TELVENT

config@WEB Relay Ladder Logic Manual

S2200-AAA-00003 V6.0

Telvent
10333 Southport Dr. S.W.
Calgary, AB, Canada T2W 3X6

Phone: +1 (403)253.8848
Fax: +1 (403)259.2926
E-mail: info.canada@telvent.abengoa.com

Telvent
7000A Hollister Rd.
Houston, TX 77040-5337

Phone: +1 (713)939.9399
Fax: +1 (713)939.0393
E-mail: info.usa@telvent.abengoa.com

config@WEB Relay Ladder Logic Manual

For Reference Only

© Copyright 2008 by Telvent USA, Inc.

The information contained in this document is confidential and proprietary to Telvent USA, Inc. It is not to be copied or disclosed for any purpose except as specifically authorized in writing by Telvent. Although the information contained herein was correct and verified at the time of publication, it is subject to change without notice.

Manual No. **S2200-AAA-00003**

Document Approval

Rev	Date	Description	ECO #	Review Approved
0.0	08-04-03	Original release	N/A	
1.0	09-26-03	Updated & expanded	11316	
2.0	01-07-04	Updated with example chapter	11366	
3.0	08-03-04	Numerous examples added to example chapter	11485	
4.0	03-23-05	Corrected number of points supported	11528	
5.0	08-06-07	Updated for CA_P2 firmware	11674	
6.0	10-08-08	Updated for D0 firmware	11748	Dan Stark 16-OCT-2008

Dan Stark, Senior
Technical Specialist,
RTU S/W Engineering

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Overview

1.1.1 Features

- Windows 2000, XP development environment
- IEC 61131-3 certification (six languages, LD, FBD, ST, IL, SFC, FC)
- Develops powerful applications without requiring the programmer to know complex high-level computer languages
- Debugging tools provide:
    - On-line monitoring
    - On-line changing of variables
    - Off-line simulation

1.1.2 Application

ISaGRAF PRO is a program for Telvent config@WEB line of RTUs that supports IEC 61131-3 programming languages. IEC 61131-3 (sometimes shortened to IEC 1131-3) was developed by the International Electro-technical Commission as a way to standardize industrial automation. The languages supported are:

- Sequential Function Block (SFB) graphical language
- Function Block Diagram (FBD) graphical language
- Flow Chart (FC) graphical language
- Ladder Diagram (LD) graphical language
- Instruction List (IL) language
- Structured Text language

Notice that four of the languages supported are graphical.

The ISaGRAF PRO Programming Software adds logic functions to the config@WEB line of RTUs. This software must be installed on any PC being used to create/modify/debug logic written for the RTU.

The Telvent RTU firmware interfaces with the ISaGRAF PRO software via TCP/IP. Additional software provided by Telvent must be added to the ISaGRAF PRO directory to facilitate the connection between the PC and the RTU.

Telvent sells and supports two versions of ISaGRAF PRO as follows:

- S2200-RLL-001XX 256 Point Relay Ladder Logic Kit
- S2200-RLL-002XX Unlimited Point Relay Ladder Logic Kit

1.2 Points Supported

The firmware supports inputs or outputs of the following types. Each driver supports the number of points shown below.

- Analog Inputs (read/write) – 256
- Analog Outputs (read/write) – 256
- Binary Inputs (read, 256 / write, 256)
- Binary Input Momentary Change Detect (read) – 256
- Binary Output (read/write) – 256
- Counter (read/write) – 256
- Select Before Operate Control (read/write) – 256

A driver is provided to read/write each of these data types. Unused point types may have their drivers deleted from the system. Deleted drivers may be restored to the system at a later time. The system is shipped with all drivers installed configured for 6 points for each driver.

- Points may be added to the system for writing outputs from the logic.
- Any point active in the RTU may be mapped as an input to the logic.
- Any point active in the RTU may be mapped as an output from the logic.

1.2.1 Multiple Programs in the RTU

ISaGRAF for config@WEB can support multiple programs running simultaneously in the RTU. This gives the user great freedom and great responsibility. If you create and download multiple programs, be sure they either run in harmony with each other, or do not interfere with each other. If you wish to clear the RTU of a particular program, delete the particular program in ISaGRAF Workbench (that is, on your PC), then compile and download the new project (without the offending program). If you wish to clear the RTU of all programs, create an empty project, compile it, and download it.

1.3 Reference Documents

You will find the following documents useful in the development and operation of RLL programs for Telvent RTUs.

Table 1-1 Reference Documents

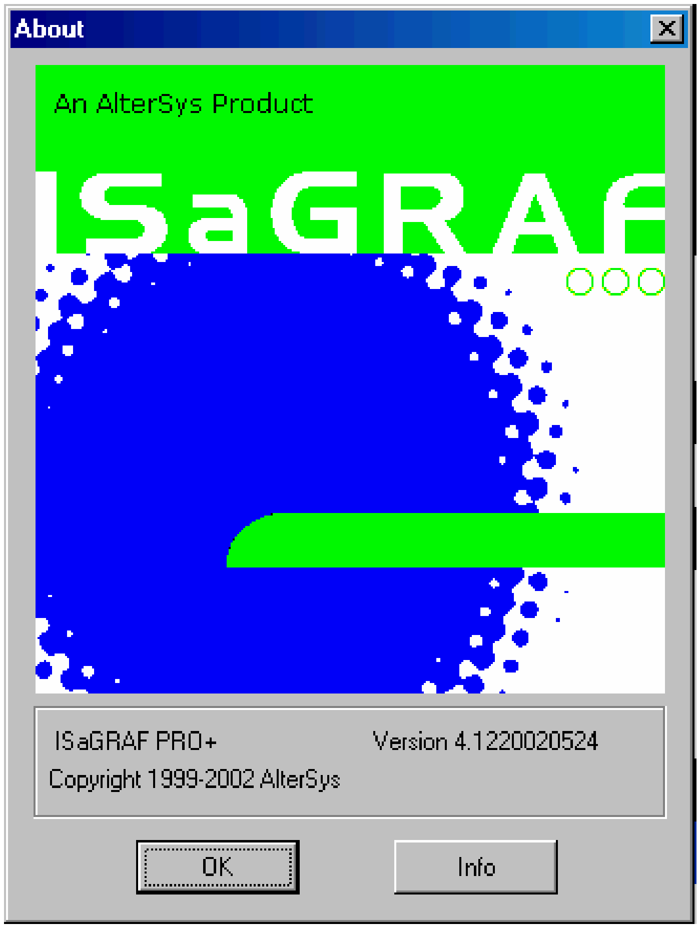
Document Name	Author	Publisher
SAGE 2200 Operation & Maintenance Manual	N/A	Telvent (included on Telvent Installation CD)
config@WEB Protocols Manual	N/A	Telvent (included on Telvent Installation CD)
Programming Industrial Control Systems Using IEC 1131-3	R.W. Lewis	The Institution of Electrical Engineers
IEC 1131-3 Programming Methodology	Flavio Bonfatti, Paola Daniela Monari, Umberto Sampieri	ISaGRAF
ISaGRAF PRO Workbench	N/A	ISaGRAF (included on Telvent Installation CD)
ISaGRAF PRO Getting Started	N/A	ISaGRAF (included on Telvent Installation CD)

1.4 How to Determine Your Number-Of-Points Supported

Version 4.12 (128 or unlimited points) had different limits from version 4.20 (256 or unlimited points).

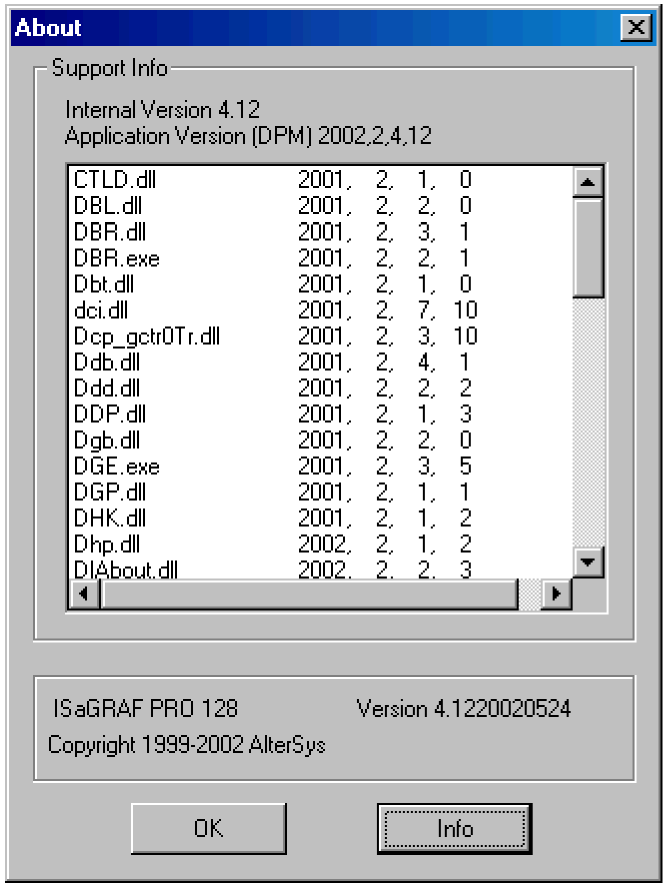
With the unlimited point count dongle, the help screen displays the following:

Figure 1-1 Help Screen for Version 4.12 Using Unlimited Dongle



Using the same ISaGRAF program limited point count dongle (128), the help screen changes the name from ISaGRAF PRO+ to ISaGRAF PRO 128 as shown below.

Figure 1-2 Help Screen for Version 4.12 Using 128 Dongle



You can also find your point count for your particular program and dongle using the I/O wiring tool to determine the number of possible points.

Start ISaGRAF and create a new program. Go to the I/O wiring tool. There should be 13 devices, each with 6 points (the default configuration for a new program). Change the number of points on one of the devices to 56. This should work. Then change the number of points to 57. This should fail if it is limited to 128 points.

CHAPTER 2

Installation

2.1 RTU Requirements

You must have a Telvent config@WEB RTU and the RTU firmware must be A8 or higher.

Although not required for installation of RLL, having a working config@WEB interface connection over Ethernet is required for downloading and testing of RLL programs.

2.2 PC Requirements

Your PC must be running one of these operating systems: Windows 98SE, 2000, NT, or XP.

2.3 Installation Package Contents

The installation package consists of:

1. ISaGRAF Workbench Installation CD
2. ISaGRAF PRO Installation Dongle for parallel or USB port
3. Telvent Installation CD (includes Telvent manuals & ISaGRAF PRO Workbench manual & ISaGRAF PRO “Getting Started”).
4. Telvent config@WEB Relay Ladder Logic Manual (this manual)

2.4 Installation Procedure

2.4.1 Step 1: Install Sentinel Driver

The driver for the dongle must be installed before other elements of the ISaGRAF package. Follow the steps below.

1. Make sure that the latest driver is installed. Go to the "Sentinel" directory on the installation CD and launch the executable
2. Once installed, go to C:\Program Files\Rainbow Technologies\Sentinel System Driver (or wherever you chose to install the driver) and launch SetupSysDriver.exe
3. Click on "Configure Driver"
4. Make sure there is a USB port listed here. If not, click on "Add", choose "USB" as Bus Type and "USB" as Port Type

By adding the USB port to the list, the software should now be able to detect USB keys.

2.4.2 Step 2: Install ISaGRAF PRO

5. Insert the ISaGRAF PRO Installation CD into your PC's CD reader. Follow ISaGRAF's instructions for installation (see ISaGRAF PRO "Getting Started" included on the Telvent Installation CD).

Note: Please be aware that there are two different license options. One allows limited I/O points while the other allows unlimited I/O points.

6. Check ISaGRAF Workbench installation CD (using Windows Explorer) for an updated Sentinel driver. If there is one on the CD it should be under a directory with a name like "Sentinel Driver". Run the program under that directory to install the latest driver. Alternatively you can visit the rainbow.com website and download the latest driver.

Note: We recommend updating the Sentinel driver to the latest available from the Rainbow.com website to ensure proper operation of ISaGRAF software. If you get any warning messages during installation or operation it is probably due to old Sentinel drivers.

7. Install the dongle. This should be done AFTER ISaGRAF and Sentinel driver installation.

Note: There are two types of dongle: Parallel-port dongle and USB dongle. The parallel-port dongle has a feed-through connector to accommodate other parallel port devices. The USB dongle is a USB termination.

2.4.3 Step 3: Install Telvent-Provided Components

Note: ISaGRAF v4.12 has a somewhat different file structure than v4.20 and later. The directions below differ according to the version of ISaGRAF.

2.4.3.1 ISaGRAF V4.12:

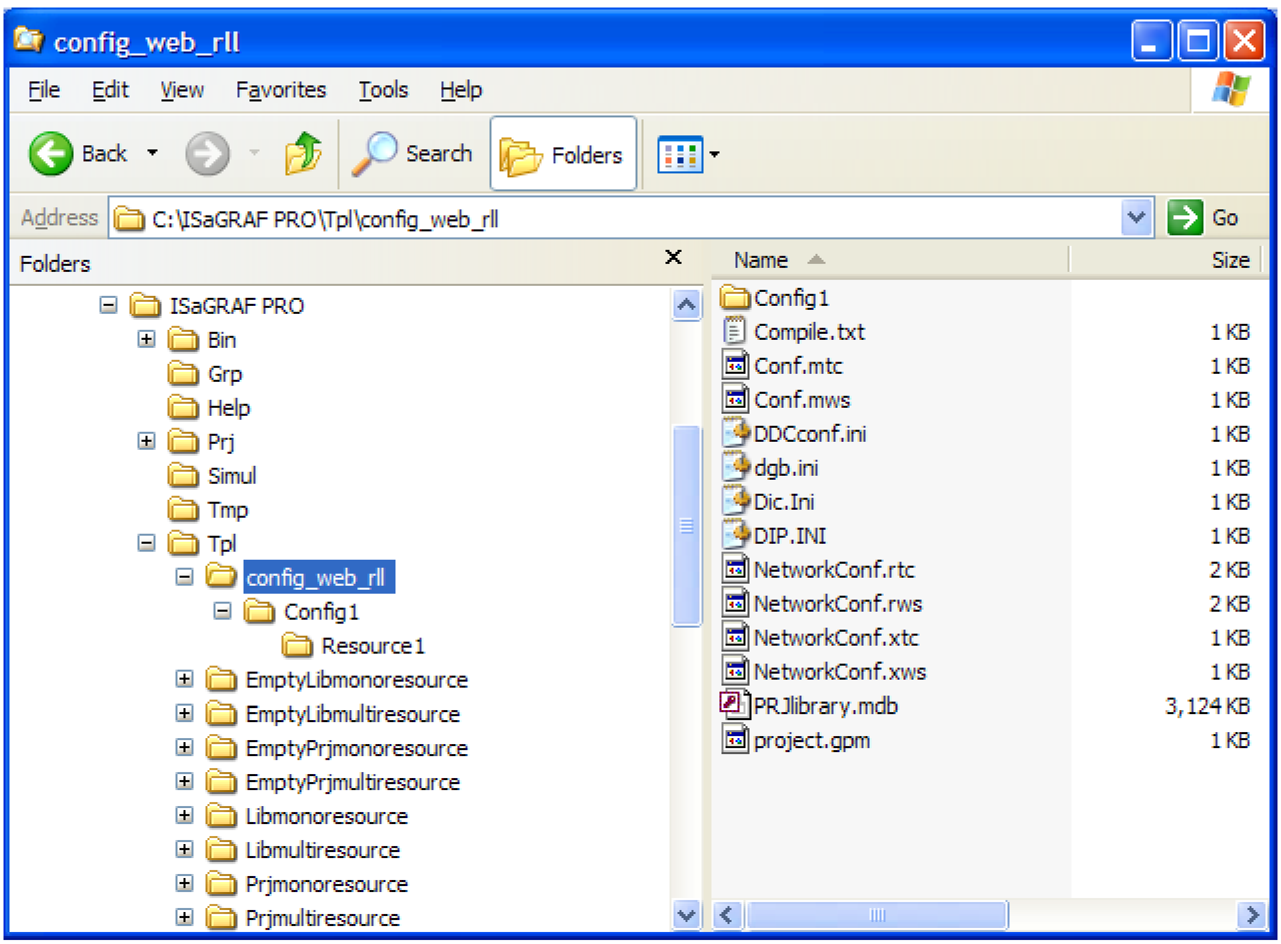
Find the directory “c:\ISaGRAF PRO\Tpl”. From the Telvent Installation CD, extract the file “config_web_rll.zip” into the “c:\ISaGRAF PRO\Tpl” directory.

Note: Be sure to use the Extract command in WinZIP. Make sure “Use folder names” is checked in the Extract dialog box. Do not select all and drag the files into the ISaGRAF PRO\Tpl folder.

This config_web_rll folder will serve as the template for all other programs developed using ISaGRAF PRO.

The config_web_rll folders in Figure 2-1 (for V4.12) have been expanded to show what you should have after extracting the zip files.

Figure 2-1 Directory Structure for V4.12 After Installation of the Telvent-Provided Zip File



2.4.3.2 ISaGRAF V4.20 (and later):

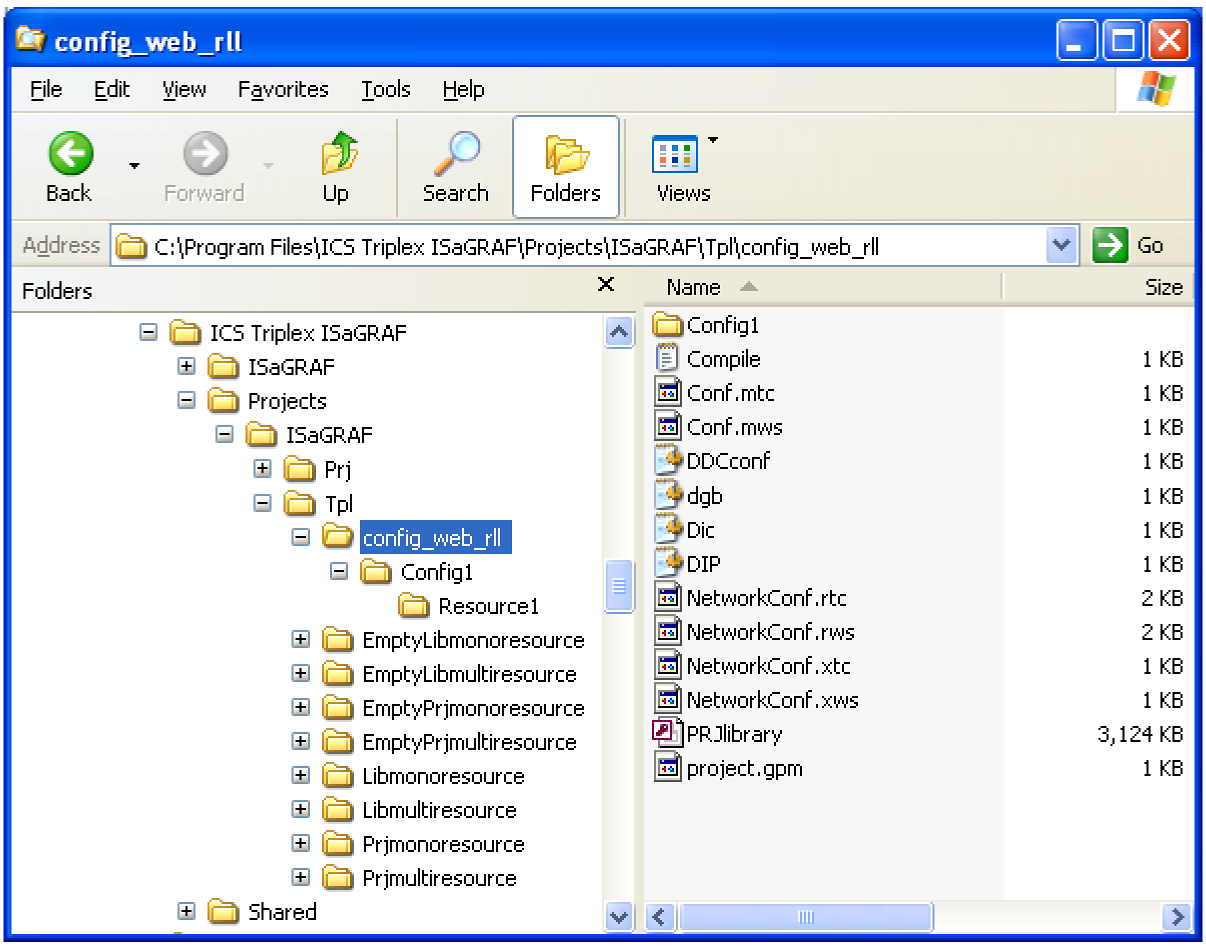
Find the directory "C:\Program Files\ICS Triplex ISaGRAF\Projects\ISaGRAF\Tpl". From the Telvent Installation CD, extract the file "config_web_rll.zip" into the "C:\Program Files\ICS Triplex ISaGRAF\Projects\ISaGRAF\Tpl" directory.

