3. Turn and release the DI1 switch on the Demo box to turn off the output. Notice the drive RUN LED is now off and the corresponding changes in your Ladder Diagram.

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

Since the variables we're working with right now are embedded I/O points, we need to open the Global Variables list.
5. Double-click **Global Variables** in your Project Organizer. The Global Variables pane will launch in a new tab in the main project workspace.

![Project Organizer with Global Variables pane](image)

6. Locate the `_IO_EM_DO_09` embedded I/O variable in the Global Variable list, and notice that the logical value checkbox is empty.

![Global Variables pane with _IO_EM_DO_09 variable](image)
7. Turn and release the DI0 switch on the Demo box. Notice there is now a checkmark in the Logical Value for _IO_EM_DO_00. You may have also noticed a checkmark appear in the _IO_EM_DI_00 logical value as you turned on the switch and then noticed the checkmark go away as you released the switch.

8. Click on the Motor_Circuit-POU (POU stands for Program Organization Unit) tab to return to viewing your ladder diagram.

9. You have completed debugging your motor seal-in circuit, but what we really want to accomplish is for the motor to spin when it is turned on and to be able to vary the speed that the motor spins at. The drive is configured to vary its speed based on the voltage present at its 0-10v analog input. The motor currently doesn't spin when turned on because the Micro850 analog plug-in output point that connects to the drive analog input point needs to be programmed to output something other than 0 volts.

One of the new features of Connected Components Workbench Developer Edition version 8.0 is Run Mode Change (RMC). This feature will allow us to implement program changes without having to stop the process being controlled by the Micro850 controller (i.e., the controller remains in RUN mode).

10. Click on the Run Mode Change icon in the upper right Run Mode Change toolbar. Notice that the rung no longer displays the blue and red status colors and the Toolbox re-appears on the right for programming.
11. Double click on **Rung** in the Toolbox to add a new rung. Alternatively, you could have clicked on Rung and dragged the new rung under the existing rung. A third way to add a new rung is to use the keyboard shortcut <Ctrl>0.

For a list of all of the keyboard shortcuts available when programming in Ladder Diagram, search Connected Components Workbench software Help for ‘LD language keyboard shortcuts’.

The Demo kit has a dial labeled **SPEED COMMAND** along the right hand side. This dial is actually a voltage potentiometer and is wired to the first analog input on the Micro850 analog input plug-in module. When the dial is turned clockwise, it increases the voltage at the analog input and when it is turned counter-clockwise, it decreases the voltage at the analog input. The goal of our new rung is to assign the value received at the analog input to the analog output connected to the drive. That way, turning the potentiometer will vary the drive speed.
12. Drag and drop an **Instruction Block** from the Toolbox onto rung 2. The Instruction Block Selector screen opens up automatically to allow you to select which of the more than 150 instructions you would like to choose from. By default the instructions are sorted alphabetically by Name, but you may also sort them by Type or Category by clicking on those column headings. Since we want to assign the value of one variable (the analog input) to another (the analog output), type ‘**mov**’ in the Search box for the **MOV(e)** instruction and click **OK**.
13. In order to find out more information about the MOV instruction block, click F1 on your keyboard to bring up Help on the instruction. Once Help opens up (be patient, it may take a while to open the first time), expand the Arguments to see that MOV can be used with many different data types, but does have the restriction that the data type of the input \((i1)\) and output \((o1)\) variables must match.

![MOV](image)

**MOV**

MOV moves a copy of the value in input \((i1)\) to the output \((o1)\).

**Tip:** The MOV instruction displays in the Block Selector when it is launched from a Ladder Diagram POU or a Function Block Diagram POU, but it does not display in a Structured Text POU. Structured Text programs use the " := " assignment operator instead of the MOV function.

### Arguments

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter Type</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN</td>
<td>Input</td>
<td>BOOL</td>
<td>When (EN = \text{TRUE}), execute the direct link to an output computation. When (EN = \text{FALSE}), there is no computation. Applies only to LD programs.</td>
</tr>
<tr>
<td>(i1)</td>
<td>Input</td>
<td>BOOL - DINT - REAL - TIME - STRING - SINT - USINT - INT - UINT - UDINT - LINT - ULINT - DATE - LREAL - BYTE - WORD - DWORD - LWORD</td>
<td>Input and output must use the same data type.</td>
</tr>
<tr>
<td>(o1)</td>
<td>Output</td>
<td>BOOL - DINT - REAL - TIME - STRING - SINT - USINT - INT - UINT - UDINT - LINT - ULINT - DATE - LREAL - BYTE - WORD - DWORD - LWORD</td>
<td>Input and output must use the same data type.</td>
</tr>
<tr>
<td>ENQ</td>
<td>Output</td>
<td>BOOL</td>
<td>Enable out. Applies only to LD programs.</td>
</tr>
</tbody>
</table>
### The elementary IEC 61131-3 data types supported by the Micro800 are defined as follows:

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOOL</td>
<td>Boolean, True or False</td>
</tr>
<tr>
<td>SINT</td>
<td>8-bit signed integer, range -128 to +127</td>
</tr>
<tr>
<td>USINT</td>
<td>8-bit unsigned integer, range 0 to 255</td>
</tr>
<tr>
<td>BYTE</td>
<td>8-bit unsigned integer, range 0 to 255</td>
</tr>
<tr>
<td>INT</td>
<td>16-bit signed integers, range -32768 to 32767</td>
</tr>
<tr>
<td>UINT</td>
<td>16-bit unsigned integers, range 0 to 65535</td>
</tr>
<tr>
<td>WORD</td>
<td>16-bit unsigned integers, range 0 to 65535</td>
</tr>
<tr>
<td>DINT</td>
<td>32-bit signed integers, range -2147483648 to +2147483647</td>
</tr>
<tr>
<td>UDINT</td>
<td>32-bit signed integers, range 0 to 4294967295</td>
</tr>
<tr>
<td>DWORD</td>
<td>32-bit signed integers, range 0 to 4294967295</td>
</tr>
<tr>
<td>LINT</td>
<td>64-bit signed integers, range -9223372036854775808 to 9223372036854775807</td>
</tr>
<tr>
<td>ULINT</td>
<td>64-bit unsigned integers, range 0 to 18446744073709551615</td>
</tr>
<tr>
<td>LWORD</td>
<td>64-bit unsigned integers, range 0 to 18446744073709551615</td>
</tr>
<tr>
<td>REAL</td>
<td>32-bit floating values, range -3.40282347E+38 to 3.40282347E+38</td>
</tr>
<tr>
<td>LREAL</td>
<td>64-bit floating values, range -1.7976931348623157E+308 to 1.7976931348623157E+308</td>
</tr>
<tr>
<td>TIME</td>
<td>32-bit words, range 0 to 1193h2m47s294ms</td>
</tr>
<tr>
<td>DATE</td>
<td>32-bit words, range 1970-01-01 to 2038-01-18</td>
</tr>
<tr>
<td>STRING</td>
<td>Up to 252 characters, can contain ASCII codes from 0 to 255</td>
</tr>
</tbody>
</table>

14. Minimize the Help screen and return back to the ladder diagram.

15. Hover your mouse cursor over one of the yellow warning triangles and note that the tooltip indicates that it is ‘Expecting variable name. Referring to undefined variable.’ This is expected because no variable has been entered yet for i1 or o1.
16. Click the mouse in the upper half of the gray input variable block (to the left of i1) in order to bring up the variable selection dropdown menu, scroll down to _IO_P2_AI_00 and Enter. Recall that this is the variable assigned to the first analog input point on the plug-in module installed in slot 2.

17. Similarly, click the mouse in the upper half of the gray output variable block (to the right of o1), scroll down to _IO_P3_AO_01 and Enter. Double click within the green rung comment box just above the rung and type in “Assign the value from the SPEED COMMAND potentiometer (analog input 0) to the drive (analog output 1) to control the motor speed.”

18. Now you are ready to test the changes you made in the running system. Click the Test logic Changes icon immediately to the right of Run Mode Change.
19. Note that the changes are built and downloaded to the controller and we can immediately monitor how the new rung is operating. Now when you turn the SPEED COMMAND potentiometer, you will see the value of variable _IO_P2_AL_00 change within the MOV instruction input parameter block, which is immediately moved into variable _IO_P3_AO_01 (as seen in the MOV instruction output parameter block), which visibly changes the rotating speed of the motor (with a slight acceleration or deceleration lag).

20. Once the changes have been tested using Run Mode Change, you have two options: Accept Changes, which saves the changes that have been made, or Undo Changes, which deletes the changes that have been made. The icons for Accept Changes and Undo Changes are immediately to the right of the Test logic Changes icon.

21. Since the changes are working as designed, normally we would Accept Changes at this point to make them permanent. However, in preparation for the later PanelView 800 terminal programming portion of the lab, you are going to program a different rung 2 that will adjust the speed of the motor based on a frequency value entered by you from the terminal. In addition, you need to make a couple of modifications to rung 1 so that you can start and stop the motor from the terminal, in addition to the DI0 and DI1 switches. We will start with those changes first.

22. Click the Undo Changes icon. Once the Undo Changes is done, click Run Mode Change again.
23. Add a new branch underneath the existing branch on rung 1 and add a Direct Contact on the new branch. When the Variable Selector screen opens up, click on the User Global Variables – Micro850 tab, enter in ‘Start_Motor’ for the variable name and click OK. This creates a new Global Variable named Start_Motor that the PanelView 800 will be able to write to for starting the motor.

![Variable Selector](image1.png)

![Ladder Diagram](image2.png)
24. Next add a Reverse Contact between the current Reverse Contact and Coil on rung 1. Create and assign a new Global Variable named Stop_Motor to the new Reverse Contact.
25. Add a new rung and place a SCALER Instruction Block on the rung.

26. As you have already seen, the analog output that adjusts the motor speed is controlled by I/O variable _IO_P3_AO_01 that ranges from 0-65,535 in order to vary the motor speed from 0.0-60.0 hertz. The desired motor speed in hertz will be written to a new variable from the PanelView 800 terminal. The SCALER instruction will continuously scale the 0.0-60.0 value supplied by the terminal to the 0-65,535 value required by the analog output variable.

27. One thing that you must keep in mind when assigning variables to instruction blocks is the required data type of each input and output. Hover the mouse pointer over the SCALER instruction block and you will get a tooltip that summarizes the input and output variable data types. Notice that all of the variables must be of data type REAL.
28. Double click in the lower half of the variable block to the left of Input and create a new User Global Variable named Speed_Command. Notice that the data type field is automatically filled in as REAL for you.
29. For the other input blocks, enter in the REAL constant values as shown below. Note that REAL constants must always include a decimal point and at least one number to the right of the decimal point in order to distinguish them from integer constants.

![Diagram of SCALER_1 block with constants entered]

30. Ideally we would just enter in variable _IO_P3_AO_01 in the output block, but notice what happens when you do this.

![Diagram of SCALER_1 block with _IO_P3_AO_01 entered]
31. If you left it this way and tried to build the project, you would get a build error with a similar description. Instead, you need to assign a new variable of data type REAL to Output, then convert the REAL value of that variable into a UINT value that can be assigned to _IO_P3_AO_01. Create and assign a new REAL variable to the output block named AO_REAL.

![Diagram showing SCALER block with parameters: Speed_Command, 0.0, 60.0, 0.0, 65535.0.](image)

32. Add a new Instruction Block to the right of SCALER and select ANY_TO_UINT. As the name implies, this instruction block takes an input variable of any data type and converts it into an output variable of data type UINT. Assign AO_REAL as the input variable and _IO_P3_AO_01 as the output variable. Include the following rung comment for your new rung – “The value of Speed_Command in hertz, which will be entered on the PV800 terminal, is scaled and assigned to the 2nd analog output on the 3rd plug-in module to control the speed of the motor.” (To save typing time, copy the text above and paste it into the rung comment.)

![Diagram showing SCALER and ANY_TO_UINT blocks with parameters: Speed_Command, 0.0, 60.0, 0.0, 65535.0.](image)
33. Click the **Test logic Changes** icon (to the right of **Run Mode Change**) and once the change is implemented, double click in the Speed_Command block and enter 10.0 for the **Logical Value** for Speed_Command.

34. Click the **Logical Value** checkbox for Start_Motor and verify that the motor starts running. Note that due to tolerances in the electronic components in the system, the drive may display a speed value slightly different from 10.0.

35. Click the **Logical Value** checkbox again for Start_Motor to uncheck it, then check the **Logical Value** checkbox for Stop_Motor and verify that the motor stops spinning. Uncheck Stop_Motor before closing the Variable Monitoring screen.
36. Now that your changes are verified, click the **Accept Changes** icon.

![Accept Changes icon](image)

37. Click the **Connected** down arrow and select **Disconnect**.

![Connected options](image)

38. Another new feature of Connected Components Workbench Developer Edition version 8.0 is the ability to archive multiple copies of a project. Especially during development, it can be useful to be able to go back to a previous version of the project if changes that have been made need to be undone. It also provides a chance to document changes from one version of the project to the next. Archive a copy of your current project by selecting **File ➜ Archive Manager**...
39. Enter “Simple motor circuit program.” for the Description and click Archive.

40. Click Close.

Congratulations! You have now completed the Micro800 section of the lab!
PanelView 800 HMI Design using Connected Components Workbench

About This Lab
Now that you've learned the basics of creating a Micro800 program, let's create a PanelView800 application that will work with it.

Add the Terminal and Setup Communications
Add a PanelView 800 terminal to the project and then configure the application.

1. From View → Device Toolbox, expand Catalog, Graphic Terminals and PanelView 800 to bring up the list of PanelView 800 HMI catalog numbers. Double-click on 2711R-T7T to add it to the Project Organizer.

2. Double-click the Graphic Terminal (Labeled as PV800_App1*) icon in Project Organizer to bring up the PanelView 800 device configuration screen.

3. Double check that the Landscape orientation is highlighted then select OK.
4. On the **Settings** tab, go to the **Protocol** menu located under **Communication**. Verify that the protocol selected is **Serial | Allen-Bradley CIP** as shown.

![Protocol Menu](image)

5. Leave the rest of the communication settings at the default state. Notice that the name of the controller we will be using is **PLC-1**.

<table>
<thead>
<tr>
<th>Name</th>
<th>Controller Type</th>
<th>Address</th>
<th>Timing</th>
<th>Auto-Denot</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLC-1</td>
<td>Micro800</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. Rename the PV800_App1* as **“MotorStarter”** by right-clicking on the icon and selecting **Rename** from the menu.

![Rename Menu](image)
Create and Use Tags

This application uses read and write tags to define how objects interact with the addresses of the Micro850 controller. You must create these tags before you can assign them to the objects in the application. Only objects that interact with the controller require a tag. Objects such as screen navigation buttons, drawings, and screen text do not require tags.

1. Double-click on the tags icon to open the Tag Editor.

2. Verify that the External tab is selected.

3. Click the Add button to add a tag.

4. Click in the Address field and select the (...).
5. Select **Speed_Command** from the list of variables and click **OK**.