Note: Be sure to use the Extract command in WinZIP. Make sure "Use folder names" is checked in the Extract dialog box. Do not select all and drag the files into the ISaGRAF PRO\Tpl folder.

This config_web_rll folder will serve as the template for all other programs developed using ISaGRAF PRO.

The config_web_rll folders in Figure 2-2 (for V4.20 and later) have been expanded to show what you should have after extracting the zip files.

Figure 2-2 Directory Structure for V4.20 After Installation of the Telvent-Provided Zip File



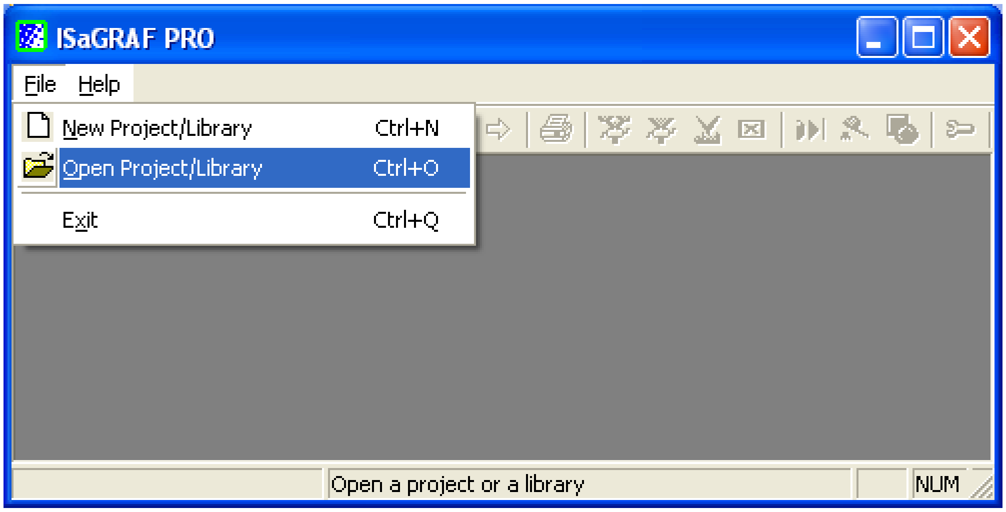
2.4.4 Step 4: Updating The Database

After you install ISaGRAF, you will have a folder on your desktop. Open the ISaGRAF folder on your desktop and double-click the ISaGRAF icon.

Note: If you get any warning messages during installation or operation, contact the makers of ISaGRAF for the latest drivers.

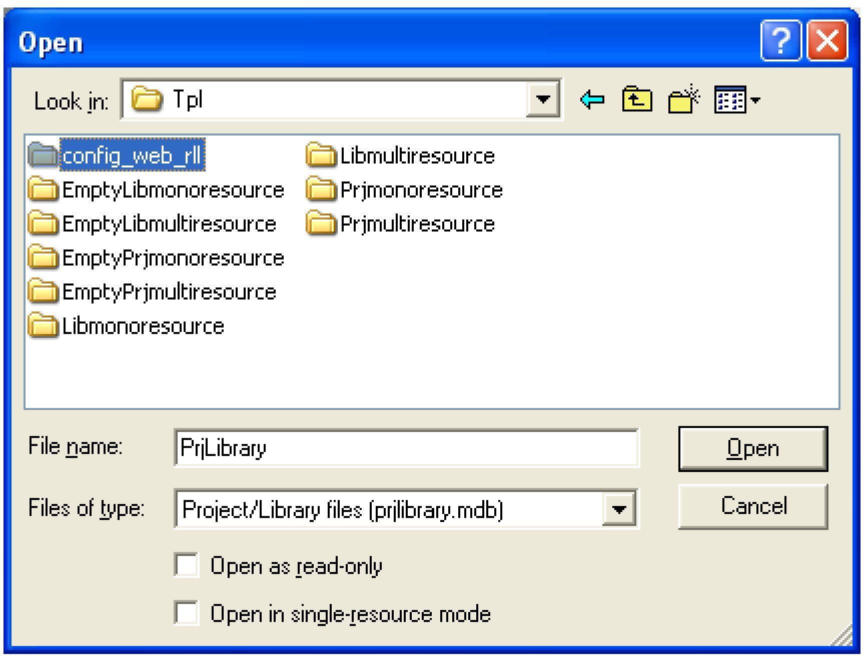
Under File, click Open Project/Library.

Figure 2-3 Launching ISaGRAF PRO



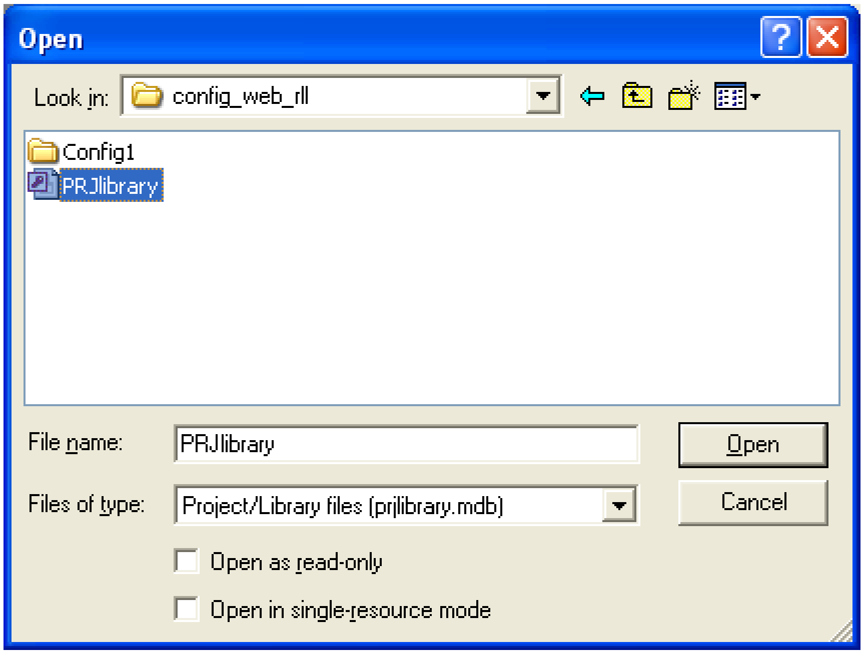
Navigate to Tpl directory and select config_web_rll as shown:

Figure 2-4 Selecting config_web_rll



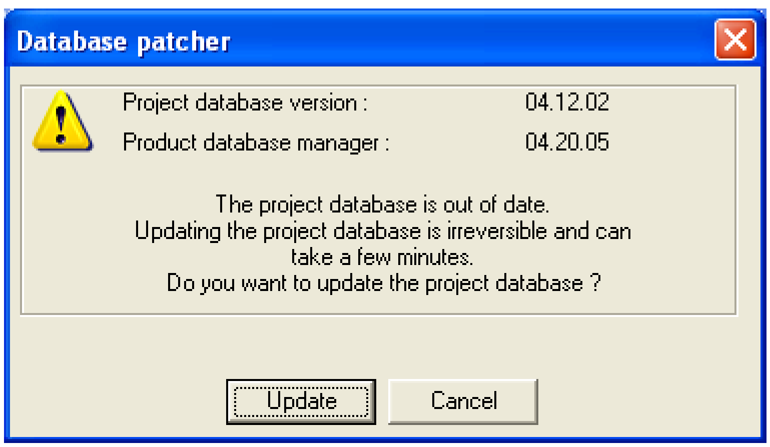
In the next pop-up select PRJlibrary:

Figure 2-5 Selecting PRJlibrary



Depending on the version of ISaGRAF Workbench, you may get the following notice:

Figure 2-6 Database Patcher Notice



Select "Update". This will take a few seconds to convert the template to the newer format.

Installation is complete.

CHAPTER 3

Operation

3.1 Introduction

The basic sequence of operation for Telvent RLL is:

1. Update the ISaGRAF database (if this has not been done, see the Installation chapter)
2. Create and test a program on your PC
3. “Wire” the variables in the program to Telvent drivers
4. Download the program to a Telvent config@WEB RTU
5. Map the required points on the RTU
6. Test the program on the RTU using the ISaGRAF Debug function
7. Test the program on the RTU in the real world

Although it is possible to create six different types of programs in ISaGRAF, Telvent supports only Ladder Diagram (LD) and Function Block Diagram (FBD) programs. Please consult the makers of ISaGRAF for other support.

To complete the exercises in this chapter, your RTU must be connected and operational through the Ethernet – TCP/IP config@WEB interface port.

3.2 Creating Simple Programs

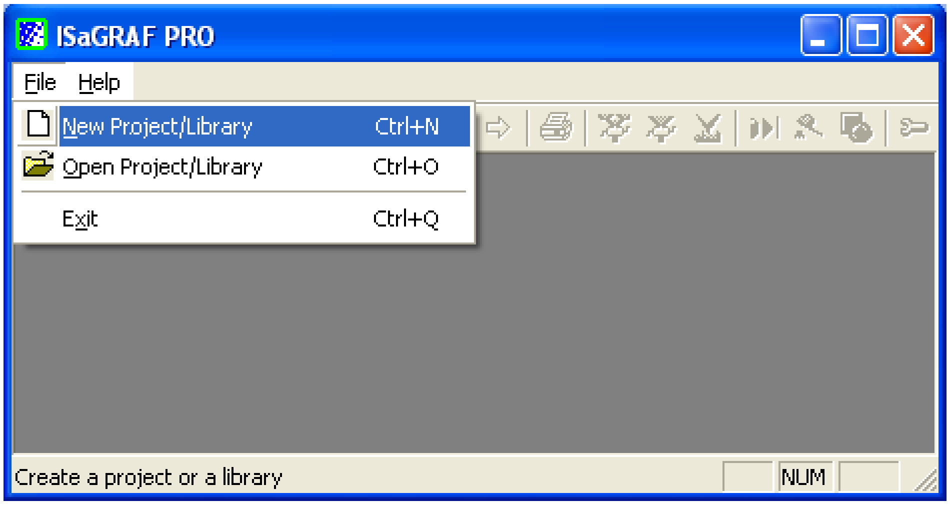
The main object of this chapter is to create and operate a few very simple programs in both Ladder Diagram (LD) and Function Block Diagram (FBD), then download those programs to your RTU. The programs use status points to trip SBOs on the RTU.

3.2.1 Starting a New Project

A project can be a single program or a group of programs. A project must be compiled as a unit and downloaded to the RTU as a unit. Only one project at a time can run on the RTU.

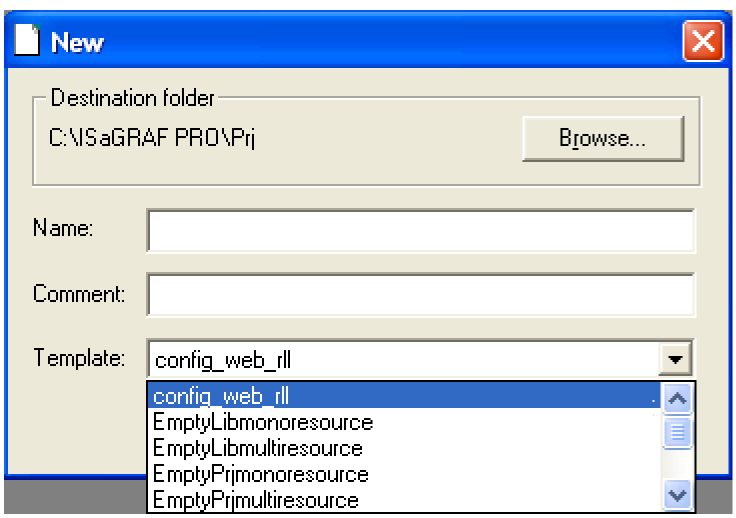
Under File, click New Project/Library. The next time you want to work on this project, you will be able to find it under Open Project/Library.

Figure 3-1 Launching a New Project



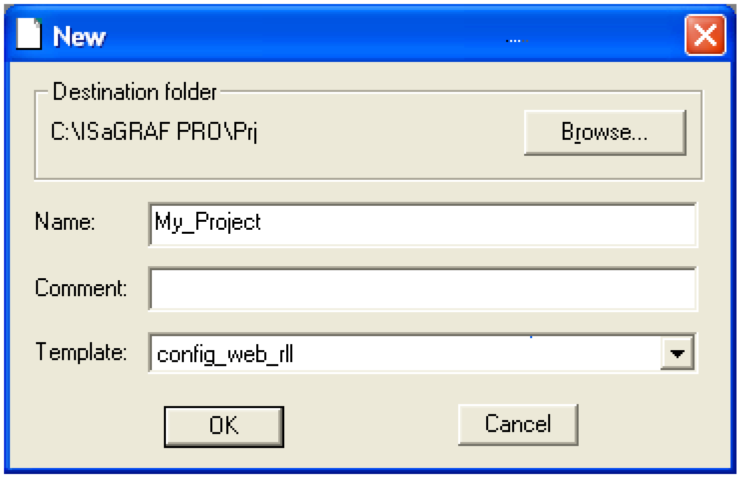
From the Template list, select config_web_rll as shown. This is the only template that will work for Telvent applications.

Figure 3-2 Selecting the Template



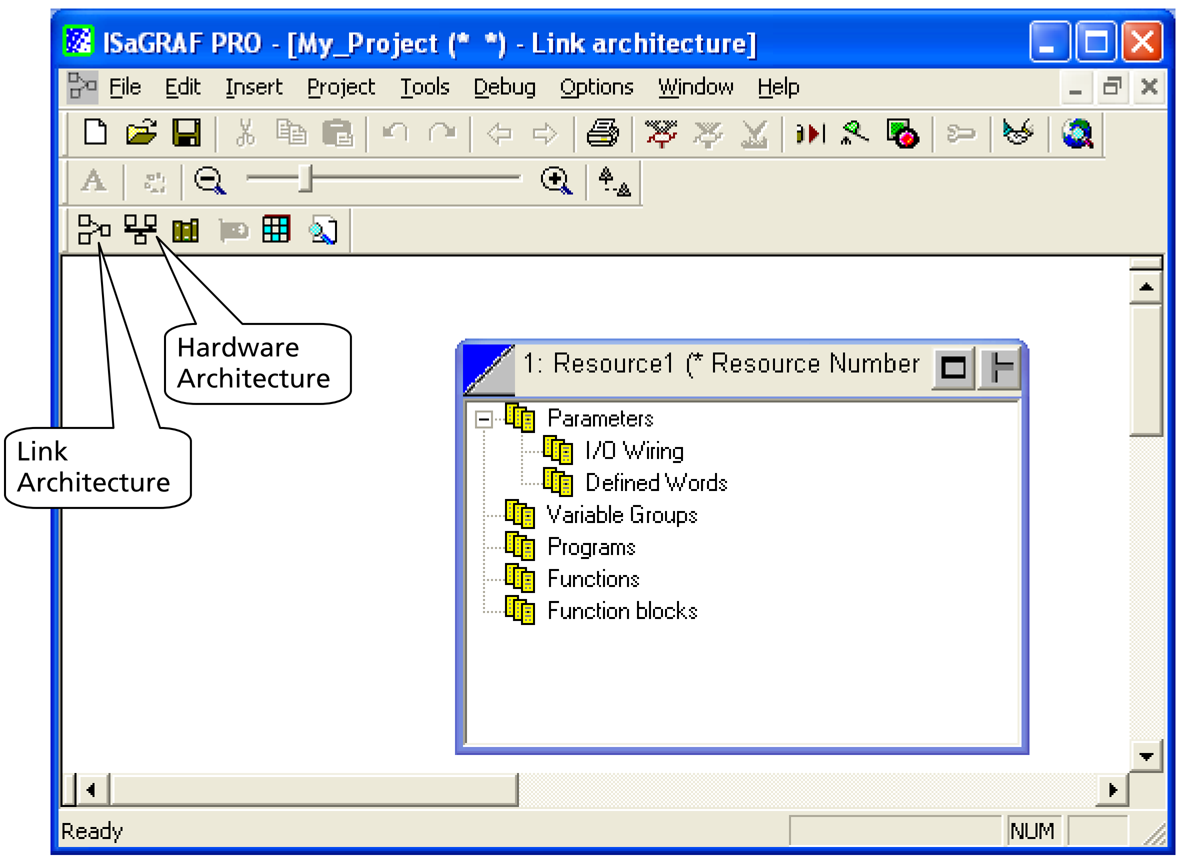
In the "Name" field, type an appropriate name. In the example, "My_Project" is used. Note that blank spaces are not allowed in the name. You may enter a Comment about the project or you may leave this field blank.

Figure 3-3 Naming a New Proiject



Click OK. A project screen similar to the one below will appear. This view is called Link Architecture.