6. The **Data Type** is automatically filled in when the tag is selected.

7. Type "**Speed_Command**" in the **Tag Name** field.

   ![Variable Selector](image)

   **Note:** The Tag Name and Address do not have to match, but it is good programming practice.

8. Click in the **Controller** field and select **PLC-1**. This is the controller name defined on the Communication tab.

   ![Variable Selector](image)

9. Optionally, enter the tag description.
10. Repeat steps 3 through 9 to add the **Start_Motor** and **Stop_Motor**.

11. Add one more tag by selecting the **Add** button.

12. Select the (...) in the **Address** field.

13. Select the **I/O – Micro850** tab.

14. Select the **_IO_EM_DO_09** tag and select **OK**.

15. Type “**Motor_Ind**” in the **Tag Name** field.
16. Click in the **Controller** field and select **PLC-1** and optionally, enter a description.

![Controller field example](image)

17. When done, the **Tag Editor** should look like this.

![Tag Editor example](image)

18. Save the application by selecting the **Save** icon (Looks like a disk) from the toolbar.
Create the Motor Control Screen

New applications open with the first screen created. Each screen is created with a default name, Screen_NN, where NN is a number from 1 to 99. The first screen is named Screen_1. You can rename the screen using a maximum of 50 characters.

This is what the Motor Control screen will look like when the following sections are complete.

1. Right-click 1- Screen 1 and select Rename.

2. Type “MOTOR_CONTROL” into the text field as shown below and press Enter.
3. If the screen tab is not open already, double click the **screen icon** next to the newly named screen to open it for editing. It should look similar to this:

![Screenshot of screen icon](image)

Verify the screen border is highlighted (in red) indicating the screen is selected. You can modify screen properties as long as nothing on the screen is selected.
4. Select **View → Properties Window** from the tool bar to view the Properties Window. It will show up on the right side of the screen.

<table>
<thead>
<tr>
<th>View</th>
<th>Device</th>
<th>Debug</th>
<th>Layout</th>
<th>Tools</th>
<th>Communications</th>
</tr>
</thead>
</table>
| Project Organizer | Ctrl+Alt+P
| Device Toolbox  | Ctrl+Alt+D
| Toolbox        | Ctrl+Alt+X
| Error List     | Ctrl+Alt+E
| Output         | Ctrl+Alt+O
| Quick Tips     | Ctrl+Alt+Q
| User-defined Object Library |  
| Validation Results |  
| Cross Reference Browser | Ctrl+W, Ctrl+C
| Document Overview |  
| Parameters |  
| Toolbars |  
| Full Screen | Shift+Alt+Enter

![Properties Window](image)

**Note:** The Properties Window can also be activated by right-clicking on the object, in this case the screen, and selecting **Properties**. If you wanted to see the Properties of a push button, right-click on that button and select Properties.

5. Set the **Horizontal Grid Spacing** and **Vertical Grid Spacing** to 5.

![Screen Properties](image)

**Note:** From the Properties Window, the Screen Name can also be edited, among other basic settings. Feel free to take the time to explore.
Create a Push Button

1. From the Toolbox, locate *Momentary Push Button*, and drag-and-drop it onto the middle of your screen.

   ![Toolbox Image]

   *Note: The “Toolbox” is context sensitive. It changes the selection of objects for HMI development versus the controller program development.*

2. Double-click the *button* to open the *States* editor. Each row is a state with a default value, text, and other format options. Move the scroll bar to see additional options. It should look similar to the one shown below:

   ![States Editor Image]

3. Edit State 1:

   ![Edited States Image]

   - Click in the *Caption Text* field and type “START MOTOR”
- Click (...) button in the **Background Color** field and select **green** from the color palatte.

- Click the (...) button in the **Caption Text Color** field and select **black**.

- Check the **Caption Font Bold** checkbox.

- Change the **Caption Font Size** to “28”.

---

62 of 134
4. Edit State 2:
   - Click in the Text field and type “MOTOR STARTED”
   - Select green as the Background Color
   - Select black as the Caption Text Color.
   - Check the Caption Font Bold checkbox.
   - Change the Caption Font Size to “28”.

5. Click row 1 so that this is the state displayed on the screen in CCW.

6. Verify your States window matches the image below and click OK to close the window and apply the changes.

7. Make sure the button is still selected and go to the Properties window. (View ➔ Properties)

8. Configure the following properties as shown below:
   - Height: 120
   - Left: 0
   - Top: 360
   - Width: 200
9. Click the **Write Tag** drop down arrow in the **Connections** section and select **Start_Motor** from the list of tags.

The **Start Motor** button is complete. Next, create the **Stop Motor** button by using a copy and paste operation.

### Create Stop Motor

8. **Verify the Start Motor push button is selected.**

![Start Motor Button](image)

10. **Click the Copy icon ( ) on the toolbar or press the Ctrl+C keys on your keyboard.**

11. **Click the Paste icon ( ) on the toolbar or press the Ctrl+ keys.**

12. **Move the pasted button to the right of the original push button.**

![Copy Paste Buttons](image)

13. **Double click the new button to open the States editor.**

14. **Edit state 1:**
   - Replace the **Caption Text** with “STOP MOTOR”
   - Select red as the **Background Color**
   - Select white as the **Caption Text Color**
   - Double check the **Caption Font Bold** checkbox is checked.
   - Double check the **Caption Font Size** to “28”.

15. **Edit State 2:**
   - Replace the **Caption Text** with “MOTOR STOPPED”
   - Select red as the **Background Color**
   - Select white as the **Caption Text Color**
   - Double check the **Caption Font Bold** checkbox is checked.
   - Double check the **Caption Font Size** to “28”.
16. Click **row 1** so that this is the state displayed on the screen in CCW.

17. Click **OK** when done.

18. Make sure the button is still selected and go to the **Properties** window.

19. Assign tags to the **Stop Motor** push button by clicking the drop down arrow for the **Write Tag**, on the **Properties** window. Assign tag **Stop_Motor** by clicking on it.

20. Verify your screen appears as shown below.

21. Save the application by selecting the **Save** icon ( ) from the toolbar.
Create a Goto Config Button

The Goto Config button allows access to the configuration mode screens of the terminal when the button is pressed at runtime. Each application should contain at least one Goto Config button if the operator needs to access the configuration screens.

1. From the Toolbox, scroll down to the Advanced section, and find the Goto Config button. Click and drag it to the center of the screen.

2. Make sure the button is still selected and go to the Properties window.

3. We will use the default colors for the object.

4. Configure the following properties as shown below:
   - **Height**: 120
   - **Left**: 400
   - **Top**: 360
   - **Width**: 200
   - **Text**: “CONFIG SCREEN”
   - **Font Bold**: True
   - **Font Size**: 28
5. Verify the screen looks as shown below.

6. Save the application by selecting the Save icon ( ) from the toolbar.
Create a Multistate Indicator

The Multistate indicator will show whether the motor is running or stopped. You will be creating the indicator.

1. From the Toolbox, locate the **Multistate Indicator** object listed in the Display section.

   ![Multistate Indicator Toolbox](image)

2. Drag the object into the center of the screen.

3. Double-click the object to open the States editor. The indicator is created with five states. This lab only uses two.

4. Delete rows 3 and 4. (Select the row, and click the **Delete** button). Select **OK** when prompted.

5. Edit state 1:
   - Accept **0** as the Value.
   - Select **red** as the Background Color
   - Delete the default Caption Text.

   We will not enter any text since an image will be placed over the multistate-indicator.
6. Edit state 2:
   - Accept 1 as the Value.
   - Select green as the Background Color
   - Delete the default Caption Text.

   ![States window](image)

   *States*

   ![States window](image)

7. Click row 1 and then click OK to close the editor.

8. Make sure the button is still selected and go to the Properties window.

9. Configure the following properties as shown below:
   - **Height**: 170
   - **Left**: 110
   - **Top**: 90
   - **Width**: 250
   - **Read Tag**: Motor_Ind
10. From the Toolbox, locate the **Image** object listed in the Drawing Tools section.

![Image object in Toolbox](image)

11. Click and drag it onto the screen, directly over the MultiState Indicator.

![Image placement](image)

12. Double-click the “X” to open the image selection window.

13. Click on the **System** tab and locate the **2001R_standard_motor-standard_motor.png**

![Image selection window](image)

14. Click **Select** to apply the picture.
15. Configure the following properties as shown below:

- **Height**: 140
- **Left**: 125
- **Top**: 110
- **Width**: 225

16. Verify the screen looks as shown below

![Screen Image](image)

17. Save the application by selecting the **Save** icon (usband) from the toolbar.
Create a Numeric Entry
The Numeric Entry will allow the user to set the speed of the motor.

1. From the Toolbox, locate the *Numeric Entry* object.

2. Drag the object into the space to the right of the Multistate Indicator.

3. Make sure the button is still selected and go to the *Properties* window.

4. Configure the following properties as shown below:
   - **Height**: 110
   - **Left**: 450
   - **Top**: 120
   - **Width**: 230
   - **Indicator Tag**: Speed_Command
   - **Write Tag**: Speed_Command
   - **Number of Decimal Places**: 1
   - **Numeric Field Width**: 4
   - **Font Bold**: True
   - **Font Size**: 28
5. Verify the screen looks as shown below

6. Save the application by selecting the **Save** icon from the toolbar.
Generate a Report

In this section, you will learn how to generate a report. It provides detailed information about the PanelView 800 application.

1. Go to the Settings tab of the PanelView 800. If it is not open, double click the Motor Starter icon in the Project Organizer.

2. Click the Generate Report button.

3. A pop up window will appear with the results of the report.
4. Select the **Save** icon from the toolbar. There are two options: save it as an excel file, or as a PDF. **Select** an option.

![Save options](image)

5. Give the file a name and save it to the desktop.

![Save file dialog](image)

6. Once the file has been saved, locate it on the desktop. **Open** the file and view the report in the preferred format. Notice the report provides images of all the screens in the application, a list of tags used, all the settings configured for the controller, as well as data usage and much more.
**Validate and Transfer the Application to the Terminal**

Before you run an application on the PanelView 800, it is strongly recommended that you validate the application for errors and warnings. The application does not need to be error-free to run, however, it is good practice to fix both errors and warnings to ensure expected system behavior.

1. Right-click on the **MotorStarter** icon in the Project Organizer and select **Validate**. An alternate way to validate the application is by selecting the Validate button on the Settings tab.

   ![Validate button](image1)

   ![Validation Results window](image2)

2. The Validation Results window opens.

3. Correct any errors if desired, and close the Validation Window.

   ![Validation Results table](image3)

4. Re-validate the application after the errors have been fixed.

   Warnings still may exist, however they can be ignored as they are not functionality errors. Move onto the next step.

5. Right-click on the **MotorStarter** icon and select **Download**. Similar to the Validate button, a Download button is also on the Settings tab.

   ![Download button](image4)
6. **Save** any changes made if prompted to do so.
7. The Connection Browser will pop up.
8. Browse for the terminal by expanding **AB_ETHIP-1, Ethernet** and selecting the **2711R-T7T**.

![Connection Browser](image)

9. Then select **OK**.

10. The **Output** window will show the progress of the download. Ensure the application downloads successfully before moving forward.

```
Connecting to graphic terminal... Succeeded
Preparing device for download. This may take some time to execute... Succeeded
Downloading application............... Application downloaded successfully
```

11. Turn to your PanelView 800 terminal. It should be on the **Main** screen.

12. Select the **File Manager** button.

13. Scroll through the list of applications using the arrow keys to find the **MotorStarter** application you created.

14. Select **Run**. If prompted with a warning, select **Yes (F1)**.

15. The application will load. Select the **START MOTOR** button, and let the motor run. Notice the Multistate Indicator will change color from red to green behind the image of the motor.

16. Select the Numeric Input button you created, enter **"10.0"** into the keypad, and press **Enter**. Verify the display on the PowerFlex 4M drive is displaying a number close to 10.0, if not exactly.

17. Select the **STOP MOTOR** button.

18. Feel free to explore some more with the screen functionality.

19. To exit the application, select the **CONFIG SCREEN** button. This will bring you back to the File Manager screen.
Learn About Additional Features of the PanelView 800 (Optional Sections)

In this section of the lab, you will learn about additional features of the PanelView 800 that will help you create a fully functional application. The topics in this section cover Goto Screen buttons, User Defined Objects, Languages, Alarms, Trends, Data Logging, Recipes, and Security.
Create a GoTo Screen Button

Now that the MOTOR_CONTROL screen has been completed, let’s create another screen which will be used to explore more options within PanelView 800. We will add Goto Screen buttons, MORE… and MOTOR_CONTROL, to navigate between the Options and MOTOR CONTROL screens. Follow the steps below to create these buttons.

1. Create a new screen by right-clicking on Screens → Add Screen

2. Rename the new screen “Options” by right-clicking on it and selecting Rename.

3. From the Toolbox, locate the Goto Screen, click and drag the Goto Screen button to the middle of the screen.

4. Right-click the object and select Properties to open the Properties window.

5. Replace Goto with “MOTOR CONTROL” in the Text field.
6. Select *dark grey* as the **Background Color**.

   ![Appearance Settings]

<table>
<thead>
<tr>
<th>Alignment</th>
<th>Center</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background Color</td>
<td>#484850</td>
</tr>
</tbody>
</table>

7. In the **Navigation** section, select **1-MOTOR_CONTROL** as the **Screen**.

   ![Navigation Settings]

<table>
<thead>
<tr>
<th>Accept Focus</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Touch</td>
<td>True</td>
</tr>
<tr>
<td>Enable Touch Indication</td>
<td>True</td>
</tr>
<tr>
<td>Function Key</td>
<td>None</td>
</tr>
<tr>
<td>Screen</td>
<td>1-MOTOR_CONTROL</td>
</tr>
<tr>
<td>Touch Indication Color</td>
<td>#0E0E0E</td>
</tr>
</tbody>
</table>

8. Configure the following properties as shown below:
   - **Height**: 120
   - **Left**: 600
   - **Top**: 360
   - **Width**: 200
   - **Border Width**: 10
   - **Font Bold**: True
   - **Font Size**: 28

9. Verify the screen looks as shown below.
10. Right click on the button, and select **Copy**.

```
MOTOR CON
```

11. Go to the **MOTOR_CONTROL** screen by double clicking on it, right click, and select **Paste**.

12. Right-click the **object** and select **Properties** to open the Properties window, if it is not already opened.

13. Replace **MOTOR_CONTROL** with “MORE…” in the **Text** field.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Rectangle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>MORE...</td>
</tr>
<tr>
<td>Text Color</td>
<td>#F8F8F8</td>
</tr>
</tbody>
</table>

14. In the **Navigation** section, select **2 - Options** as the screen.

<table>
<thead>
<tr>
<th>Navigation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept Focus</td>
<td>False</td>
</tr>
<tr>
<td>Enable Touch</td>
<td>True</td>
</tr>
<tr>
<td>Enable Touch Indication</td>
<td>True</td>
</tr>
<tr>
<td>Function Key</td>
<td>None</td>
</tr>
<tr>
<td>Screen</td>
<td>2 - Options</td>
</tr>
<tr>
<td>Touch Indication Color</td>
<td>#008000</td>
</tr>
</tbody>
</table>

15. Configure the following properties as shown below:

- **Left**: 600
- **Top**: 360
- **Border Width**: 4
16. Verify the screen looks as shown below.

17. Save the application by selecting the Save icon ( ) from the toolbar.
Learn how to Create and Use User Defined Objects

The User Defined Object (UDO) allows the user to easily duplicate single objects as well as entire screens from one screen to the next within an application. A UDO can also be used in multiple applications by exporting it to the computer as a .chu file and then importing the file into the UDO library of the new application. In this section, follow the steps to create a UDO which we will use frequently throughout the rest of this application.