Figure 3-4 "My_Project" Project Screen

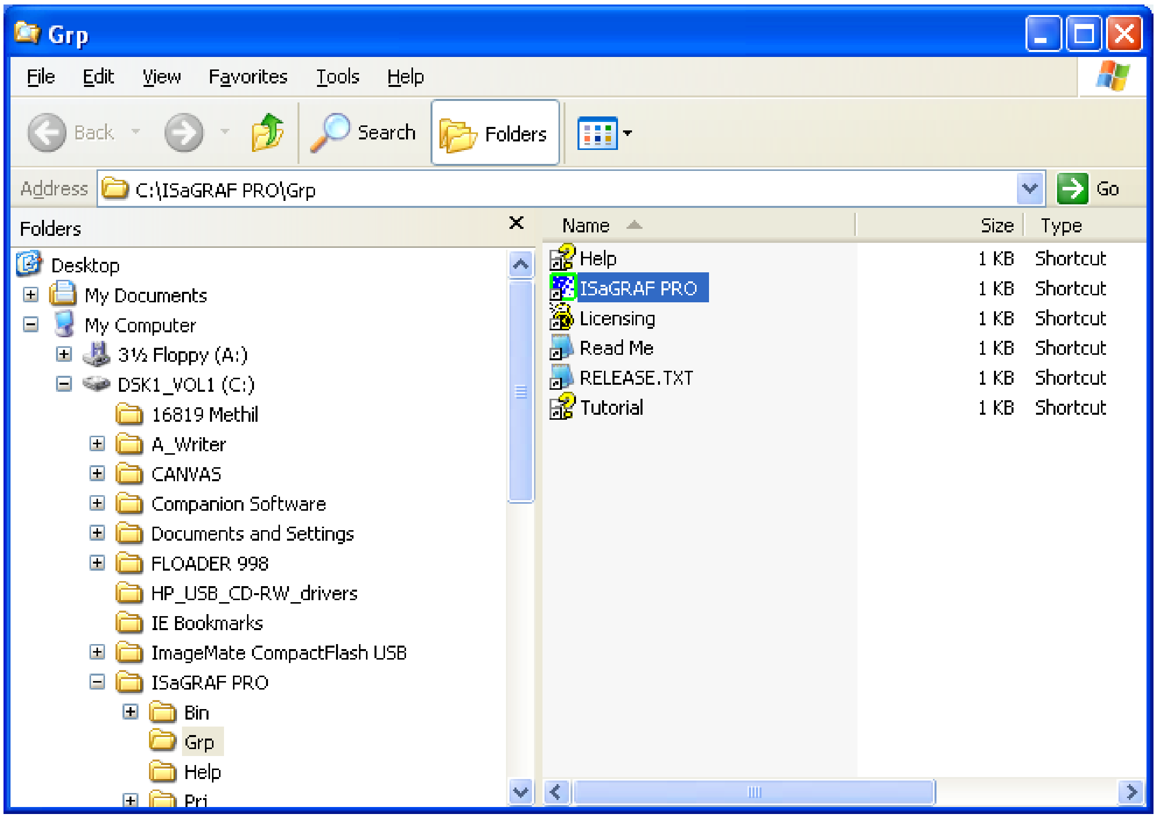


3.2.2 Opening an Existing Project

After you install ISaGRAF PRO, you will have a folder on your desktop. Open the ISaGRAF PRO folder on your desktop and double-click the ISaGRAF PRO icon.

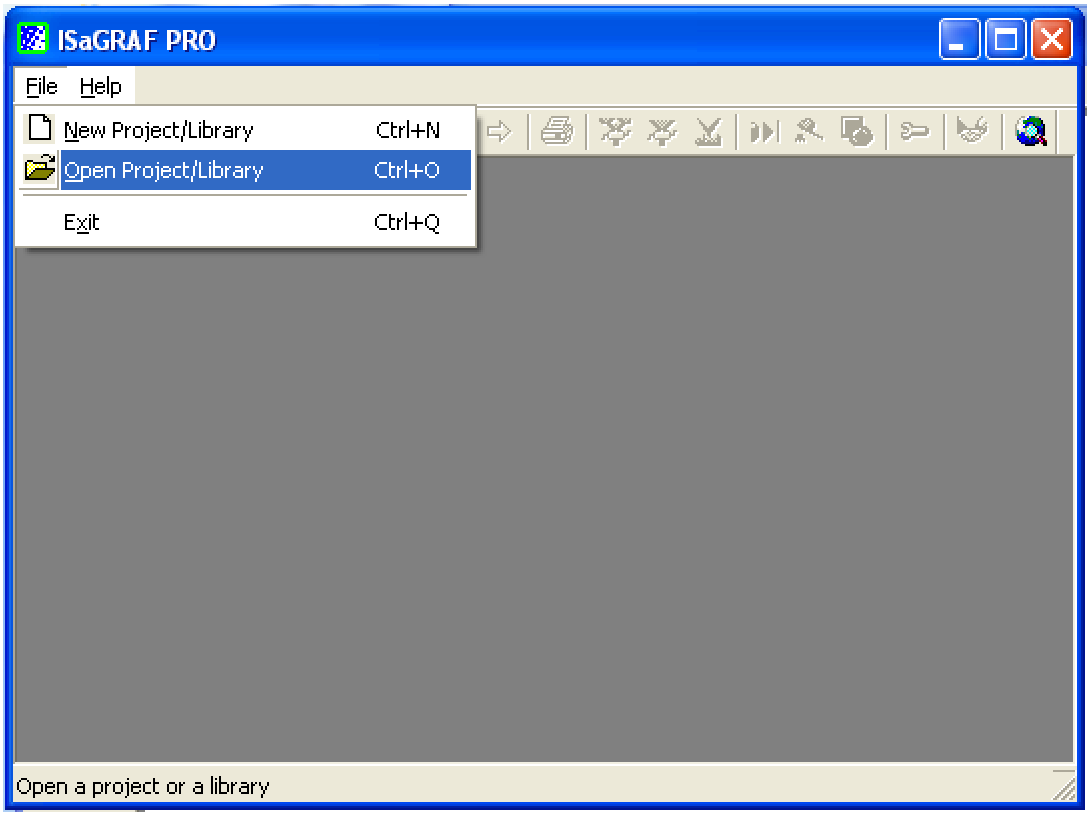
Note: If you get any warning messages during installation or operation, contact the makers of ISaGRAF for the latest drivers.

Figure 3-5 Launching ISaGRAF PRO



Under File, click Open Project/Library.

Figure 3-6 Launching a New Project



From the "Look in:" list, select the ISaGRAF PRO folder (V4,12 – see Figure 3-7) or the ISaGRAF folder (V4.20 – see Figure 3-8).

Figure 3-7 Finding an Existing Project (V4.12)

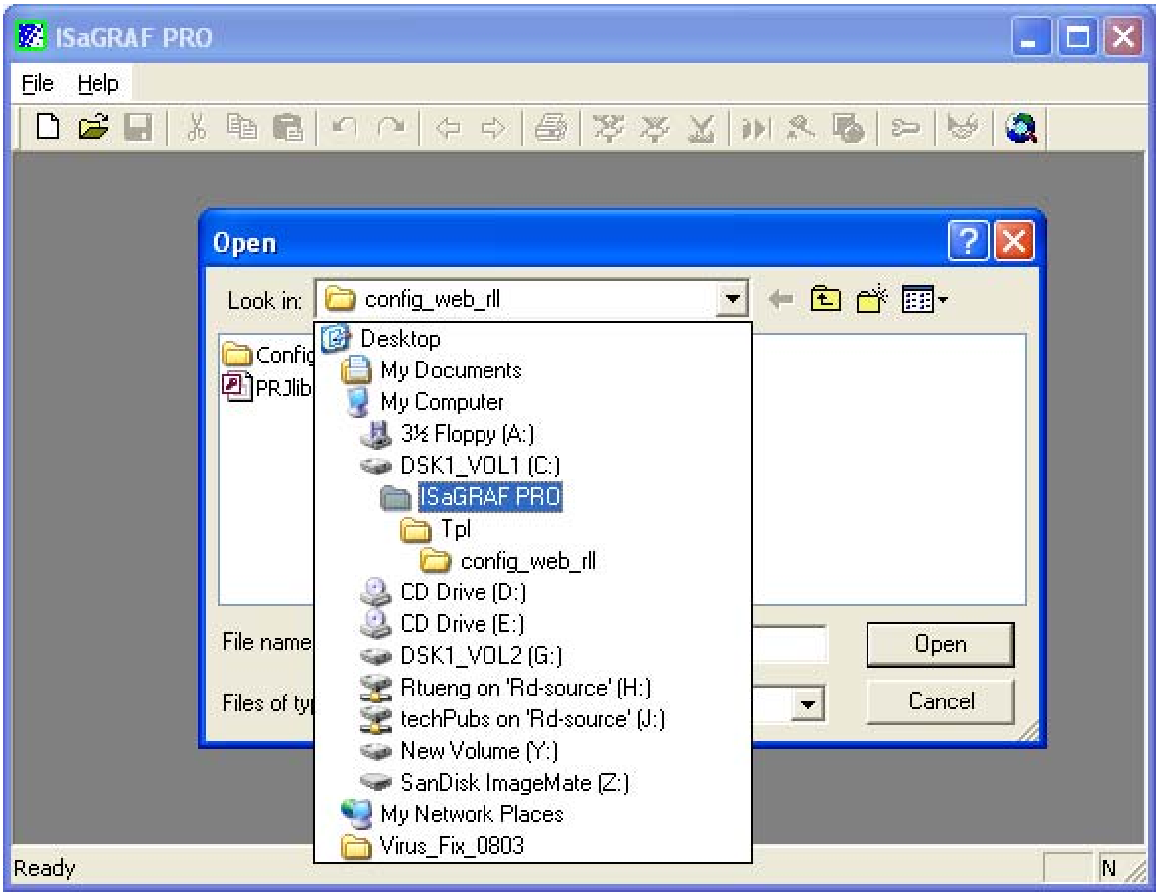
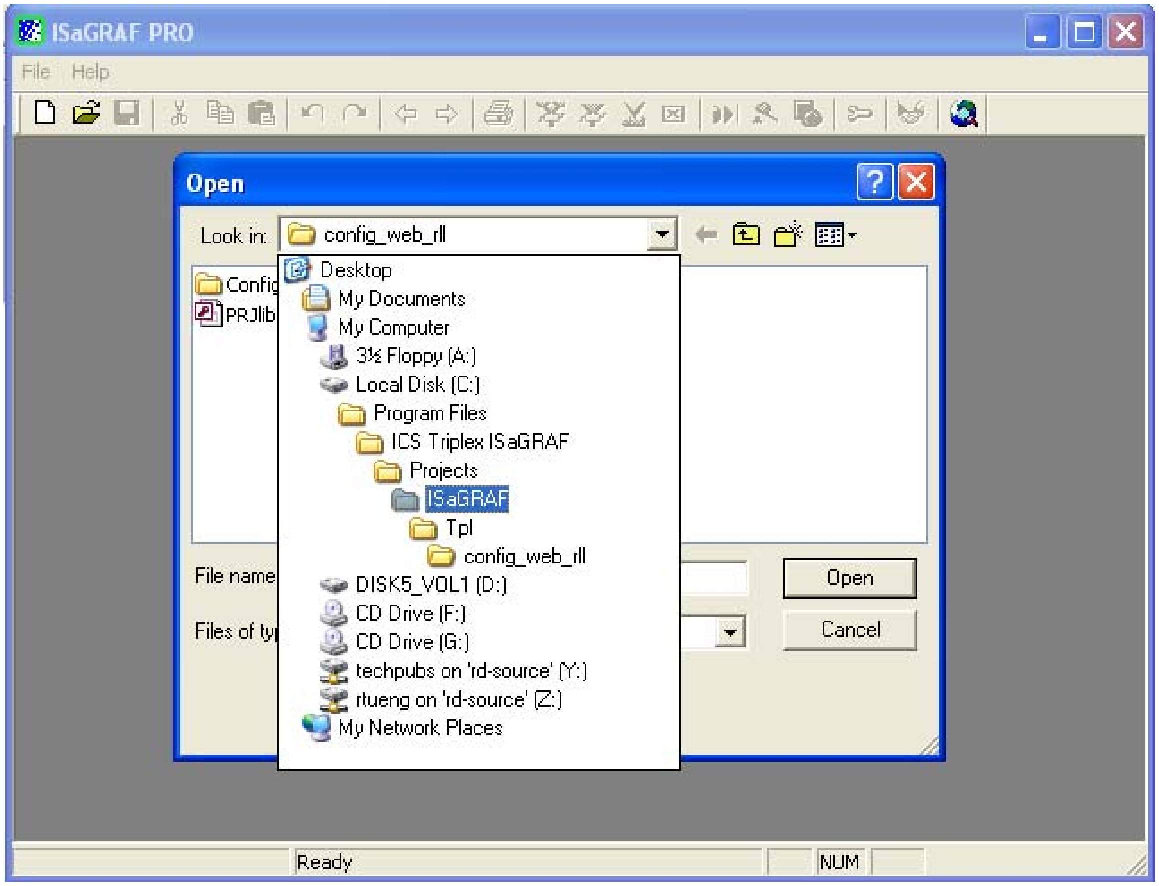
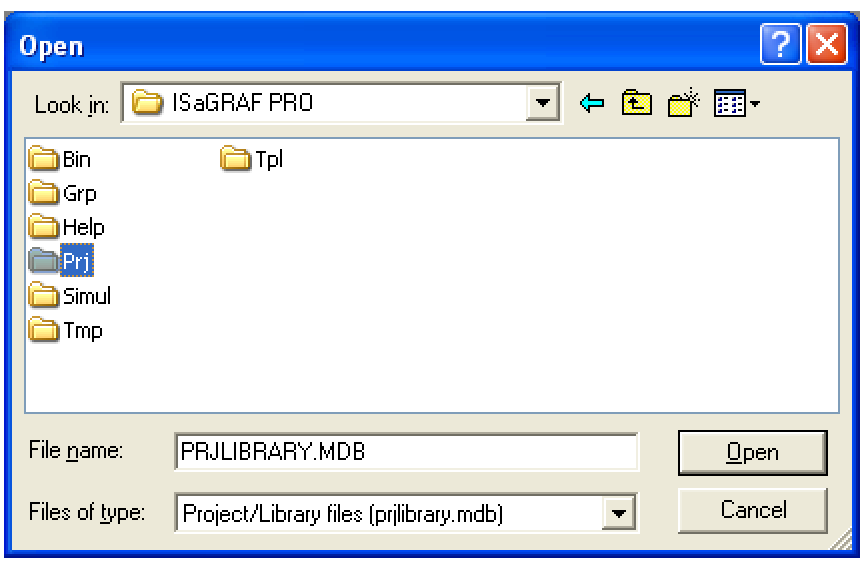


Figure 3-8 Finding an Existing Project (V4.20)



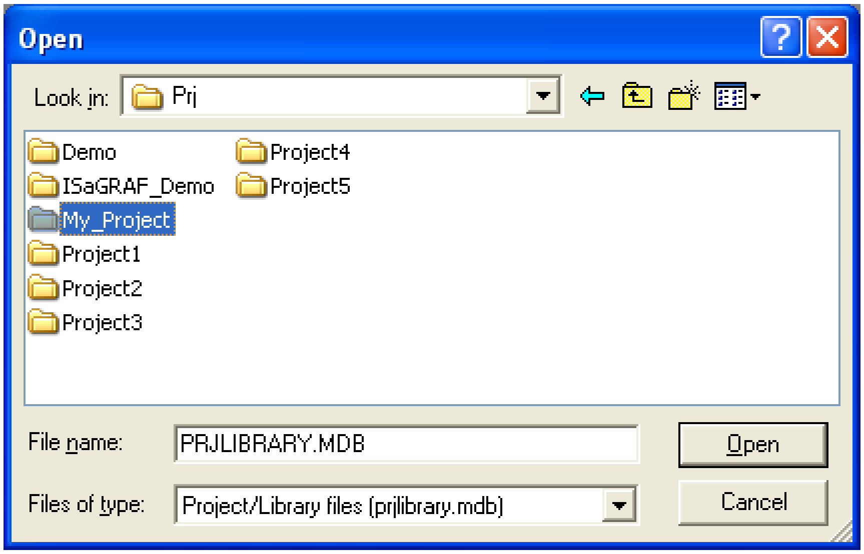
From the ISaGRAF PRO folder, open the Prj folder as shown below.

Figure 3-9 Opening the Prj folder



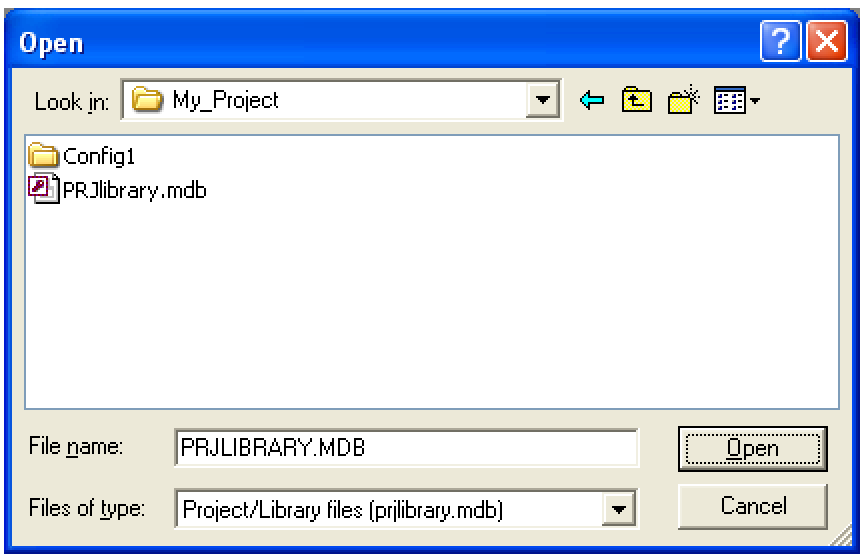
From the Prj folder, open the My_Project folder as shown below. (My_Project is the name of the project assigned under the section “3.2.1 Starting a New Project”. If you named your project something else, then, of course, that is the folder name you should now be opening.)