Create a UDO

1. First, open the UDO library. Go to View → User-Defined Object Library from the menu.

2. Go to the Options screen by double clicking the icon from the Project Organizer.

   We will create a UDO of the MOTOR CONTROL button, since it will be used frequently in the next few sections. UDO's can contain as little as one object, to an entire screen.

3. Right click on the MOTOR CONTROL button and select Add to Library
4. Type “MOTOR CONTROL GOTO” in the Name field of the pop up window.

![Create User Defined Object](image)

5. Click the Add button.

![Add button](image)

6. The UDO should now appear in the User-defined Object Library.

![User-defined Object Library](image)

You have now created a UDO, which can be used throughout the application, as well as in other applications. Now let’s add the UDO to a screen.

7. Create a new screen and rename it Alarms. (We will be using this screen in an upcoming section.)

![Screens](image)

8. Add the MOTOR CONTROL GOTO UDO to the screen by double clicking the UDO. It will be placed in the top most left corner of the screen, move it to the bottom right corner.

There are two ways to add a UDO to a screen, similar to how toolbox objects are added. The first way was just shown above. The second way is to click the UDO, and drag and drop it where you want on the screen.
Exporting a UDO

In order to use a UDO in another application, it must be exported out of the current application. Here, you will learn how to export a UDO, however, you will not be importing it into another application at this time.

1. Highlight the UDO by clicking on it.

2. Select the Export button from the toolbar.

3. Select a destination for the UDO, in this case, choose the Desktop and click Save.

4. Take a look at the desktop, there should now be a file called MOTOR CONTROL GOTO.chu. This is the UDO object. You have successfully exported a UDO to be used in other applications.
Importing a UDO

1. To import the UDO, follow similar steps for exporting. Select the **Import** button from the toolbar.

2. Browse to the **Alarm.chu** file in the folder Lab Files, located on the desktop.

3. Select the UDO and click **Open**.

4. Check that you are on the **Alarms** screen, and add the UDO by double clicking it.
5. You have just created an Alarm screen using User Defined Objects. It should look similar to the screen below.

<table>
<thead>
<tr>
<th>Alarm Message</th>
<th>Occurrence Time</th>
<th>Occurrence Date</th>
<th>Acknowledge Time</th>
<th>Acknowledge Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alarm Message</td>
<td>Occurrence*</td>
<td>Occurrence*</td>
<td>Ack Time</td>
<td>Ack Date</td>
</tr>
</tbody>
</table>

6. Save the application by selecting the **Save** icon ( Quảng cáo ) from the toolbar.
Language Switching

In this section, you will learn about language switching. The language switching functionality makes it possible to create a single application that can be used in multiple countries. For example, this application will be modified so that users can view it in English, German, or Spanish.

1. On the Options Screen, add a Multistate Push Button by double clicking the object from the Toolbox.

2. Double-click the Multi-State Pushbutton to open the States window.

It is best practice to only have as many states as you do languages in your application. Since we have three languages, we will delete the last state.
3. Click row 4, and select the Delete button.

4. Select OK to the prompt.

5. To add the language, the Value must match the language ID number. Add English to the list first.
   - **Value**: 1033
   - **Background Color**: Blue
   - **Caption Text**: English
   - Check the **Caption Font Bold** checkbox
   - Change the **Caption Font Size** to “28”

6. Now add German and Spanish as shown below:

   German:
   - **Value**: 1031
   - **Background Color**: Yellow
   - **Caption Text**: German
   - Check the **Caption Font Bold** checkbox
   - Change the **Caption Font Size** to “28”

   Spanish:
   - **Value**: 2058
   - **Background Color**: Green
   - **Caption Text**: Spanish
   - Check the **Caption Font Bold** checkbox
   - Change the **Caption Font Size** to “28”
7. Click row 1 and Select OK.

8. Edit the properties of the button:
   - Height: 120
   - Left: 500
   - Top: 110
   - Width: 200
   - Indicator Tag: $SysCurrentLang
   - Write Tag: $SysCurrentLang

9. Go to the Settings tab. (Or double click the MotorStarter application in the Project Organizer to open the Settings tab if it is not already open.)

10. Select the Languages icon.

Notice the Startup Application Language is currently set to English. Also notice that all the existing text in your application is shown below in the list.
11. Select the Manage Language List button.

We will add two more languages to this application, German and Spanish.

12. Scroll until you see German (Germany) (1031). Select the language, and then click the >> button to add the language to the terminal. Note that the number following the language is the ID that was used in the States setup.

13. Repeat the above steps in order to add Spanish (Mexico) (2058) to your application.
14. Next, add the translations. Click the drop down arrow to the right of the second column that currently shows English.

<table>
<thead>
<tr>
<th>Text ID</th>
<th>Location</th>
<th>English (United States) (1033)</th>
<th>English (United States) (1033)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Screens</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>2</td>
<td>Screens</td>
<td>Ack</td>
<td>Ack</td>
</tr>
<tr>
<td>3</td>
<td>Screens</td>
<td>Clear</td>
<td>Clear</td>
</tr>
</tbody>
</table>

15. Select German.

16. Notice that the list is empty. Translations have already been completed for you and can be found on the desktop. Go to the desktop Lab Files folder and open the Microsoft Excel file Translations.xlsx.
17. Select the translated words for German. Make sure to select all the words, but do not highlight empty spaces.

<table>
<thead>
<tr>
<th>English</th>
<th>German</th>
<th>Spanish</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
<tr>
<td>Ack</td>
<td>Bestätigen</td>
<td>Reconcier</td>
</tr>
<tr>
<td>Clear</td>
<td>Loschen</td>
<td>Tajo</td>
</tr>
<tr>
<td>Close</td>
<td>Schlieben</td>
<td>Error</td>
</tr>
<tr>
<td>START MOTOR</td>
<td>Starten Motor</td>
<td>Comenzar motor</td>
</tr>
<tr>
<td>MOTOR STARTED</td>
<td>Motor gestartet</td>
<td>Motor comenzado</td>
</tr>
<tr>
<td>Error</td>
<td>Fehler</td>
<td>Error</td>
</tr>
<tr>
<td>STOP MOTOR</td>
<td>Motor Stoppen</td>
<td>Motor Stoppen</td>
</tr>
<tr>
<td>MOTOR STOPPED</td>
<td>Motor Gestoppt</td>
<td>Motor Detenido</td>
</tr>
<tr>
<td>Error</td>
<td>Fehler</td>
<td>Error</td>
</tr>
<tr>
<td>MOTOR SPEED</td>
<td>Motor Geschwindigkeit</td>
<td>Motor Velocidad</td>
</tr>
<tr>
<td>CONFIG SCREEN</td>
<td>Konfig Bildschirm</td>
<td>Konfig exhibir</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>Automatisch</td>
<td>Automatico</td>
</tr>
<tr>
<td>MANUAL</td>
<td>Manual</td>
<td>Manual</td>
</tr>
<tr>
<td>SELECT MODE</td>
<td>Wählen Modus</td>
<td>Seleccionar Modo</td>
</tr>
<tr>
<td>MANUAL</td>
<td>Manual</td>
<td>Manual</td>
</tr>
</tbody>
</table>

18. Copy the entire selection by pressing **Ctl + C** on your keyboard.

19. Go to CCW and select all the empty boxes for the corresponding words in the German column. Paste the translated words by pressing **Ctl+V** on your keyboard.
20. Your screen should look similar when complete:

![Image of screen with text columns for English and German]

21. Change the column heading from German to Spanish using the drop down arrow and repeat the above steps to add the translations for Spanish.

There are words that are repeated in the list. Every word and phrase in the application is added to the Language list; therefore, each word and phrase must have its own translation. If it does not have a translation, it will show up blank when the translation is made.

It is suggested to set up languages last when building an application. Since it pulls words into the application in the order they were added, anything added to the application after translations have been made will need to be translated again.