Figure 3-10 Opening My_Project



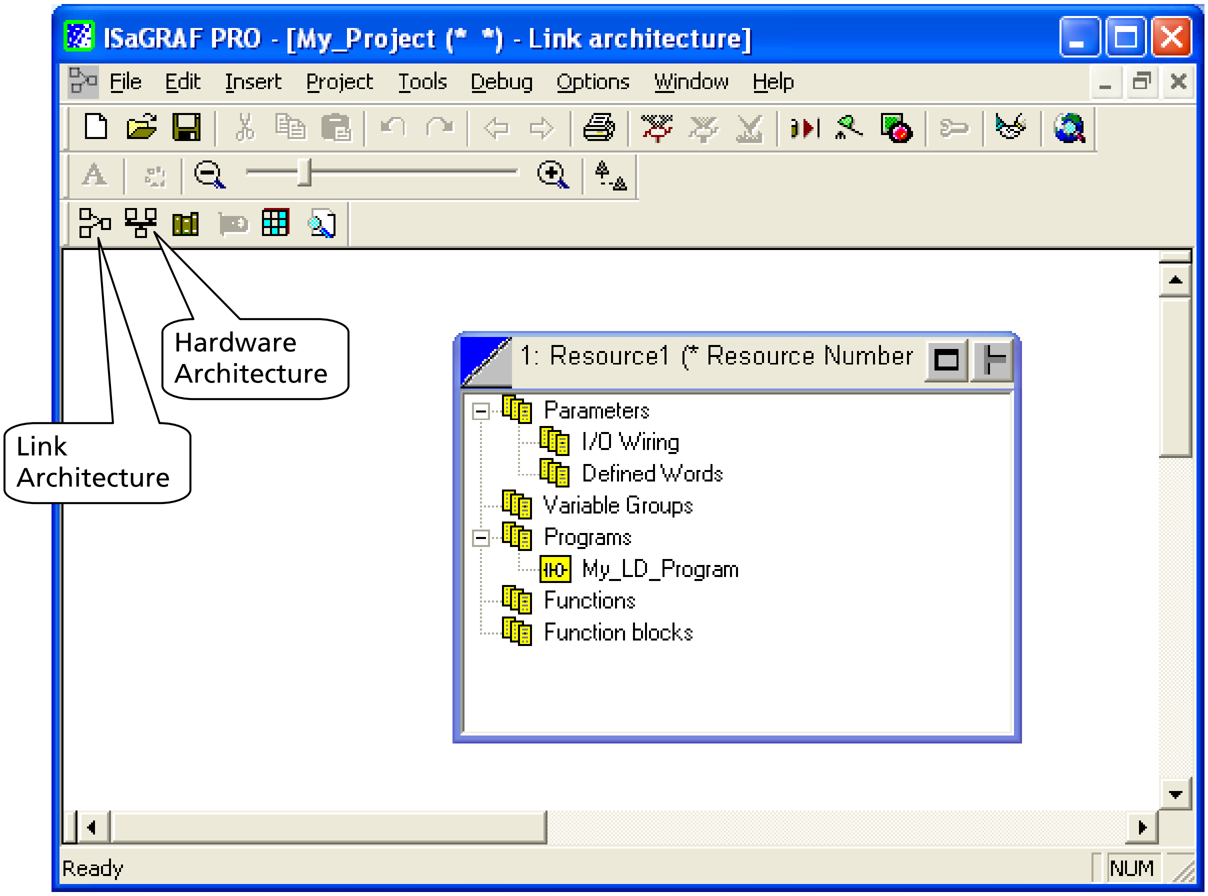
The final step is opening the PRJLIBRARY.MDB file as shown below.

Figure 3-11 Opening PRJLIBRARY.MDB Under My_Project



Click Open. My_Project project screen will appear. This view is called Link Architecture.

Figure 3-12 "My_Project" Project Screen

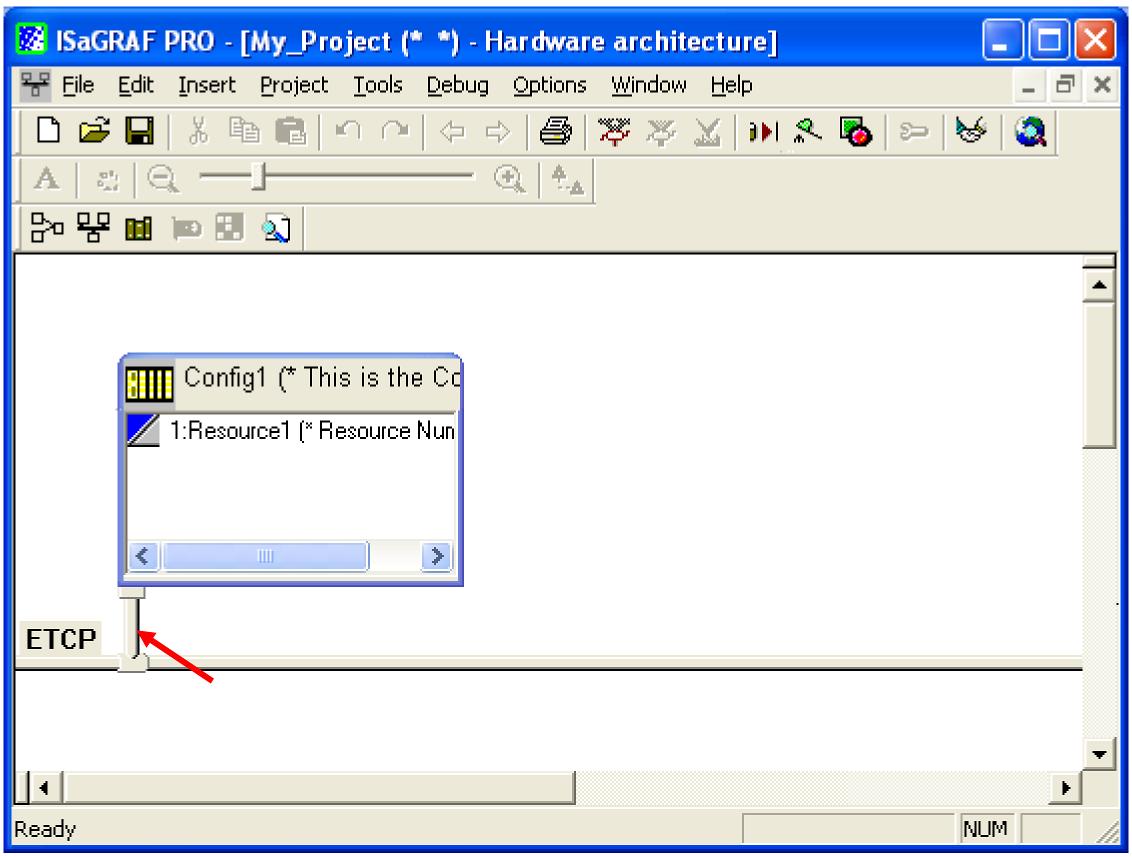


3.2.3 RTU Communications Settings

During the course of your project, you must download to the RTU over Ethernet. This means ISaGRAF PRO must know the RTU's IP address. Follow the instructions below to set this up. Once set, the IP address will be part of the project. If you want to download this project to another RTU with a different IP address, then you must repeat the procedures below to set the new IP address.

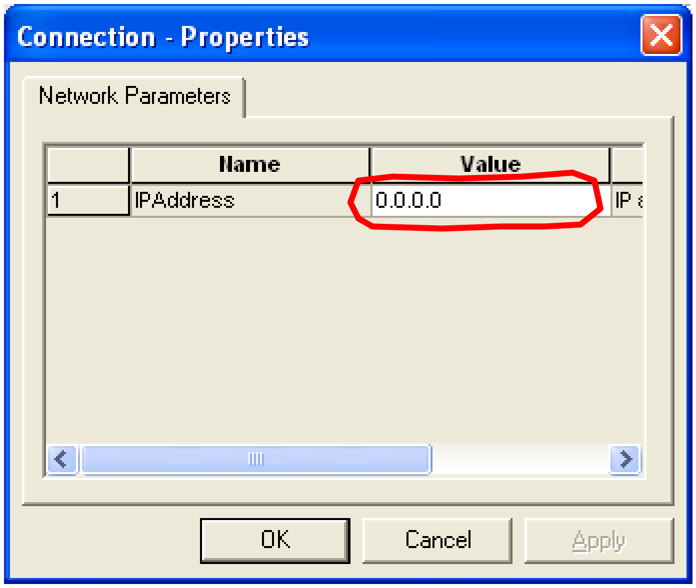
Click on the hardware architecture icon as shown by Figure 3-4. A screen similar to the one below will appear.

Figure 3-13 Hardware Architecture Screen



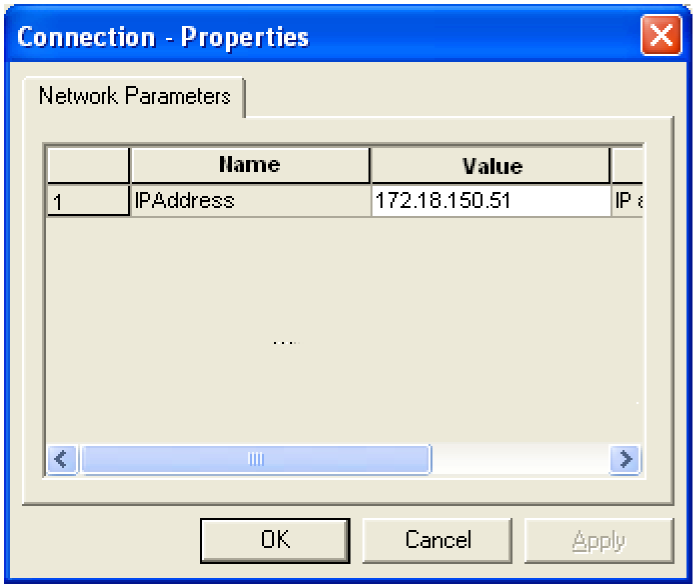
Double-click on the vertical bar shown by the arrow in Figure 3-13. A screen similar to the one shown below will appear. Enter the IP address for your RTU in the field circled below.

Figure 3-14 Connection – Properties



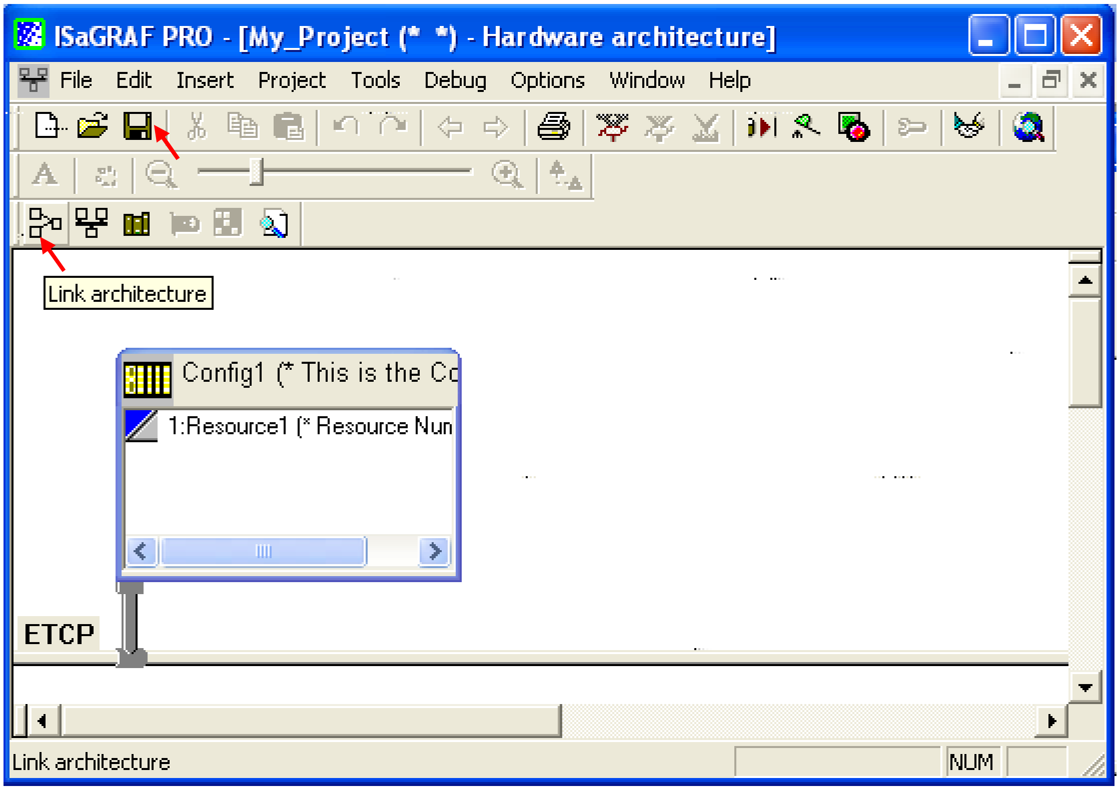
An example IP address is shown for the Connection – Properties box below. Click OK

Figure 3-15 Example of IP Address for Connection



You have completed the TCP/IP connection parameters. Click the Save icon and then the Link architecture icon as shown below.

Figure 3-16 Completing the Connection Parameters

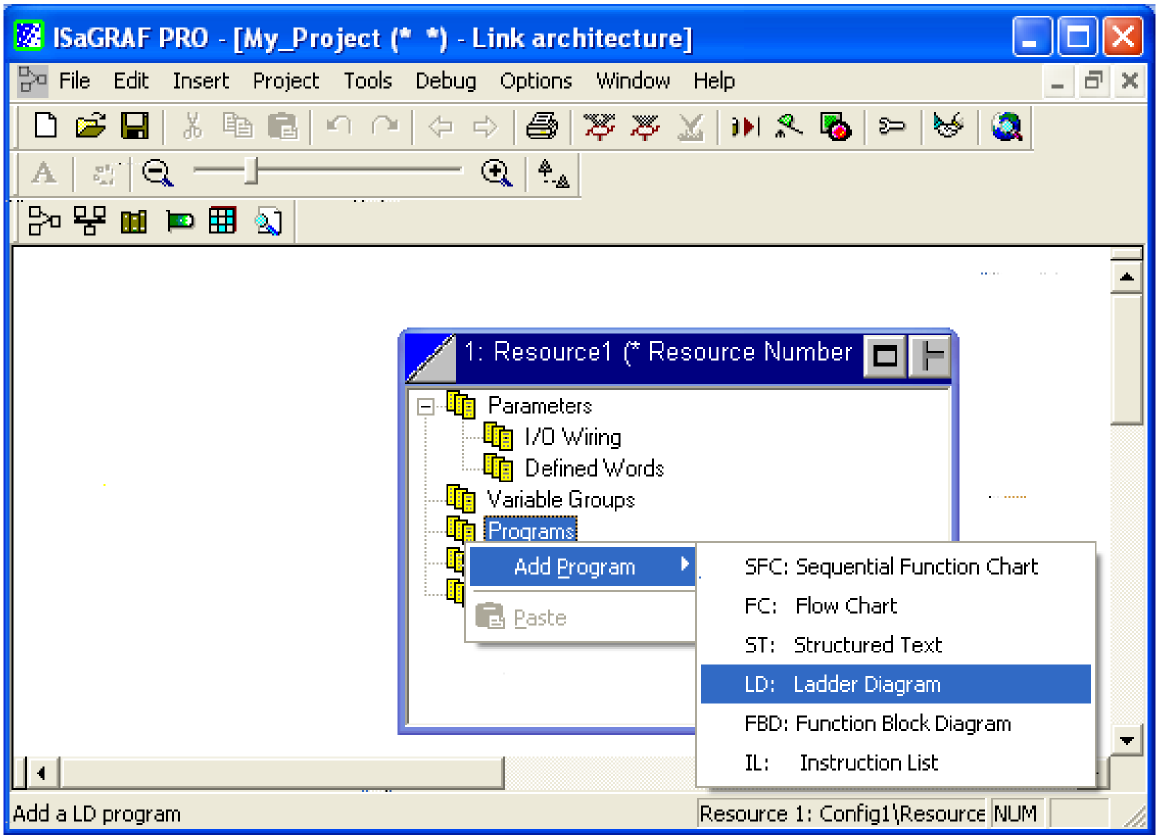


3.2.4 Starting a New Ladder Diagram (LD) Program

The reason for a project is to create programs that do useful work. The following instructions tell you how to create a simple Ladder Diagram.

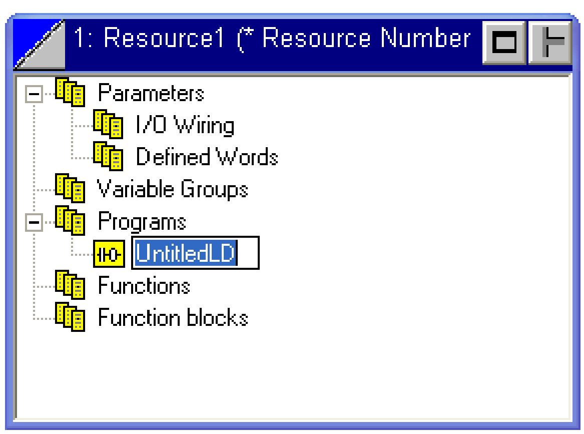
In the Link Architecture display, right-click on Programs and select LD: Ladder Diagram as shown below.

Figure 3-17 Selecting the Type of Program to Create



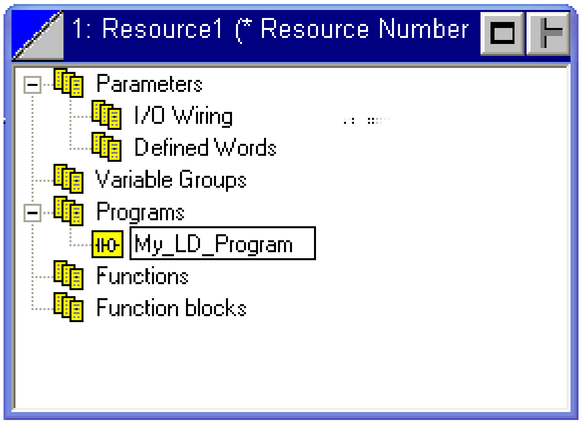
After the type of program has been selected, the generic title will remain highlighted so that you can change the name to something more meaningful.

Figure 3-18 Generic Program Name Selected



Type the name of your program. Don't put in spaces. You may simulate spaces with underscores as shown below. Hit Enter when you are done to accept the name change.

Figure 3-19 Naming the Program

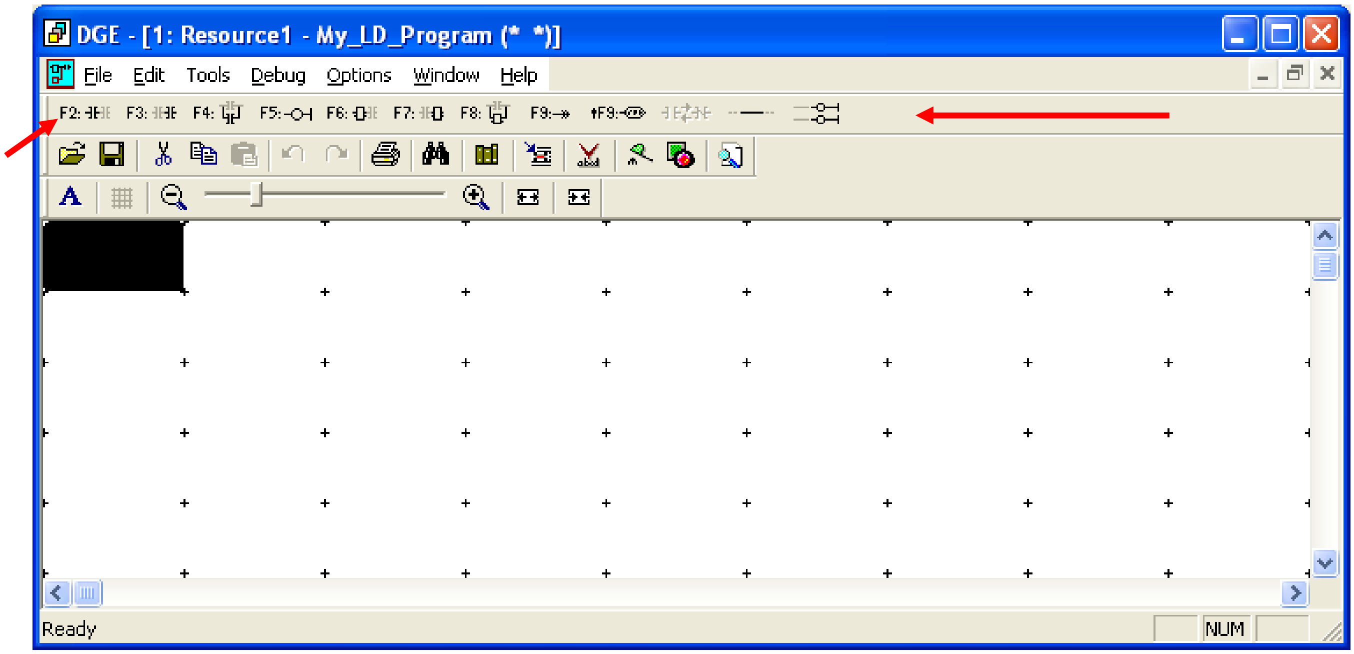


Double-click on the icon for your program. You will get a warning that the project must be saved before continuing. Click Yes.

Figure 3-20 Save Warning

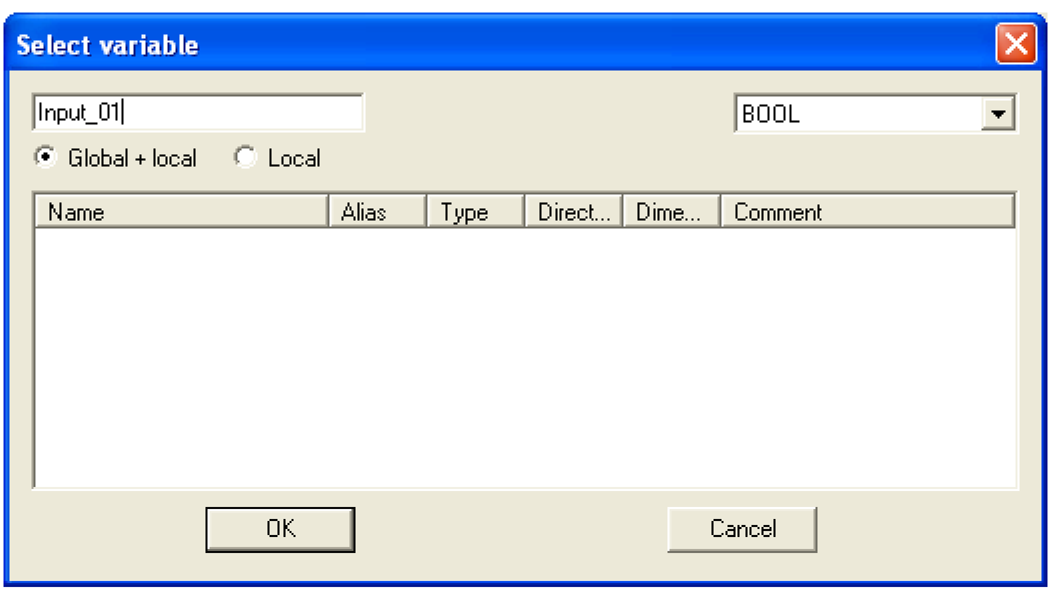
The screen that appears (shown below) is the working space for creating your Ladder Diagram (LD) program. The long arrow (not part of the real ISaGRAF program) points out the main tools for creating ladder logic. When you hover your cursor over an icon, a short explanation of the icon's function will appear in a pop-up box and at the bottom-left of the screen.

Figure 3-21 Ladder Diagram (LD) Work Space



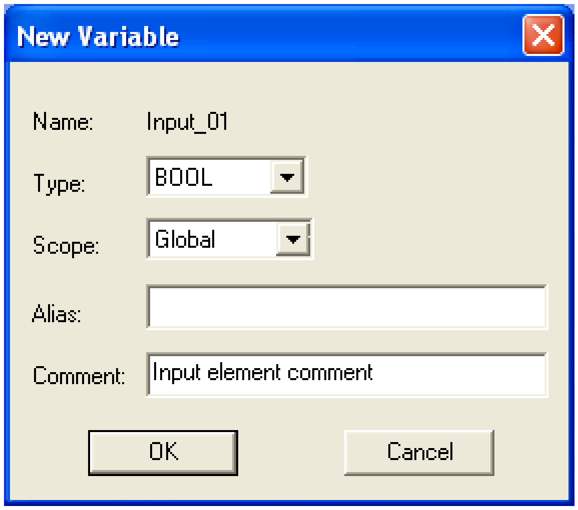
We will create a very simple program by clicking on the leftmost tool (see short arrow above), which is a "Contact on the left". When you click on the tool, you will get a "variable" dialog box as shown below. You must create a variable name for the "Contact". In the example below, "Input_01" was chosen. Do not use blank spaces in names. In this case, we will leave the default as Global + local, and leave the type of variable as Bool (it is a switch; it will be ON or OFF). Click OK.

Figure 3-22 Selecting a Variable Name



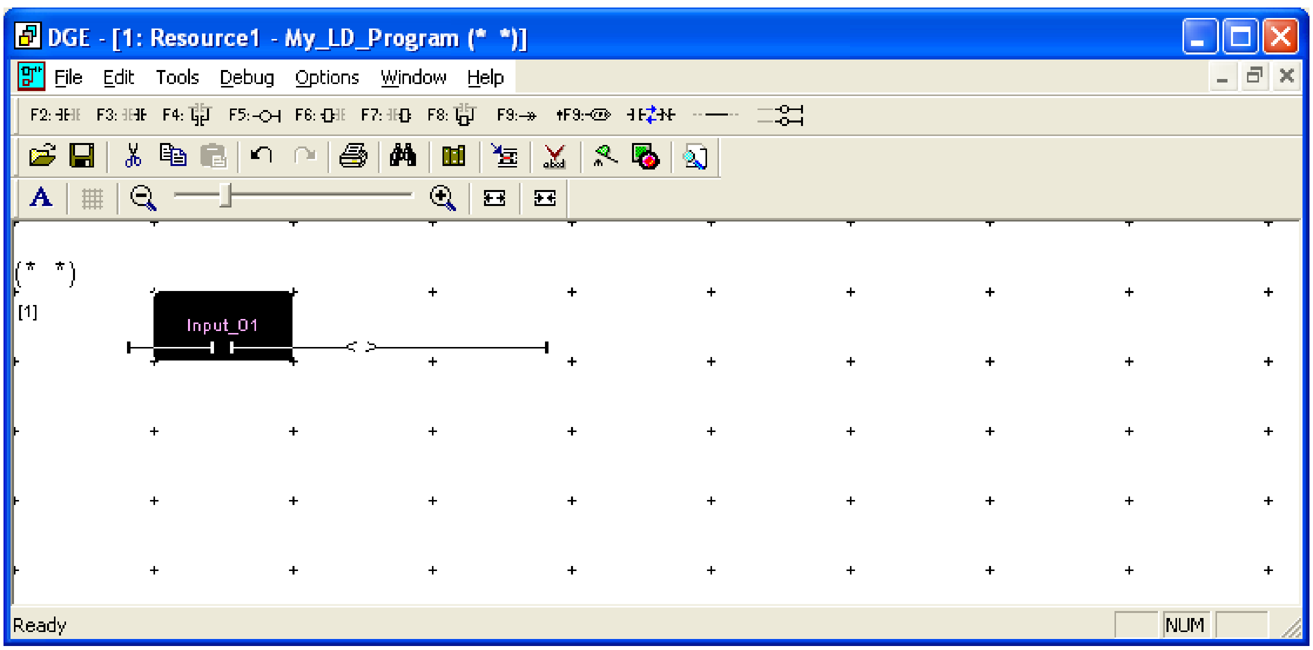
The New Variable dialog box gives you more choices. You can assign an alias to your variable. This is usually not needed, so we will leave it blank. You may enter a comment if you like. You may also leave the Comment field blank or enter a Comment later. Click OK.

Figure 3-23 New Variable



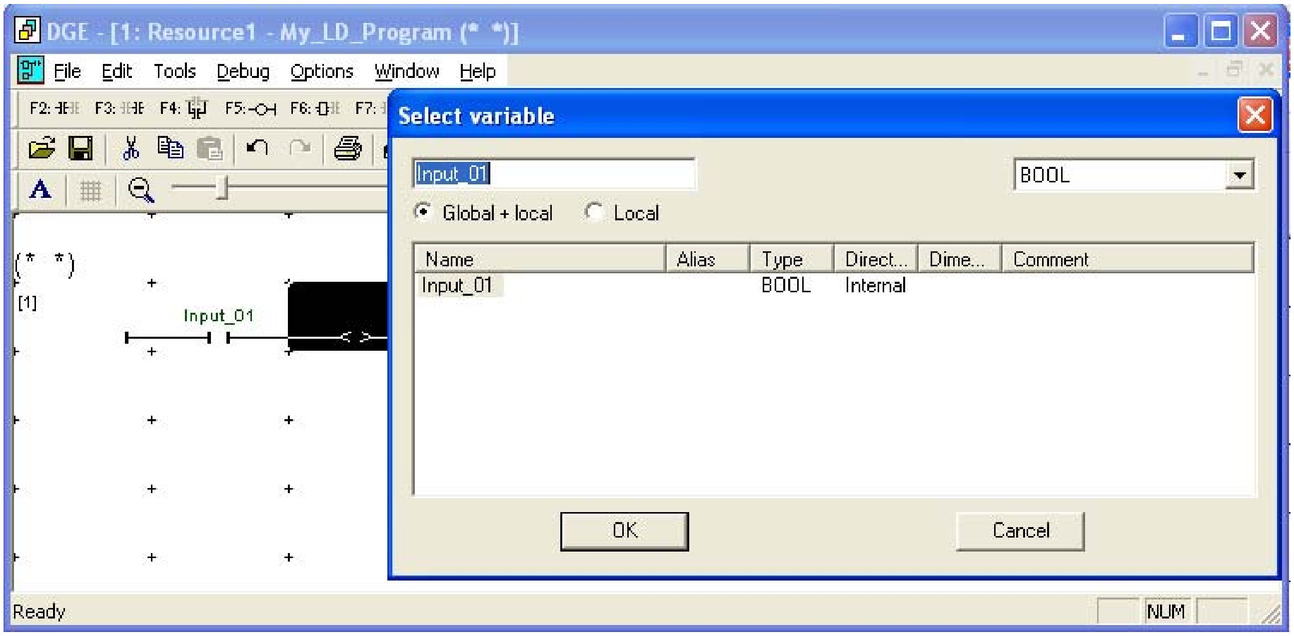
The program has automatically created an output on our ladder rung, as shown below.

Figure 3-24 Ladder Rung



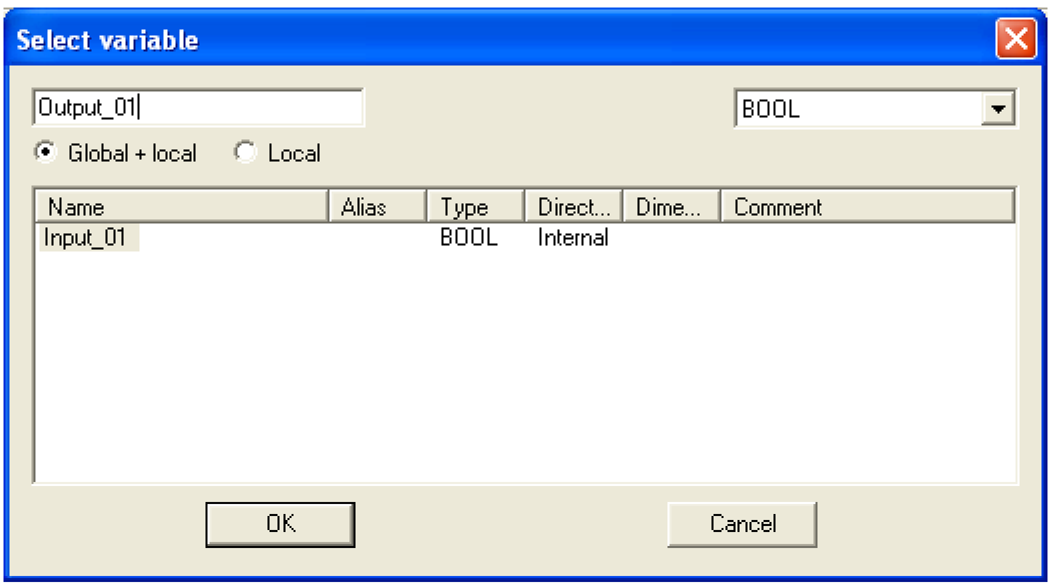
Double-click on the output element to get the screen shown below. The default for the variable name is the last variable assigned.

Figure 3-25 Naming the New Variable



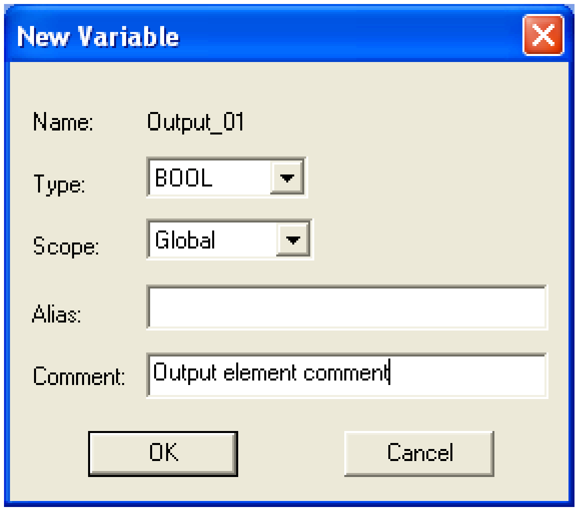
Change this to an appropriate name such as was chosen below. Click OK.

Figure 3-26 The New Variable Name



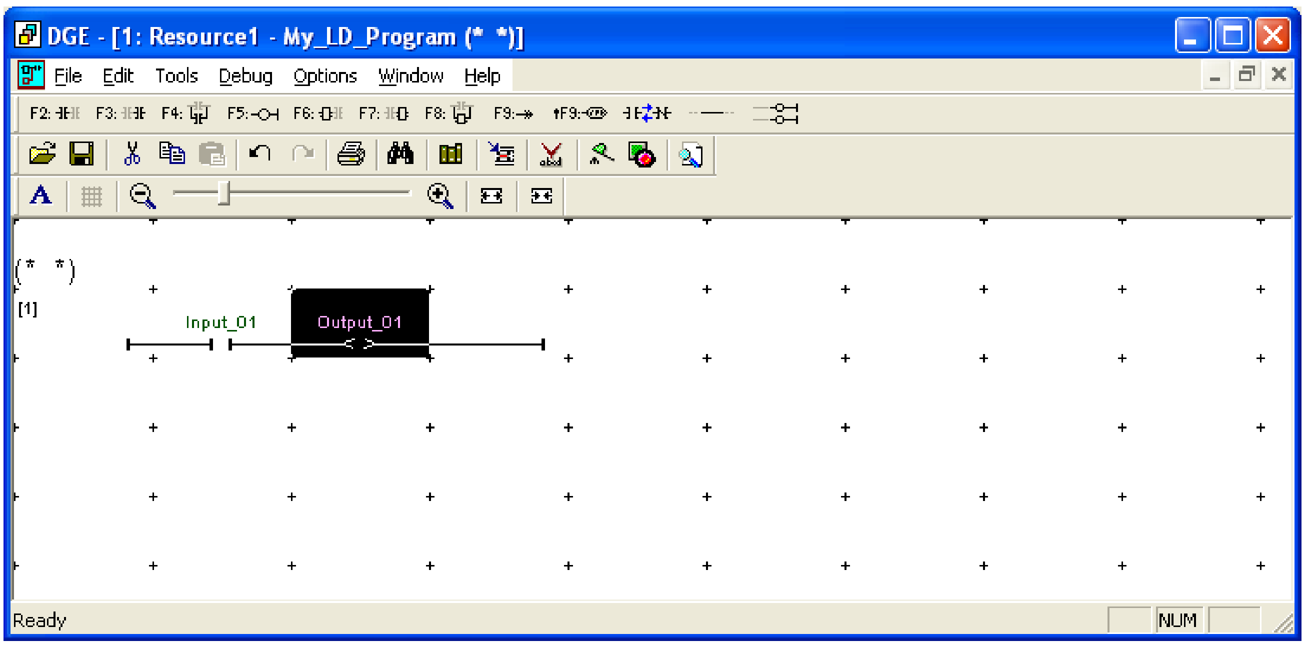
Once again, the New Variable box will come up. You may enter a comment if you like. Click OK.

Figure 3-27 New Variable



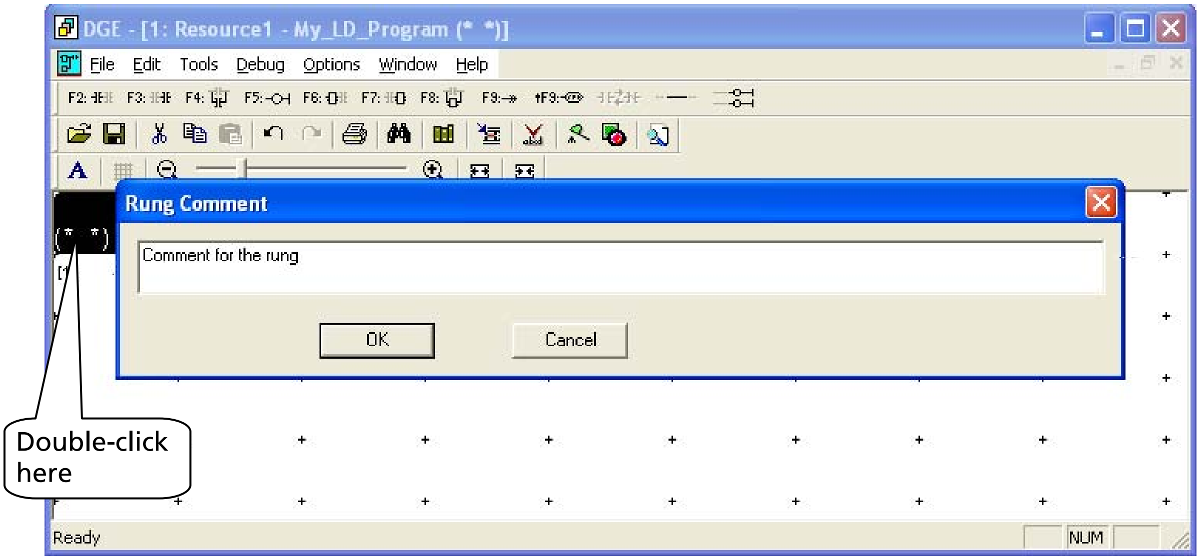
Now both elements of our ladder have variable names as shown below.