22. Double check the Startup Application Language. Select English if it is not already.

23. Validate the application by selecting Validate from the menu bar on the Settings tab. Correct any warnings if they exist.

24. Save the application by selecting the Save icon ( ) from the toolbar.

25. Click on Download icon from the menu bar.
26. **Save** any changes made if prompted to do so.

27. The Connection Browser will pop up.

![Connection Browser](image1)

28. Browse for the terminal by expanding **AB_ETHIP-1, Ethernet**, selecting the **2711R-T7T** and selecting **OK**.

![Connection Browser](image2)

29. If prompted to stop the loaded application, select **Yes** and repeat the download (previous four steps).

![Confirm](image3)
30. Select Yes to overwrite an existing application with the same name.

![Confirm Window]

31. The Output window will show the progress of the download. Ensure the application downloads successfully before moving forward.

```
Connecting to graphic terminal... Succeeded
Preparing device for download. This may take some time to execute... Succeeded
Downloading application.............
Application downloaded successfully
```

32. Load the application following the steps from the Validate and Transfer the Application to the Terminal section (pages 77-78).

33. Click the MORE… button to get to the Options display.

34. Click the English button once so that it switches to German, and then go back to the MOTOR CONTROL display. Everything should be in German.

35. Click the MORE… (Mehr) button again to go back to the Options display.

36. Click the German button once more to go to Spanish, or two times to return to English. Go back to the MOTOR CONTROL display to see the translation.

37. Once you are done exploring, select English as the language and return to the Config Screen (MOTOR CONTROL → CONFIG SCREEN) on the terminal.
38. Go back to the **Settings** tab and select **Languages**.

39. Click on the **Manage Language List** button.
40. Under **Languages on the Terminal**, select **German** and then click the << button to remove the language from the terminal.

![Manage Language List](image)

41. Repeat the previous step for **Spanish** and then click the **OK** button.

42. If a **Confirm** window opens up, click the **OK** button.

You have now completed the language switching section of the lab.
Learn how to Use Alarms

Making unexpected system or process events visible to operators is an important part of many HMI applications. In this section, you will learn about alarms and how to use them.

1. Double click the **Alarms** icon under **MotorStarter**.

2. Explore the options on the Alarm page. Take note of the **Clear Alarm History when Application is Loaded**, and the **Alarm History Size**.

3. Click the **Add Alarm** button to add a new alarm to the application.

   For simplicity, you will create an alarm that notifies the user any time the motor is started.

4. Click the drop down arrow under **Trigger**, and select **Motor_Ind**.
5. Verify that **Bit** is selected for the **Alarm Type**.

6. Use the scroll bar and scroll to the right to view the rest of the options for the alarm. Type “**Motor Started**” in the **Message** box and press **Enter**. Leave the rest of the settings at the default.

You have just configured an alarm that will notify when the motor has been started.

In order to navigate to the Alarms screen, let’s add a GoTo Screen button on the Options display.

7. Open the **Options** display by double clicking on it.

8. From the Toolbox, add a **Goto Screen** button by double clicking it.

9. Edit the following properties:
   - **Background Color**: Dark grey (#484850)
   - **Border Width**: 10
   - **Font Bold**: True
   - **Font Size**: 28
   - **Text**: ALARMS
   - **Text Color**: White (#F8F8F8)
   - **Height**: 120
   - **Left**: 0
   - **Top**: 360
   - **Width**: 200
   - **Screen**: 3- Alarms
10. The screen should look similar to the one shown below.

11. Validate selecting **Validate** from the menu on the Settings tab.

12. Save the application by selecting the **Save** icon ( ) from the toolbar.

13. Right-click on the **MotorStarter** icon and select **Download**.

14. The Connection Browser will pop up.
15. Browse for the terminal by expanding **AB_ETHIP-1, Ethernet**, selecting the **2711R-T7T** and selecting **OK**.

![Connection Browser](image)

16. If prompted to stop the loaded application, select **Yes** and repeat the download (previous three steps).

![Confirm](image)

17. Select **Yes** to overwrite an existing application with the same name.

![Confirm](image)
18. The **Output** window will show the progress of the download. Ensure the application downloads successfully before moving forward.

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show output from:</td>
</tr>
<tr>
<td>Connecting to graphic terminal... Succeeded</td>
</tr>
<tr>
<td>Preparing device for download. This may take some time to execute... Succeeded</td>
</tr>
<tr>
<td>Downloading application.............</td>
</tr>
<tr>
<td>Application downloaded successfully</td>
</tr>
</tbody>
</table>

19. Load the application following the steps from the *Validate and Transfer the Application to the Terminal* section (pages 77-78).

20. After clicking the **START MOTOR** button, you should notice a pop up banner notifying you of an alarm. Click the **Close** button.

21. Click the **STOP MOTOR** button.

22. Navigate to the Alarm screen (MORE… → ALARMS)

![ALARMS](image)

23. Use the Enter arrow button to acknowledge the alarm. Once the alarm is acknowledged, the table will display the time and date of the acknowledgement.

24. Return to the MOTOR CONTROL screen, start the motor, and this time click on the **Ack** button.

25. Stop the motor, navigate to the Alarm screen and notice for the latest alarm that the Acknowledge time and date have already been recorded.

26. Once you are done exploring alarming, return to the **Config Screen** (MOTOR CONTROL → CONFIG SCREEN) on the terminal.

You have now completed the alarm section of the lab.
Learn how to Create a Trend and Configure a Data Log

In this section you will learn about Trends and Data Logs. In some applications, it is important to monitor specific data as well as have a historical record of the data. Trending and data logs provide such functionality.

Create a Trend Object

1. First, create the tag that will be used in the Trend. Go to the Tag Editor and select Add from the External tab.
2. Click the ellipsis button in the Address field. Make sure you are on the I/O – Micro850 tab, then select _IO_P2_AI_00.
3. Assign “Speed_command_potentiometer” for the Alias, then select OK.

4. Rename the Tag Name as “Speed_command_potentiometer” as well.
5. Select PLC-1 as the Controller.

6. Create a new screen by right-clicking on Screens → Add Screen
7. Rename the new screen “Trend” by right-clicking on it and selecting **Rename**.

8. From the toolbox, add a **Trend** object by doubling clicking it.

9. Each application is allowed up to 5 trends. If more than 5 trends are added, the application will error when the validation is run.

10. Edit the following properties:

   - **Height**: 360
   - **Left**: 0
   - **Top**: 0
   - **Width**: 800
   - **Maximum Value**: 65535
11. Double click the trend object to open the **Pens** page. This is where pens are added, edited, and removed on the trend. Each trend can have up to 6 pens.

12. Select the drop down arrow under **Read Tag**, and select the tag you created, *Speed_command_potentiometer*. 
13. Click in the **Appearance Line Color** box, and select the ... button. Choose light blue as the color and select **OK**.

14. The **Pens** page should look similar when complete.

15. Select **OK** to close the **Pens** page.

There are a few Properties that must be configured for the Trend object.

16. Open the **Properties** window if it is not already open.
17. Click in the box next to **Autoscroll Configuration**, and then click the drop down arrow. There are two options, select **Remove One Then Add** if it is not already selected. This will keep a continuous line on the trend object. (This will also keep continuous data in the Data Log file.)

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Trend</strong> Trend_1</td>
</tr>
<tr>
<td><strong>Autoscroll Configuration</strong> Remove One Then Add</td>
</tr>
<tr>
<td>Label Color</td>
</tr>
<tr>
<td>Label Font Bold</td>
</tr>
<tr>
<td>Label Font Height</td>
</tr>
<tr>
<td>Label Font Italic</td>
</tr>
</tbody>
</table>

**Note:** Selecting **Remove One Then Add** will continue scrolling at the end of the time span. When a new event is added, the oldest event value is removed from the viewing area. Selecting **Remove All Then Add** will restart scrolling at the end of the time span. All events are removed from the chart and plotting will restart at the beginning of the time span.