Figure 3-28 Elements With Variable Names



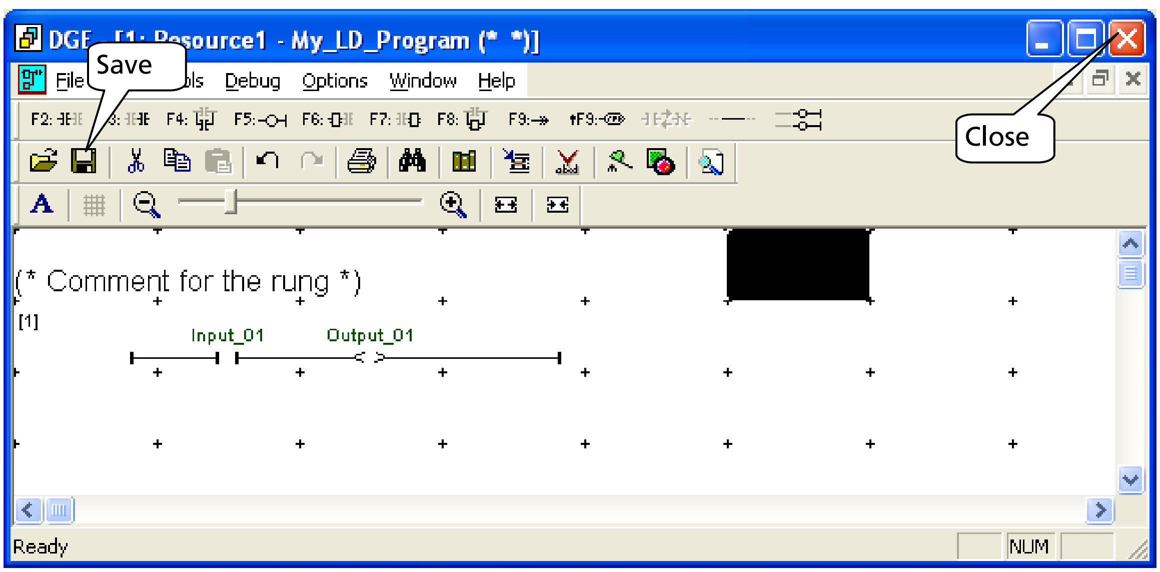
If you double-click on the comment brackets as shown below, you can add rung comments.

Figure 3-29 Adding a Rung Comment



The rung comment results are shown below.

Figure 3-30 Rung Comments



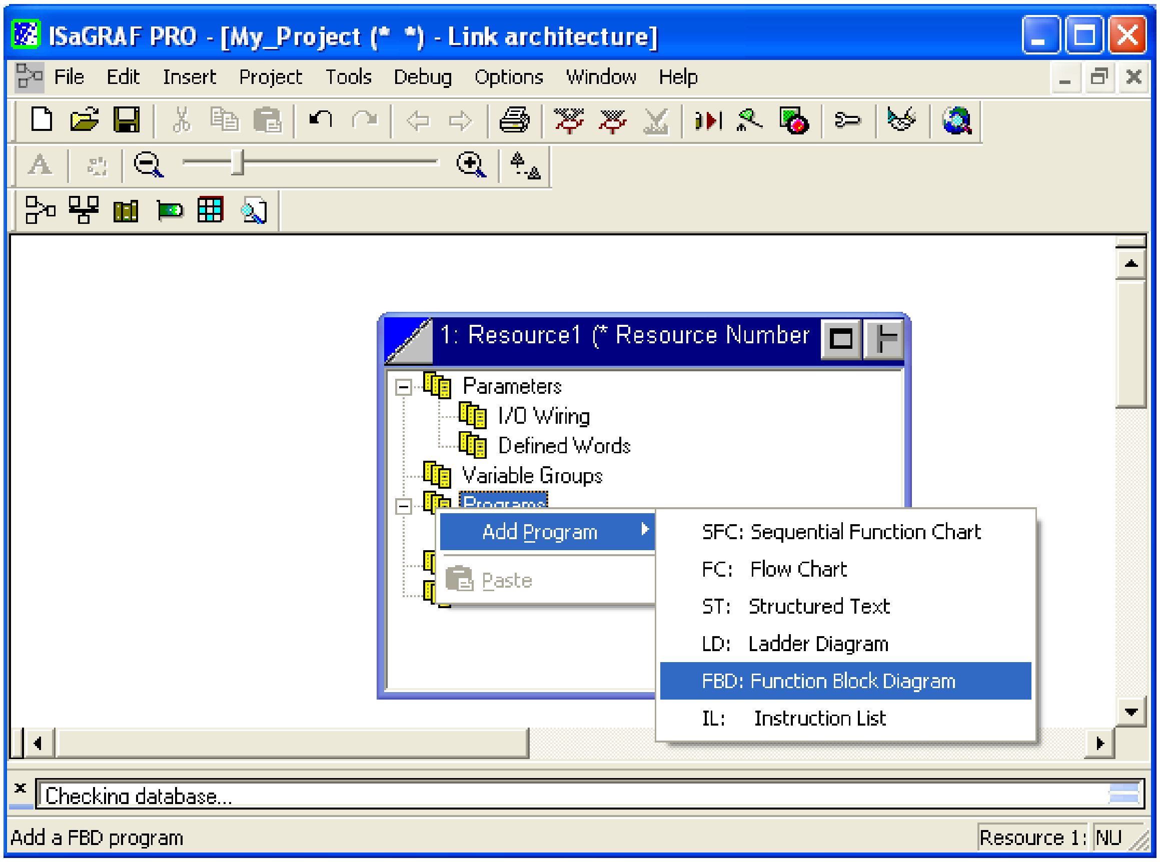
Save your program by clicking on the icon shown above. Close the LD program by clicking on the top-right X.

3.2.5 Starting a New Function Block Diagram (FBD) Program

The reason for a project is to create programs that do useful work. The following instructions tell you how to create a simple Function Block Diagram.

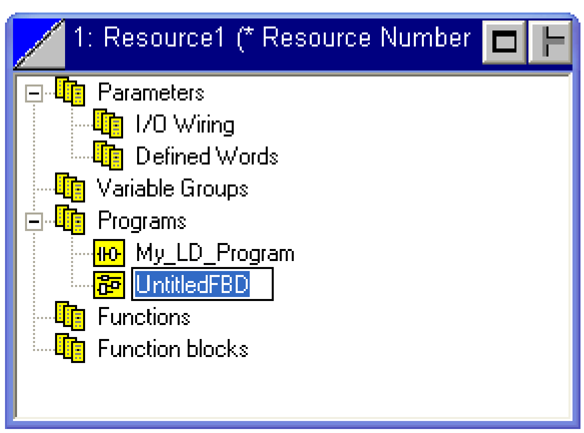
In the Link Architecture display, right-click on Programs and select FBD: Function Block Diagram as shown below.

Figure 3-31 Selecting the Type of Program to Create



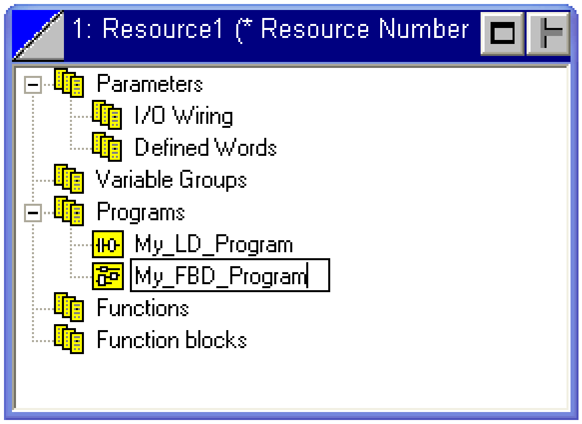
After the type of program has been selected, the generic title will remain highlighted so that you can change the name to something more meaningful.

Figure 3-32 Generic Program Name Selected



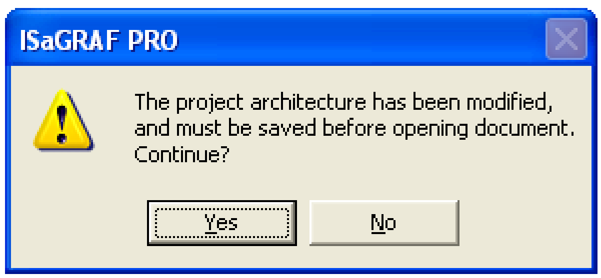
Type the name of your program. Don't put in spaces. You may simulate spaces with underscores as shown below. Hit Enter when you are done to accept the name change.

Figure 3-33 Naming the Program



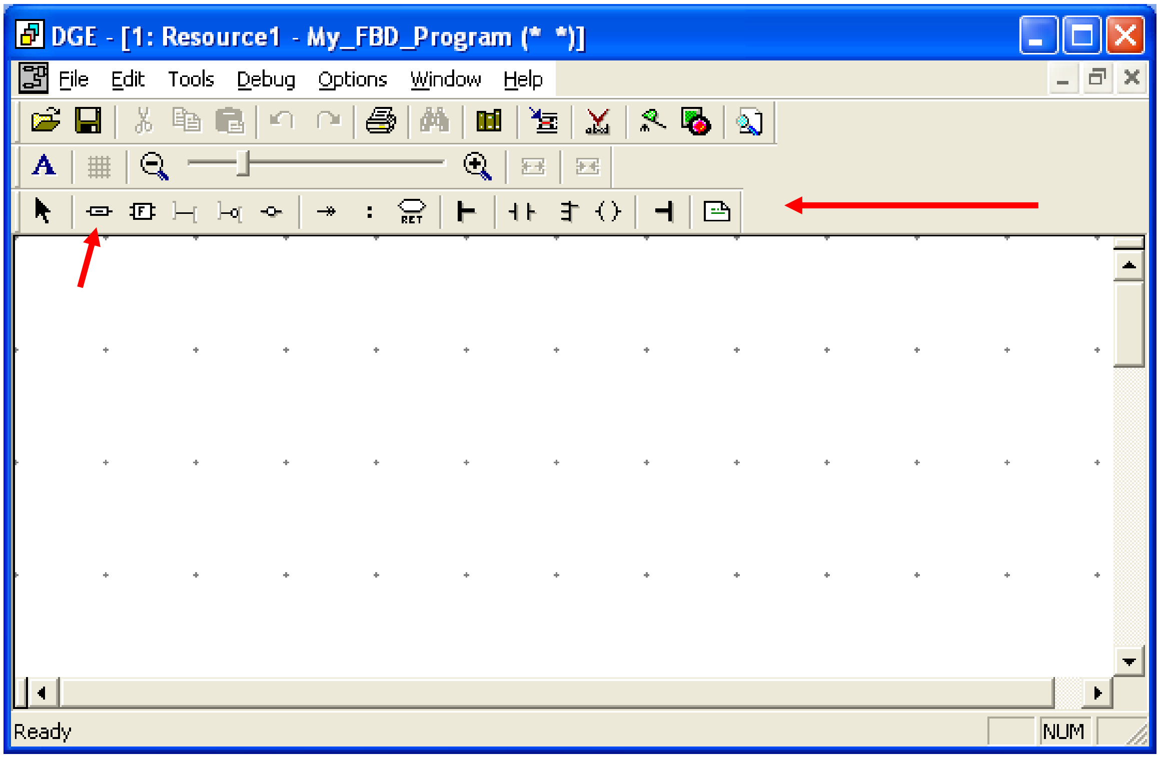
Double-click on the icon for your program. You will get a warning that the project must be saved before continuing. Click Yes.

Figure 3-34 Save Warning



The screen that appears (shown below) is the working space for creating your Function Block Diagram (FBD) program. The long arrow (not part of the real ISaGRAF program) points out the main tools for creating Function Block Diagrams. When you hover your cursor over an icon, a short explanation of the icon's function will appear in a pop-up box and at the bottom-left of the screen.

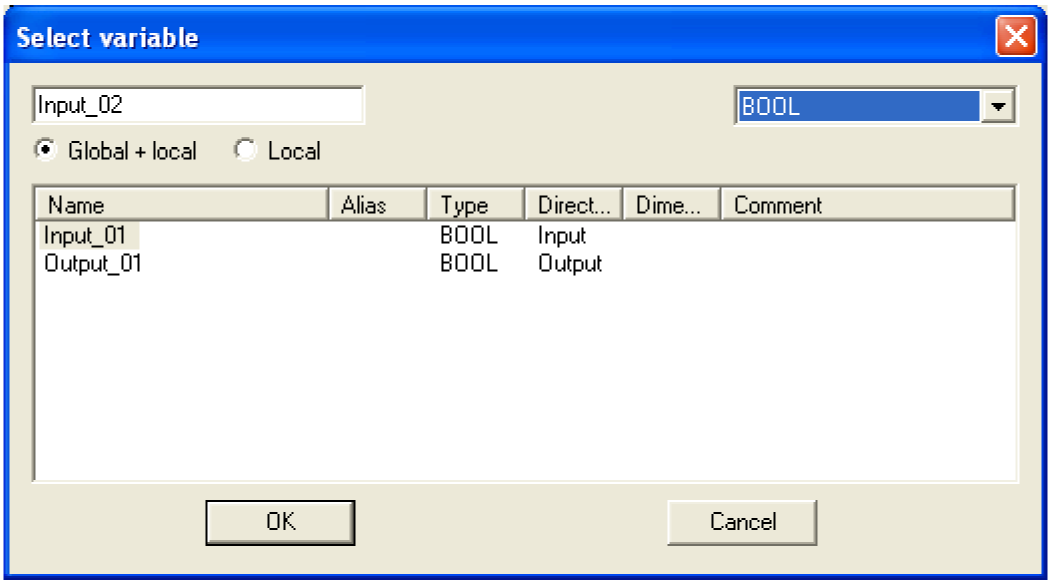
Figure 3-35 Function Block Diagram (FBD) Work Space



We will create a very simple program by clicking on the tool shown by the short arrow above. This is a "Add a variable". When you click on the tool and drag it to the workspace, then click again to drop it on the left side of the workspace, you will get a "variable" dialog box as shown below. (Some variables already exist because we created an LD program with variables in the previous section.)

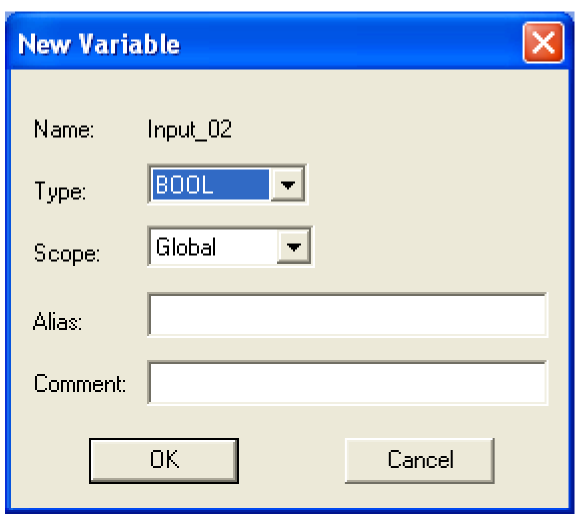
Create a new variable name for this variable. In the example below, "Input_02" was chosen. Do not use blank spaces in names. In this case, we will leave the default as Global + local, and leave the type of variable as Bool (it is a switch; it will be ON or OFF). Click OK.

Figure 3-36 Selecting a Variable Name



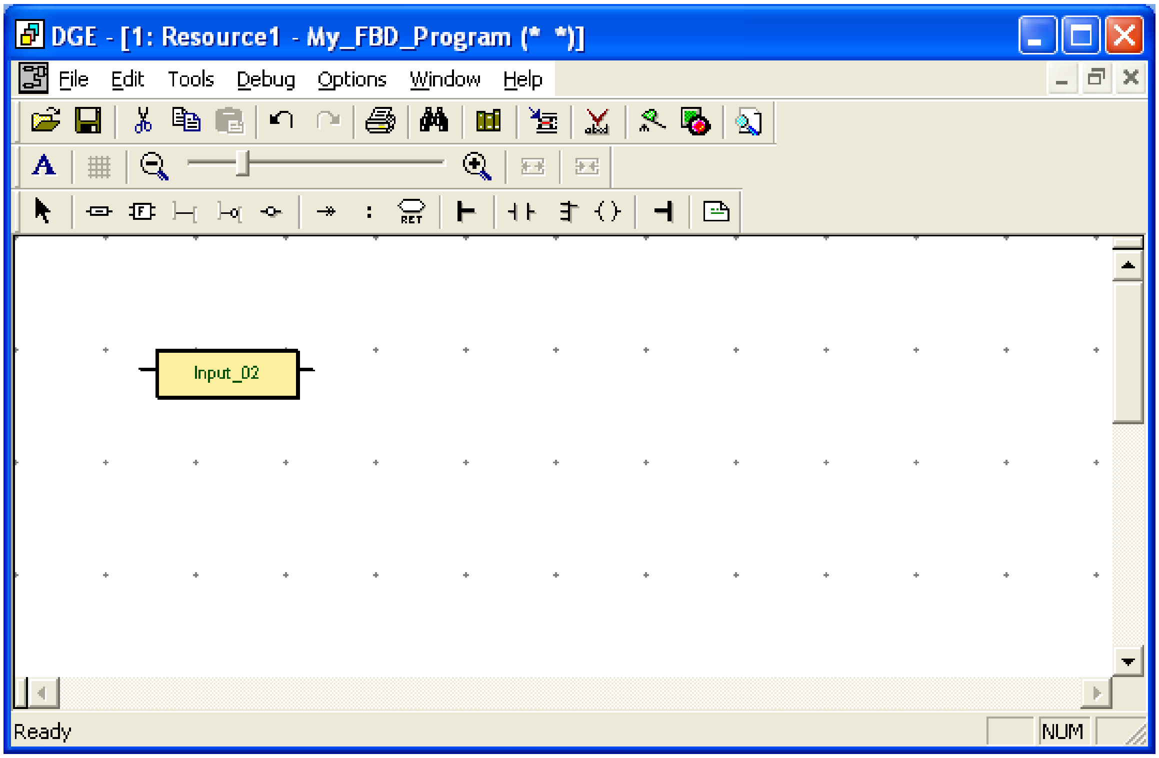
The New Variable dialog box gives you more choices. You can assign an alias to your variable. This is usually not needed, so we will leave it blank. You may enter a comment if you like. You may also leave the Comment field blank or enter a Comment later. Click OK.

Figure 3-37 New Variable



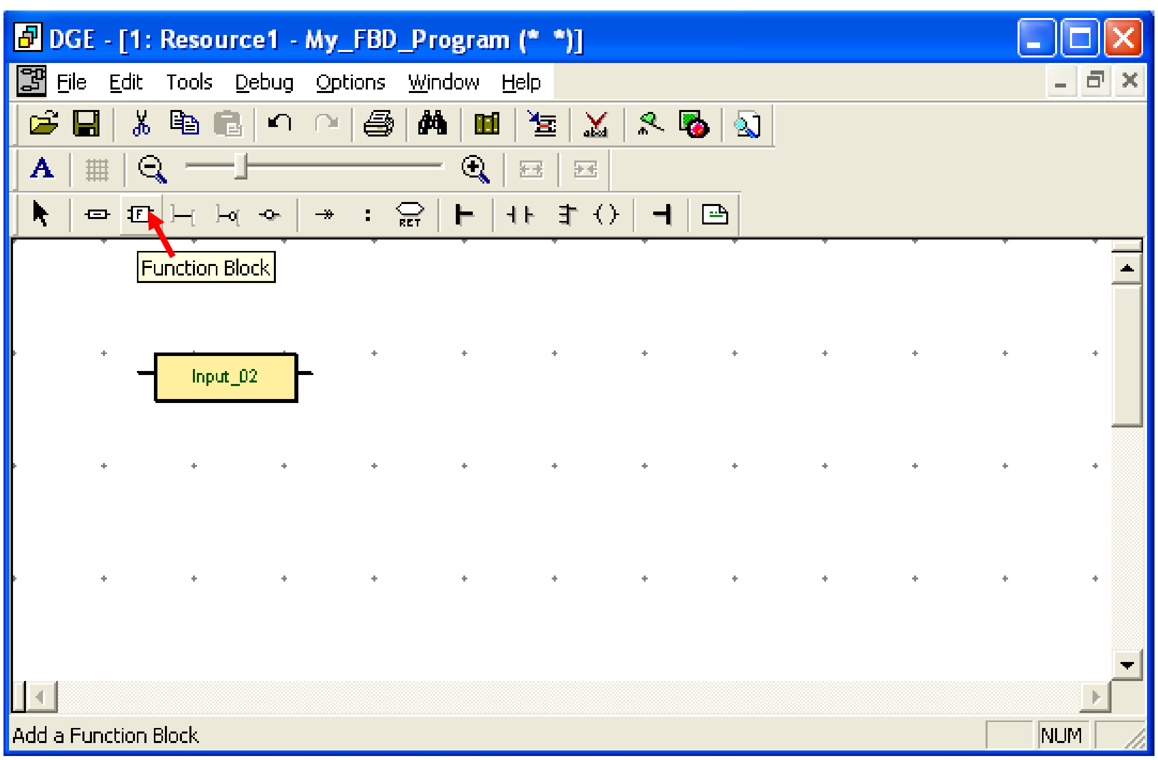
The program has created an input variable, as shown below.

Figure 3-38 FBD Workspace



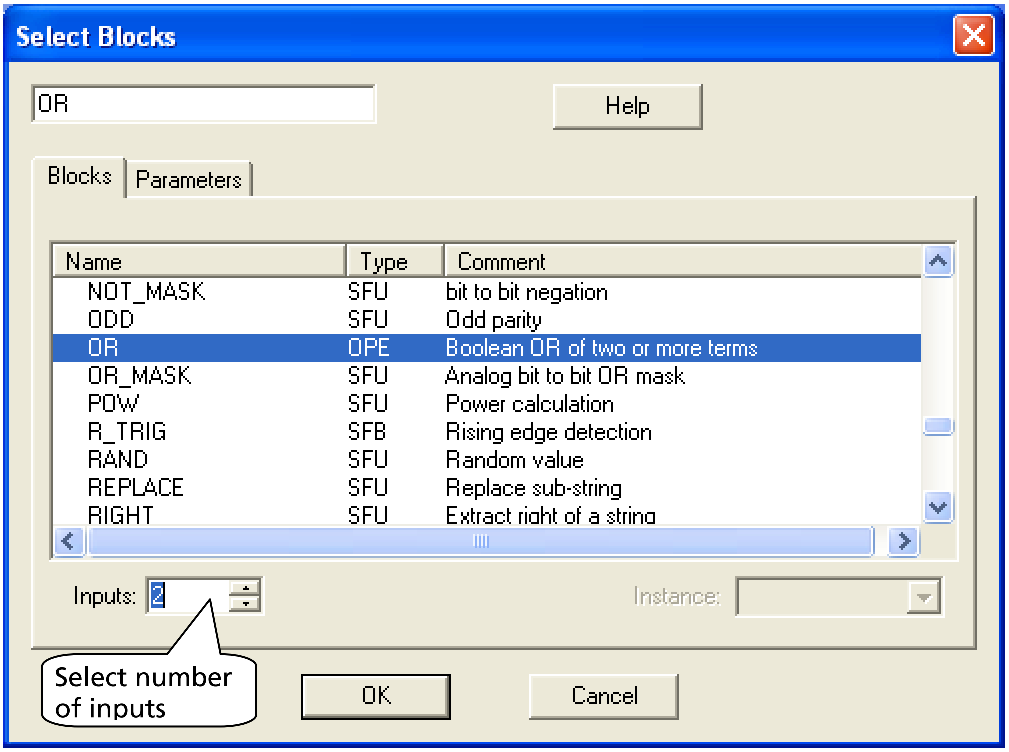
The next step is to insert a function block that acts upon our variable. For the example, click on the Function Block icon as shown below.

Figure 3-39 Creating a Function Block



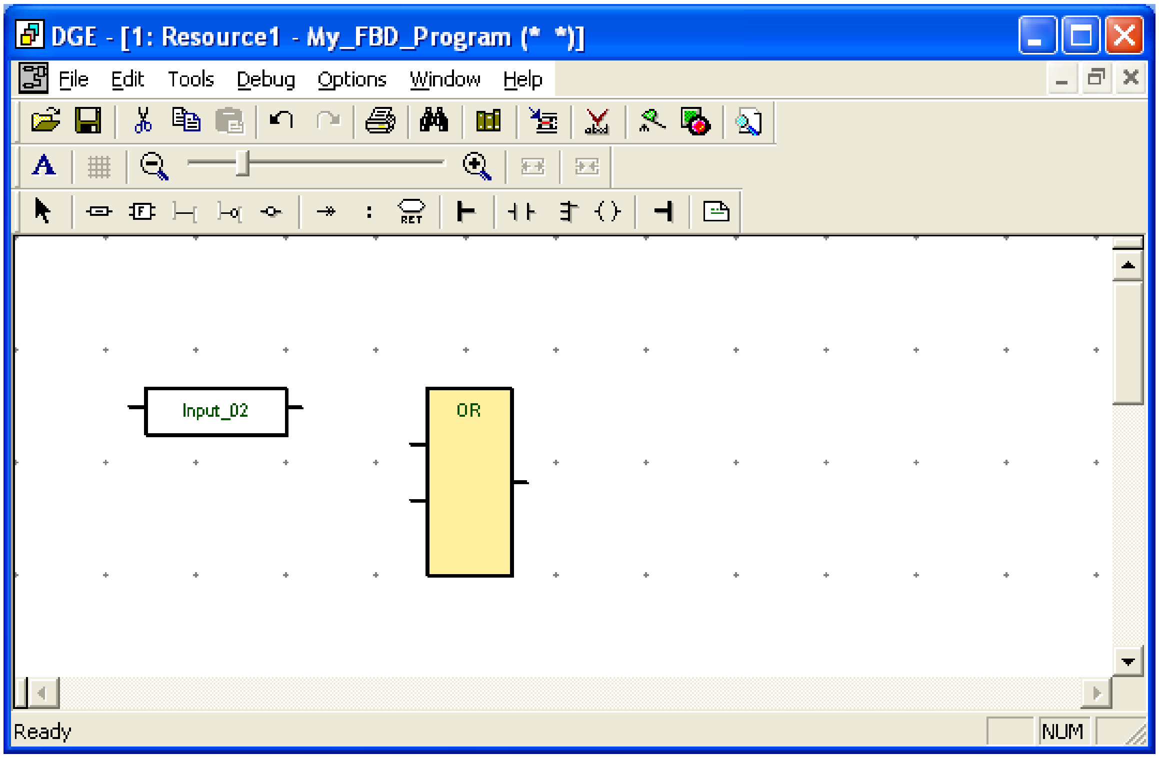
When you drag the Function Block to a place just to the right of the variable, then click again to set it down, you will get a "Select Block" dialog as shown below. For this example, we will select an OR gate as shown. Notice that you can choose the number of inputs. The default is two. The Parameters tab is not used in this case.

Figure 3-40 Selecting a Function Block



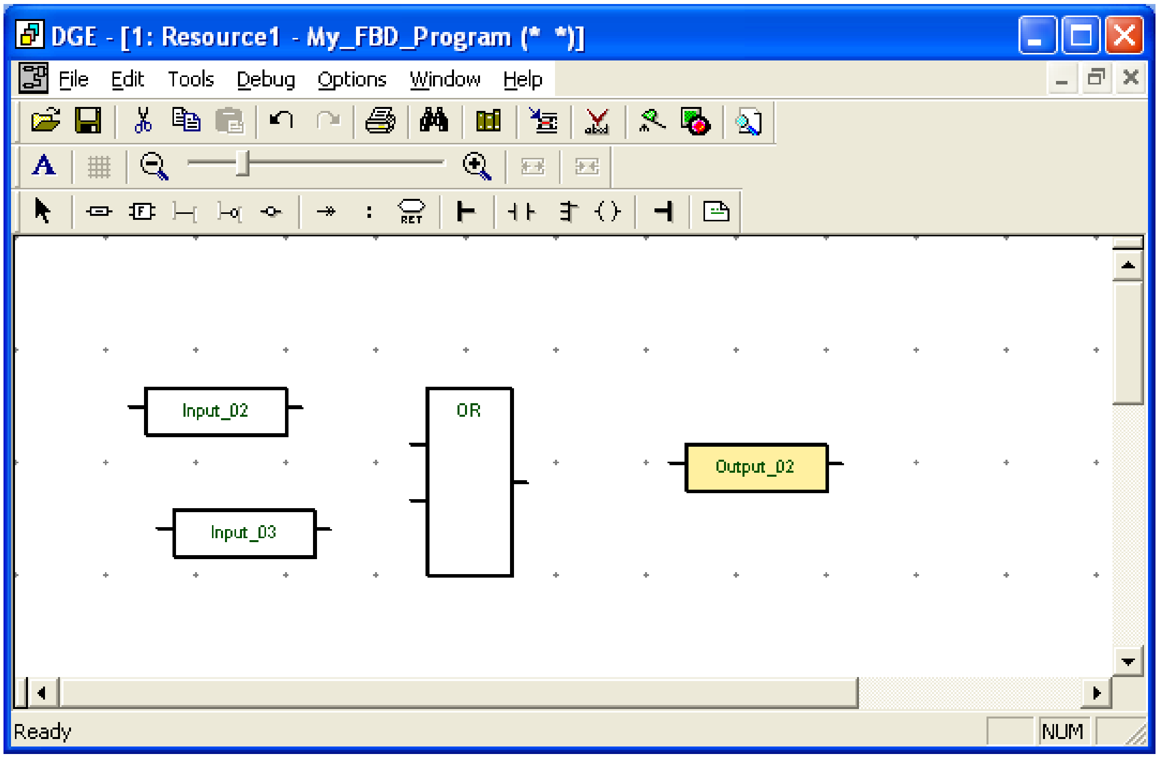
Below is the result of adding an OR Function Block.

Figure 3-41 The OR Function Block



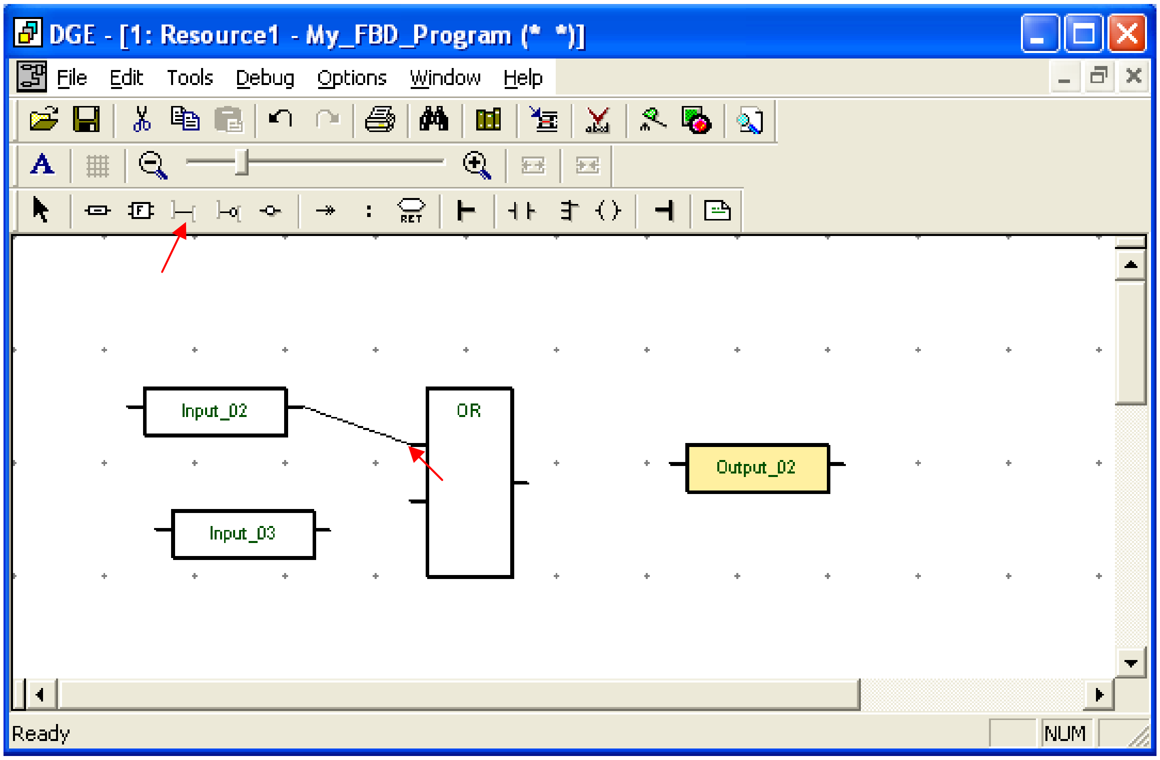
Using the same process in which we added the first input variable, add another input variable and an output variable. Your result should be as shown below.

Figure 3-42 All Variables and Function Block in Place



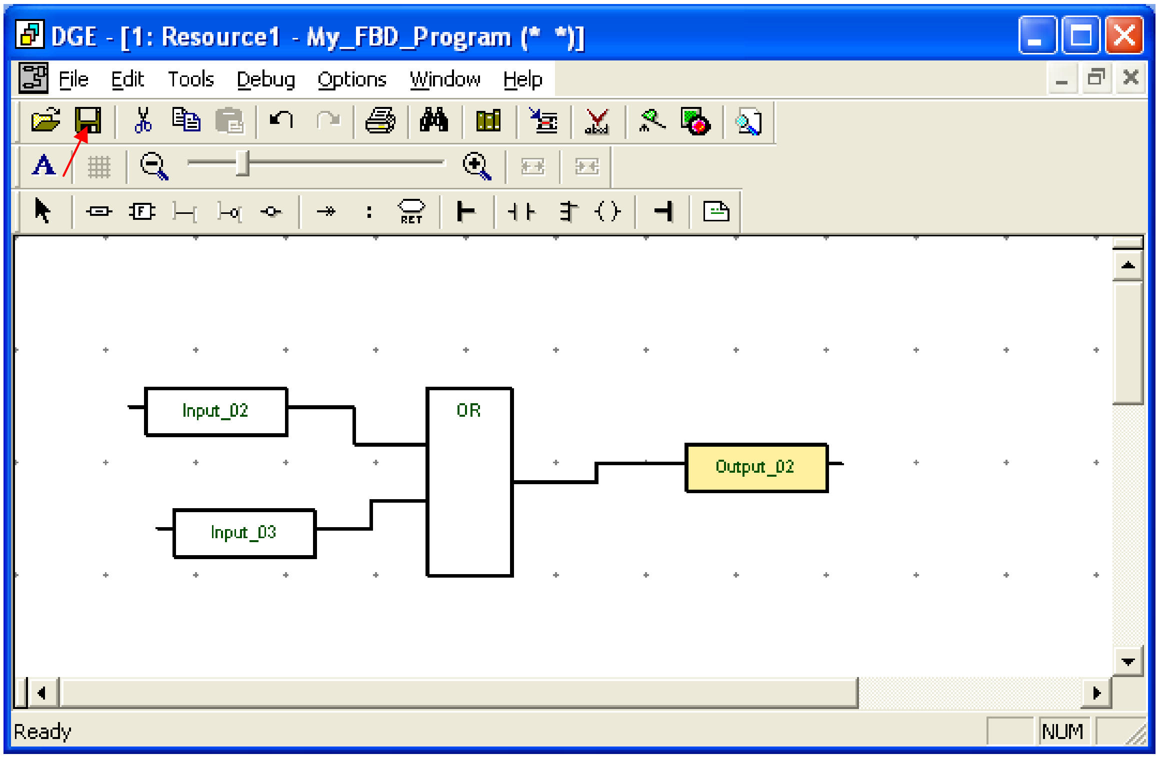
Connect the variables and the function block using the “Draw Link” tool as shown below.

Figure 3-43 Connecting the Blocks



The end result should look as below.

Figure 3-44 The Blocks Connected



Save your program by clicking on the icon shown above. Close the FBD program by clicking on the top-right X.

3.2.6 Changing Variable Attributes

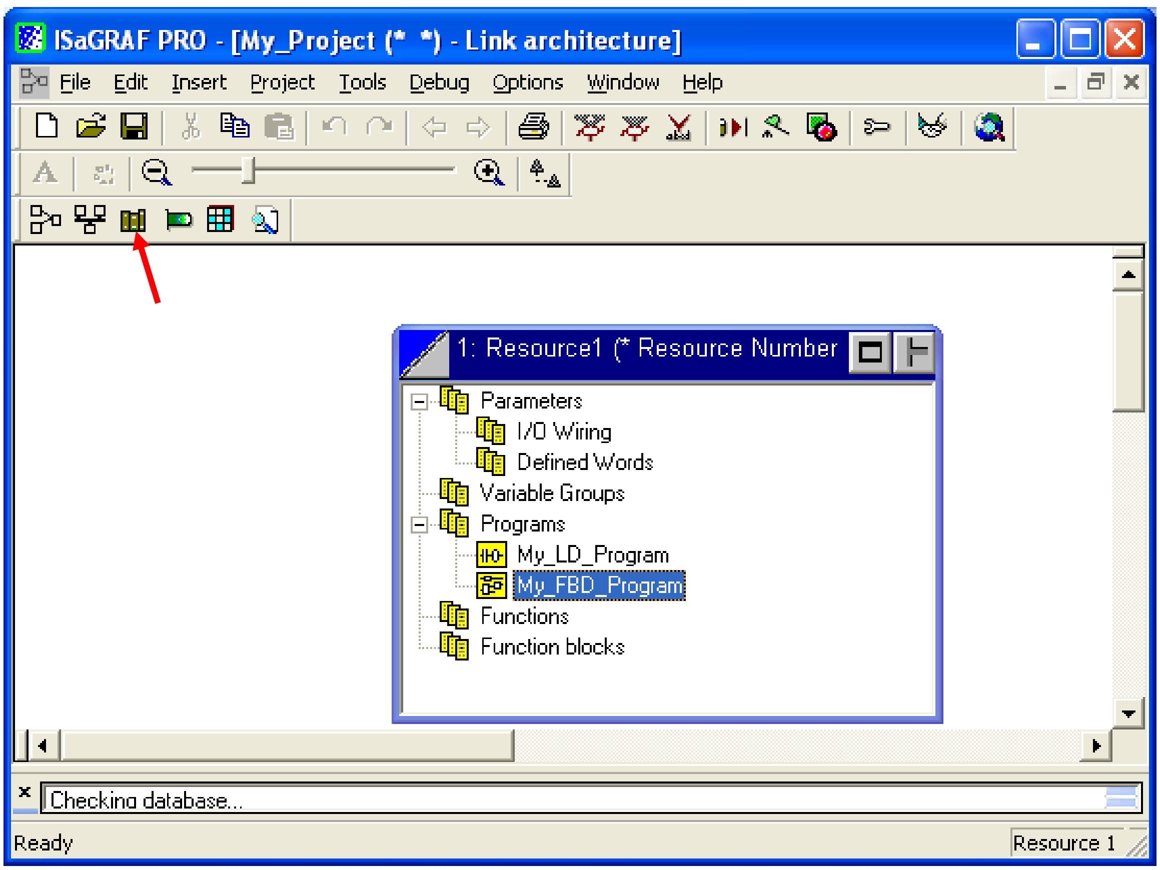
By default, variables are assigned a "Direction" of "Internal" and an "Attribute" of "Free". "Internal" and "Free" allow the variables to work just fine as stand-alone programs running only within ISaGRAF Workbench, but we eventually want our variables connected to drivers ("wired") so that they can be downloaded to do real work on an RTU. Therefore, our variables must have certain qualities that match their intended purpose.