18. Click in the box next to **Update While Off Screen**, and then click the drop down arrow. Select **True**. This will allow the trend to update continuously, even though the screen may not be displayed on the terminal. (This will also keep continuous data in the Data Log file.)

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of Y-axis Labels</strong> 3</td>
</tr>
<tr>
<td><strong>Pens</strong> (Collection)</td>
</tr>
<tr>
<td><strong>Reference Line Color</strong> #F8F8F8</td>
</tr>
<tr>
<td><strong>Sample Interval</strong> 1</td>
</tr>
<tr>
<td><strong>Sample Interval Unit</strong> Seconds</td>
</tr>
<tr>
<td><strong>Timespan</strong> 1</td>
</tr>
<tr>
<td><strong>Timespan Unit</strong> Minutes</td>
</tr>
<tr>
<td><strong>Update While Off Screen</strong> True</td>
</tr>
</tbody>
</table>

**Note:** Selecting **False** will stop the trend from plotting when a screen change occurs.
**Configure a Data Log**

Data logging allows tag values to be recorded and saved for further analysis. It is possible to store a data log on a USB drive or MicroSD card. The user can start and stop the data log in order to transfer the file to a PC, and then resume logging. The data log file is stored as Trendlog.csv.

In order to create a Data Log, there must be a trend object in the application. Only one data log file can be created per application. If the application has multiple trend objects, the trend that has the Log File Location specified in the Properties will be represented in the Data Log. It is not possible to set the Log File Location for more than one trend object at a time.

Since a Trend object has already been created, let’s configure the rest of the Properties to create a Data Log.

1. Click in the box to the right of **Log File Location**, and then click the drop down arrow.

2. There are three options: None, USB Storage or MicroSD Storage. For this lab, select **USB Storage**.

3. Click in the box next to **Log Trigger Tag**, and then click the drop down arrow. Select the **Start_Motor** tag. This tag must be set in order to start a Data Log file and it must be a Boolean tag.
4. Add the Motor Control UDO to the display. *(View → User Defined Object Library → double click MOTOR CONTROL GOTO)* Place the button in the lower right corner. This will allow navigation to the Motor Control screen.

In order to navigate to the Trend screen, let’s add a GoTo Screen button on the Options display.

5. Open the Options display by doubling clicking on it.

6. Right click on the Alarms button and select Copy. Then click anywhere on the screen, right-click, and select Paste.

Doing this minimizes the amount of changes when creating a new button, especially if keeping a uniform look within the application.

7. Edit the following properties:
   - **Text**: TREND
   - **Left**: 200
   - **Top**: 360
   - **Screen**: 4- Trend
8. The Options screen should look similar to the one shown below.

![Options screen](image)

9. Validate the application by selecting **Validate** from the menu bar on the Settings tab. Correct any warnings if they exist.

![Validate button](image)

10. Save the application by selecting the **Save** icon ( ) from the toolbar.

11. Click on **Download** icon from the menu bar.

12. Save any changes made if prompted to do so.

13. The Connection Browser will pop up.

![Connection Browser](image)
14. Browse for the terminal by expanding **AB_ETHIP-1, Ethernet**, selecting the **2711R-T7T** and selecting **OK**.

![Connection Browser](image)

15. If prompted to stop the loaded application, select **Yes** and repeat the download (previous four steps).

![Confirm](image)

16. Select **Yes** to overwrite an existing application with the same name.

![Confirm](image)

17. The **Output** window will show the progress of the download. Ensure the application downloads successfully before moving forward.

18. Load the application following the steps from the **Validate and Transfer the Application to the Terminal** section (pages 77-78).

19. From the **MOTOR_CONTROL** screen, select **START MOTOR**.

20. Navigate the **Trend** display. (MORE... → **TREND**)

21. Turn the **SPEED COMMAND** potentiometer back and forth on the demo box to see changes in the trend. The trend is showing the changes at the analog input. The value will range from 0 to 65535.
22. Notice that the trend line is constantly moving on the screen. This is because you chose “Remove One Then Add” for the Autoscroll Configuration.

23. If there were a USB device attached, the terminal would create and store a Trend Log file on it.

24. Once you are done exploring trending, return to the Config Screen (MOTOR CONTROL ⇒ CONFIG SCREEN) on the terminal.

You have now completed the trend and data logging section of the lab.
Learn how to Use Recipes

In this section you will learn the basics of building multiple recipes and how they are typically used in an application.

Building a Recipe

1. In the Project Organizer, double click on the Recipes icon located under MotorStarter.

2. This will open a separate tab where everything needed to create a recipe is located. Take a minute to explore this page.

3. Let's start building a recipe. Click on Create Recipe.

4. Notice this adds a new recipe called RECIPE_01 under the Recipe list.

5. Click on RECIPE_01 and notice two icons now appear, a gear and a clone option. Click on the gear to bring up the recipe properties window.

6. Let's rename it. Type "NORMAL_SPEED" in the Name box as well as in the File Name box.

Recipe names must not contain spaces or symbols.
7. For now, leave the Status Tag blank and select OK.

8. Since we will be controlling the motor speed with this recipe, let’s add the speed as an ingredient. Notice the first ingredient is already in the list. Let’s rename it by double clicking in the box and typing “SPEED_COMMAND”. Press Enter to keep the change.

9. Enter the following values:
   - Tag: Speed_Command
   - Data Type: Numeric
   - Value: 30

   ![Recipe Properties](image)

<table>
<thead>
<tr>
<th>Name:</th>
<th>Tag</th>
<th>Data Type</th>
<th>Min Value</th>
<th>Max Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPEED_COMMAND</td>
<td>Speed_Command</td>
<td>Numeric</td>
<td></td>
<td></td>
<td>30</td>
</tr>
</tbody>
</table>

   Now that we have the recipe configured for the NORMAL_SPEED, let’s create the HIGH_SPEED recipe. We will clone the NORMAL_SPEED recipe since most of the items are the same.

10. Click on the clone icon that appears once you click on the NORMAL_SPEED recipe.

11. Click on the new recipe and notice all of the ingredients you created are duplicated.

   ![Recipe List](image)
12. Rename the new recipe HIGH_SPEED. Leave the Status Tag blank for now and click **OK**.

![Recipe Properties](image)

13. Since this recipe requires a high speed, change the **Value** to “60”.

14. Save the application by selecting the **Save** icon ( ![Save icon](image) ) from the toolbar.

You have now created two recipes, NORMAL_SPEED and HIGH_SPEED. Feel free to explore more options on the recipes page at this time. In the next section, you will learn how to build a recipes display.
Building a Recipe Display

1. Create a new screen by right-clicking on Screens ➔ Add Screen

2. Rename the new screen “Recipe” by right-clicking on it and selecting Rename.

In the essence of time, the display has already been put together and saved as a UDO.

3. Open the User-Defined Object Library. (View ➔ User-Defined Object Library)

4. In order to use the UDO, it must first be imported into the application. Click the Import button and browse to Desktop ➔ Lab Files ➔ Recipe.chu and select Open. It will appear in the UDO Library.

5. Double-click the Recipe icon.
6. The screen will look similar to the one shown below. Notice this UDO included the Motor Control GoTo Screen button already.

7. Click on the **Recipe Table** to explore the properties.
8. Click on the **Recipe Selector** to explore the properties.

9. Click on each action button to explore the properties.

10. Click on one of the arrow buttons to explore the properties.

    Notice the option to change the Key Type in the Navigation section. This is important to note since the Toolbox only provides the option to add a Key to the display.

    In order to navigate to the Recipe screen, let's add a Goto Screen button on the Options display.

11. Open the **Options** display by double clicking on it.

12. Right click on the **Alarms** button and select **Copy**. Then click anywhere on the screen, right-click, and select **Paste**.

    Doing this minimizes the amount of changes when creating a new button, especially if keeping a uniform look within the application.
13. Edit the following properties:
   - **Text**: RECIPE
   - **Left**: 400
   - **Top**: 360
   - **Screen**: 5-Recipe

14. The screen should look similar to the one shown below.

15. Validate the application by selecting **Validate** from the menu bar on the Settings tab.

16. Save the application by selecting the **Save** icon ( ) from the toolbar.

17. Click on **Download** icon from the menu bar.

18. Save any changes made if prompted to do so.
19. The Connection Browser will pop up.

20. Browse for the terminal by expanding **AB_ETHIP-1, Ethernet**, selecting the **2711R-T7T** and selecting **OK**.

21. If prompted to stop the loaded application, select **Yes** and repeat the download (previous four steps).
22. Select Yes to overwrite an existing application with the same name.

23. The Output window will show the progress of the download. Ensure the application downloads successfully before moving forward.

24. Load the application following the steps from the Validate and Transfer the Application to the Terminal section (pages 77-78).

25. Press the MORE... button to get to the Options display.

26. Press the RECIPE button to get to the Recipe display.

27. Under the Recipe Selector box, use the up and down arrows to point to either the NORMAL_SPEED or the HIGH_SPEED recipe, and then press the Enter arrow to select that recipe.

28. Now that one of the recipes is highlighted in the Recipe Selector box, press the Restore button to display the recipe “ingredients”.

29. Use the DI0 switch on the demo box to start the motor and observe the speed on the drive display.

30. Press the Download button and observe the speed change (if any).

31. Repeat the previous four steps for the other recipe and observe the speed change.

32. Once you are done exploring, use the DI1 switch on the demo box to stop the motor and return to the Config Screen (MOTOR CONTROL → CONFIG SCREEN) on the terminal.

You have now completed the recipes section of the lab.
Learn how to Add Security to the Terminal

Most applications are created for multiple users, however you may want to restrict access for some users. In this section, you will learn about security and how to use it.

1. Go to the **Settings** Tab.
2. Click on **User Accounts** in the menu list.

3. Take a minute to explore the Security options on this page.

4. Add a user by clicking the **Add User** button.
5. In the pop up display, fill in the following:
   - **User Name**: Engineer
   - **Password**: pvc
   - **Confirm Password**: pvc
   - **Password Modifiable**: Yes

6. Then select **Add**

7. Notice once the user is added, the **Design** right is automatically checked, as well as the **Password – Modifiable**, since 'Yes' was selected in the pop up window.

8. Add another user and assign the following:
   - **User Name**: Operator
   - **Password**: 123
   - **Confirm Password**: 123
   - **Password Modifiable**: Yes

9. Then select **Add**
10. Notice the **Operator** is not given **Design** rights automatically.

<table>
<thead>
<tr>
<th>User</th>
<th>Password - Reset</th>
<th>Password - Modifiable</th>
<th>DESIGN</th>
<th>TREND</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Users*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENGINEER</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>OPERATOR</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Next, let’s assign the Engineer access to all screens, but the Operator only to MOTOR_CONTROL, Options, and Alarms.

11. Click the **Add Right** button

12. Click on the ellipsis button next to **RIGHT1**.

13. This provides a text box. Type “**Trend**” in the box and press Enter.

14. Assign the **Engineer** rights to view the **Trend** screen by checking the box in the Trend column.

Since we are preventing the Operator from viewing the Trend screen, we need to apply security to the trend screen.

15. Open the **Trend** screen by double clicking on it.

16. Open the **Properties** for the **Trend** screen.
17. In the **Screen** section, click in the box next to **Access Rights**, and select **Trend**.

![Properties](image)

- **Background Color**: #F8F8F8
- **Focus Cursor Color**: #999999
- **Focus Cursor Thickness**: 4

- **Common**
  - **Description**: Trend
  - **Name**: Trend

![Access Rights](image)

- **Horizontal Grid Spacing**: 10
- **Screen Number**: 4
- **Vertical Grid Spacing**: 10

You have successfully secured the Trend screen.

Now that you have set up the security, users need to be able to log in and out of the application.

18. Open the **Options** screen by double clicking on it.

![Options](image)

19. From the Advanced section of the Toolbox, add a **Login** button by double clicking on it.

![Toolbox](image)

20. Edit the following properties for the button:

- **Height**: 60
- **Left**: 160
- **Top**: 55
- **Width**: 150
- **Font Size**: 28
21. From the Toolbox, add a *Logout* button.

22. Edit the following properties for the button:
   - **Height**: 60
   - **Left**: 160
   - **Top**: 140
   - **Width**: 150
   - **Font Size**: 28

23. From the Toolbox, add a *Change Password* button.

24. Edit the following properties for the button:
   - **Height**: 60
   - **Left**: 110
   - **Top**: 220
   - **Width**: 250
   - **Font Size**: 28

With the Change Password button, the users will only be able to change their password if the Password Modifiable box is checked when adding the user.
25. Your screen should look similar to the one shown below:

![Login, Logout, Change Password, English](image)

You have added all the buttons required for the users to log in and out of the application, as well as change their own password. Take note that you did not add a Reset Password button to the application. TheReset Password button allows the user to reset a password without the use of an old password. This button requires a current user name and a new password only. It should be used on a secured screen to prevent password changes by unauthorized users. A warning is also generated during validation if this button is placed on an unsecured screen.

26. Validate the application by selecting **Validate** from the menu bar on the Settings tab.

![Validate](image)

27. Save the application by selecting the **Save** icon ( ) from the toolbar.

28. Click on **Download** icon from the menu bar.

29. **Save** any changes made if prompted to do so.

30. The Connection Browser will pop up.

![Connection Browser](image)
31. Browse for the terminal by expanding **AB_ETHIP-1, Ethernet**, selecting the **2711R-T77** and selecting **OK**.

![Connection Browser](image)

32. If prompted to stop the loaded application, select **Yes** and repeat the download (previous four steps).

![Confirm](image)

33. Select **Yes** to overwrite an existing application with the same name.

![Confirm](image)
34. The **Output** window will show the progress of the download. Ensure the application downloads successfully before moving forward.

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show output from: Graphic Terminal</td>
</tr>
<tr>
<td>Connecting to graphic terminal... Succeeded</td>
</tr>
<tr>
<td>Preparing device for download. This may take some time to execute... Succeeded</td>
</tr>
<tr>
<td>Downloading application...............</td>
</tr>
<tr>
<td>Application downloaded successfully</td>
</tr>
</tbody>
</table>

35. Load the application following the steps from the *Validate and Transfer the Application to the Terminal* section (pages 77-78).

36. To test the security, select the **MORE...** button on the terminal to go to the **Options** screen.

37. Select the **Login** button

   ![Login Button](image)

   For **User**, type in **“Engineer”**
   For **Password**, type in **“pvc”**

38. Notice you can navigate to the Alarm, Trend, and Recipe screens without issue.

39. Now select the **Logout** button.

40. Log in as the **Operator**

   **User:** **Operator**
   **Password:** **123**

41. Notice you can navigate to all screens except the Trend screen. When you try to navigate to the Trend screen, the terminal prompts you to login as an approved user.

42. Login as the **Engineer** again.
43. Select the **Change Password** button

![Change Password button]

Enter the following:

- Old Password: “**pvc**”
- New Password: “**800**”
- Confirm Password: “**800**”

44. Select **Enter**

45. Logout and then try logging in as the **Engineer** with the old password (**pvc**) - it shouldn’t work. Now try logging in with the new password (**800**). You should successfully log in.

46. Once you are done exploring, return to the **Config Screen** (MOTOR CONTROL → CONFIG SCREEN) on the terminal.

You have now completed the security section of the lab.

Congratulations! You have now completed the PanelView 800 section of the lab!