For instance, we would give a variable that has been assigned to an input status switch a "Direction" of "Input" with an "Attribute" of "Read". We would give an output actuator variable a "Direction" of "Output" and an "Attribute" of "Write".

The following instructions illustrate how this is done.

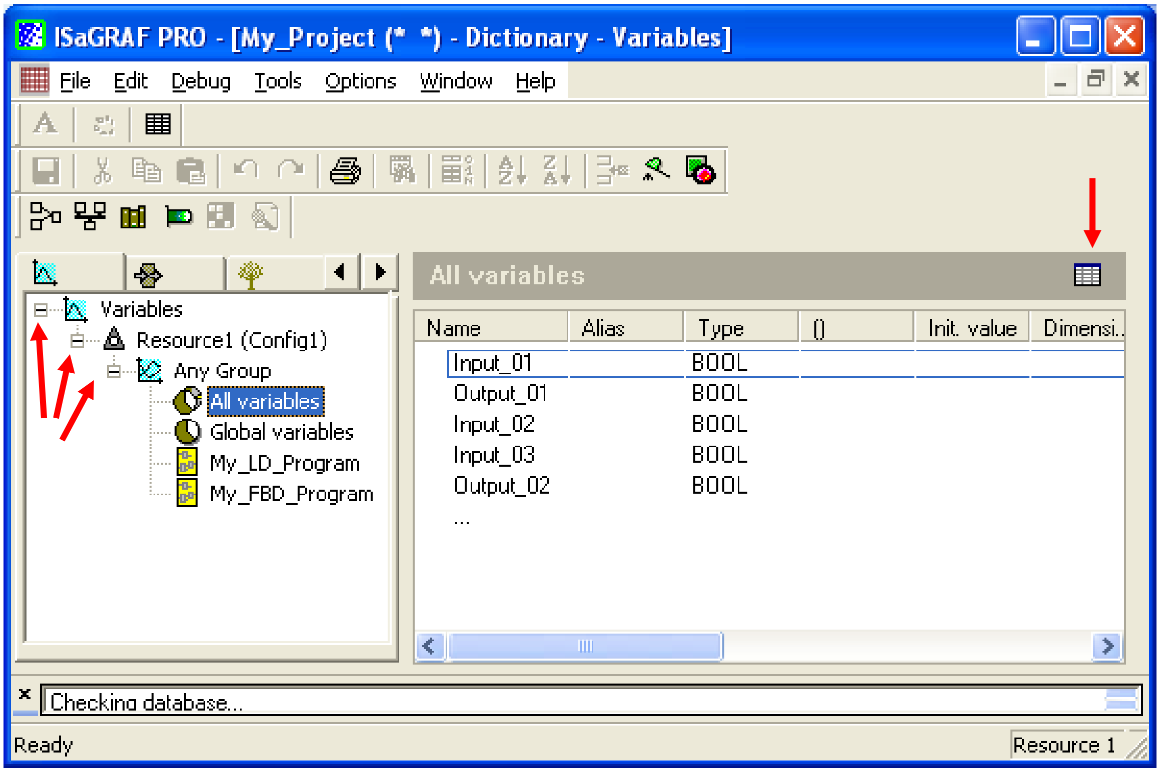
Back at the Link Architecture view, click on the Dictionary icon shown by the arrow below.

Figure 3-45 Selecting the Dictionary



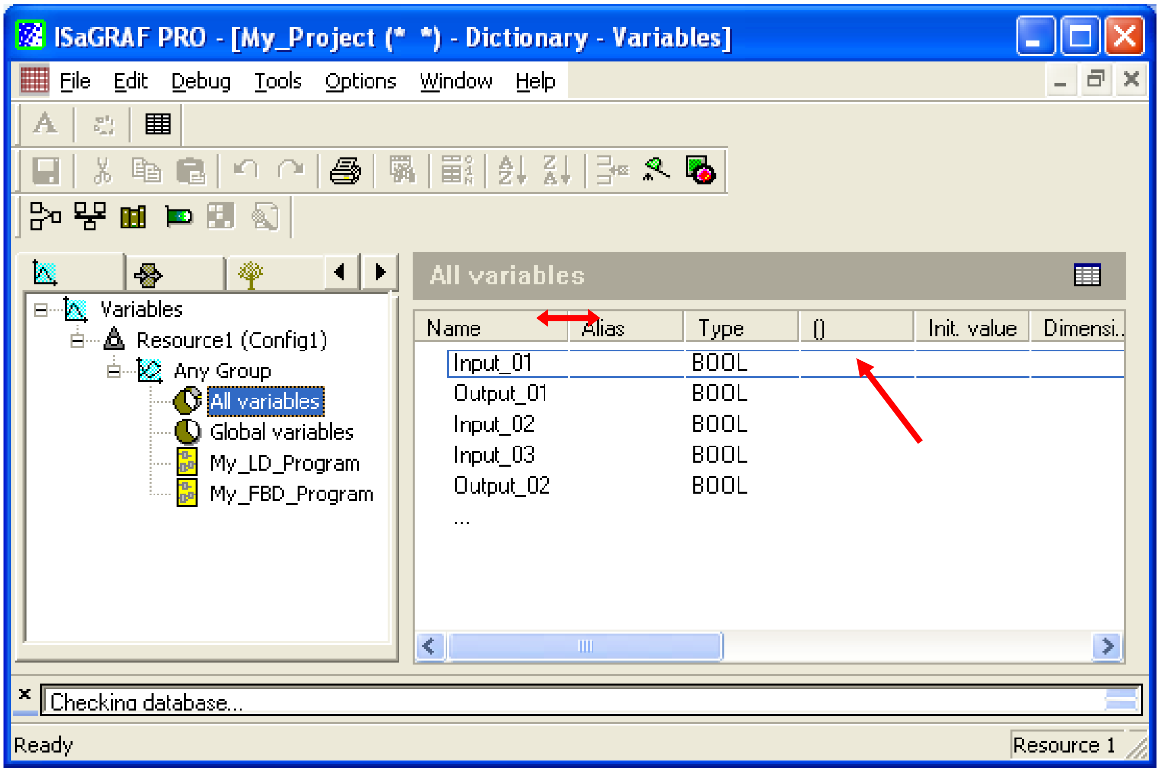
Expand the variables tree (shown in arrows below-left) to All variables. You must see the variable list exactly as shown. If the list is not exactly as shown, click on the icon shown by the downward arrow on the right.

Figure 3-46 Expanding the Variables Tree



You may expand the column width to see the full name of the variables (see double-headed arrow below). Double-click in the blue outlined variable row as shown by the large arrow below.

Figure 3-47 Opening a Variable Attribute Box



The default functions for our Input variables includes “Direction: Internal” and “Attribute: Free”, as shown in Figure 3-48. These functions must be changed to “Direction: Input” and “Attribute: Read” as shown in Figure 3-49.

Figure 3-48 Variable Attribute Box - Default

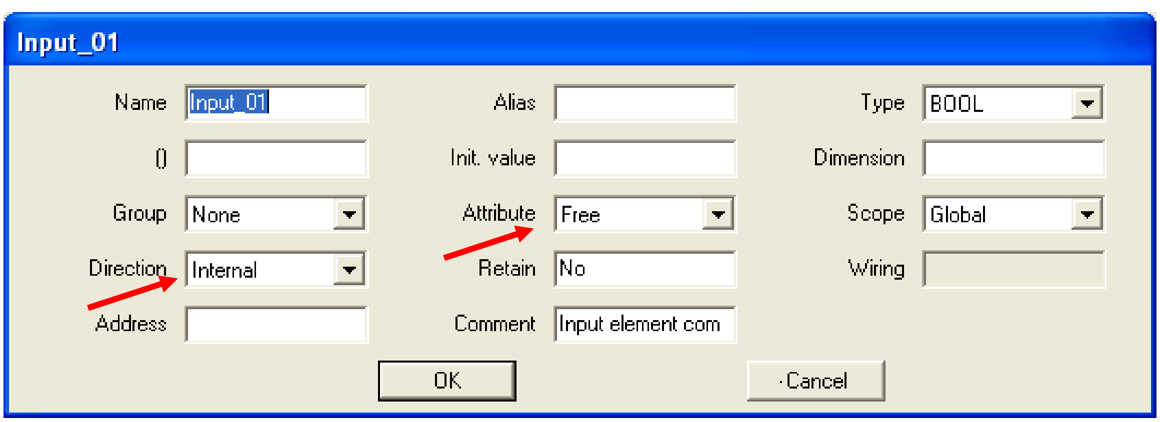
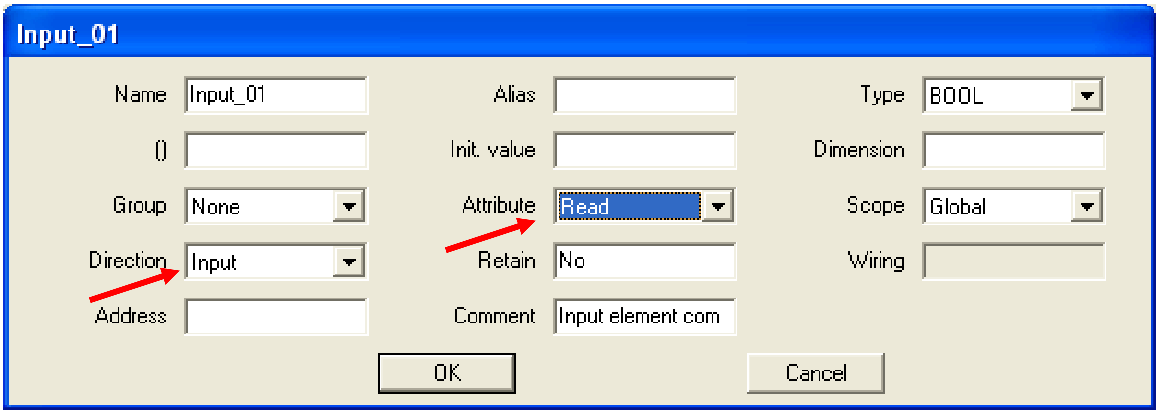
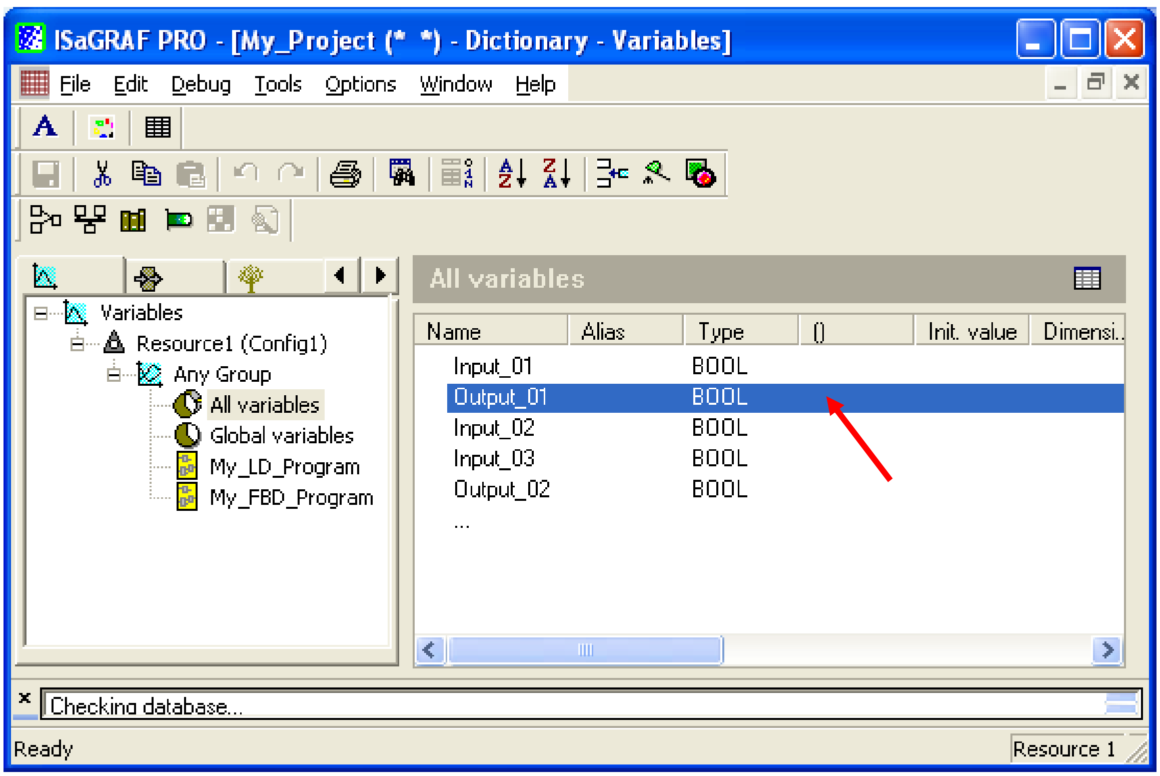


Figure 3-49 Variable Attribute Box – Input Variable



Click OK. Change the other input variables to the above attributes.

Now the Output variables attributes must be changed. Double-click on the Output_01 variable row.



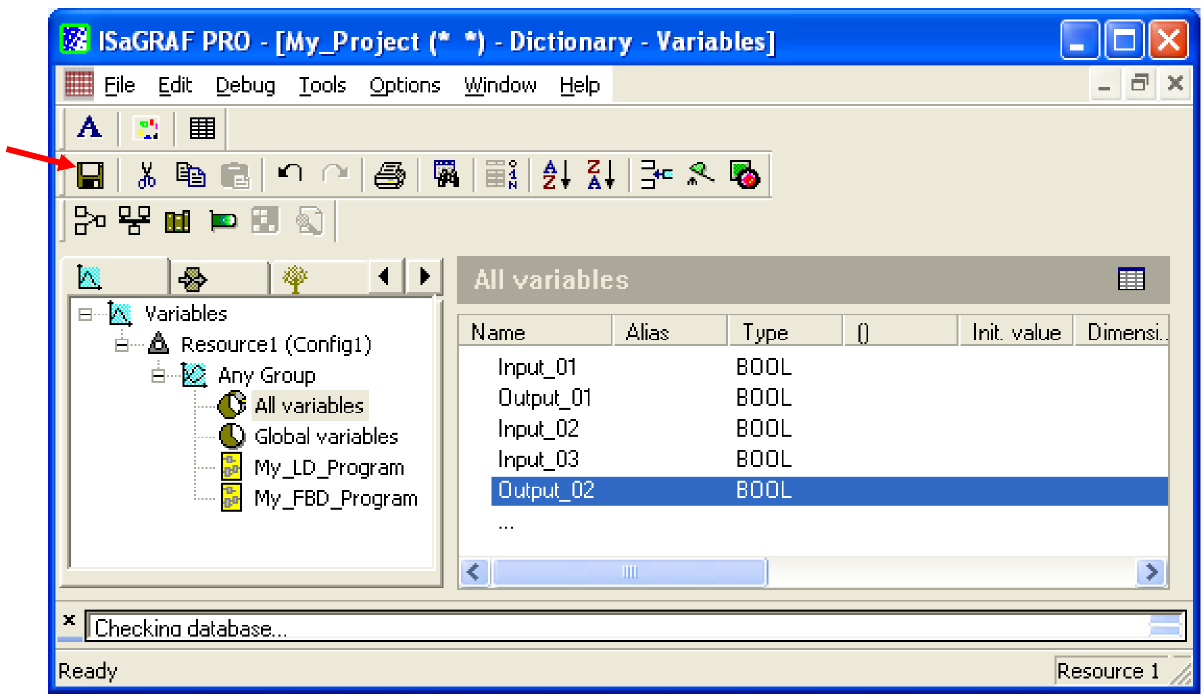
Change the Direction to Output and the Attribute to Write as shown below. Click OK.

Figure 3-50 Changing the Output Variables Attributes

Repeat this process for Output_02.

Save the new variable attributes. See arrow at left below.

Figure 3-51 Saving the New Variable Attributes

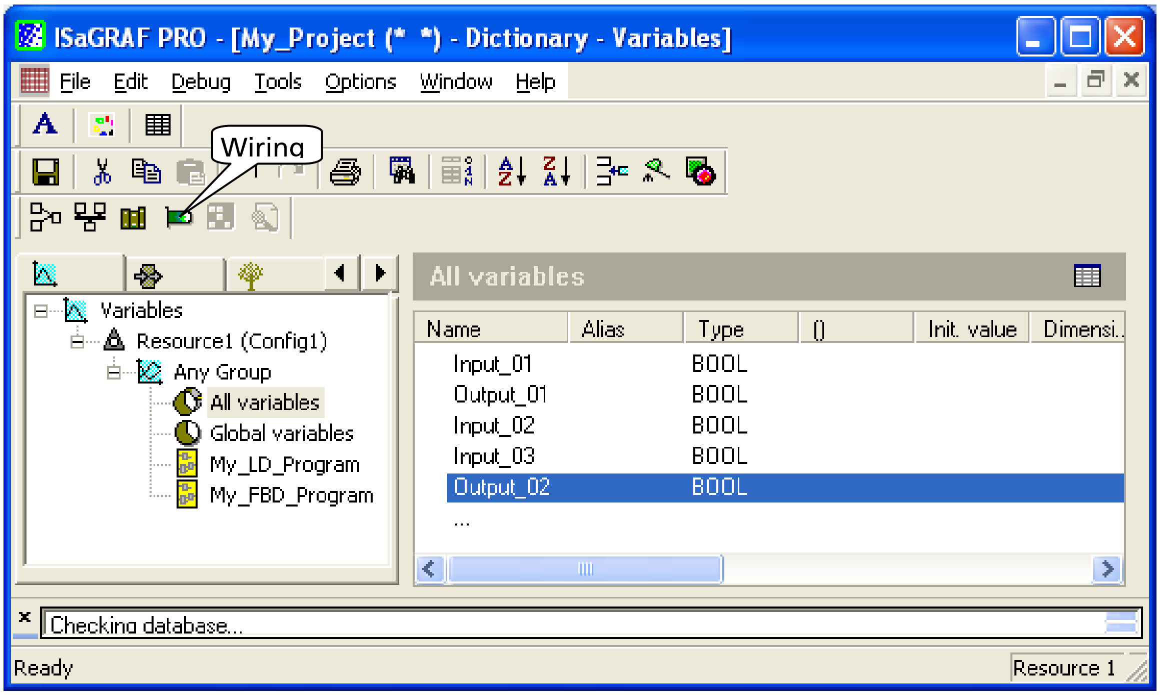


3.2.7 "Wiring" Input/Output Points

"Wiring" refers to the process of assigning the proper drivers to variables so that they will work in the real world of the RTU. The variables you will wire must first have the proper attributes as detailed in the previous section.

Click on the Wiring icon as shown below.

Figure 3-52 Beginning the Wiring Process

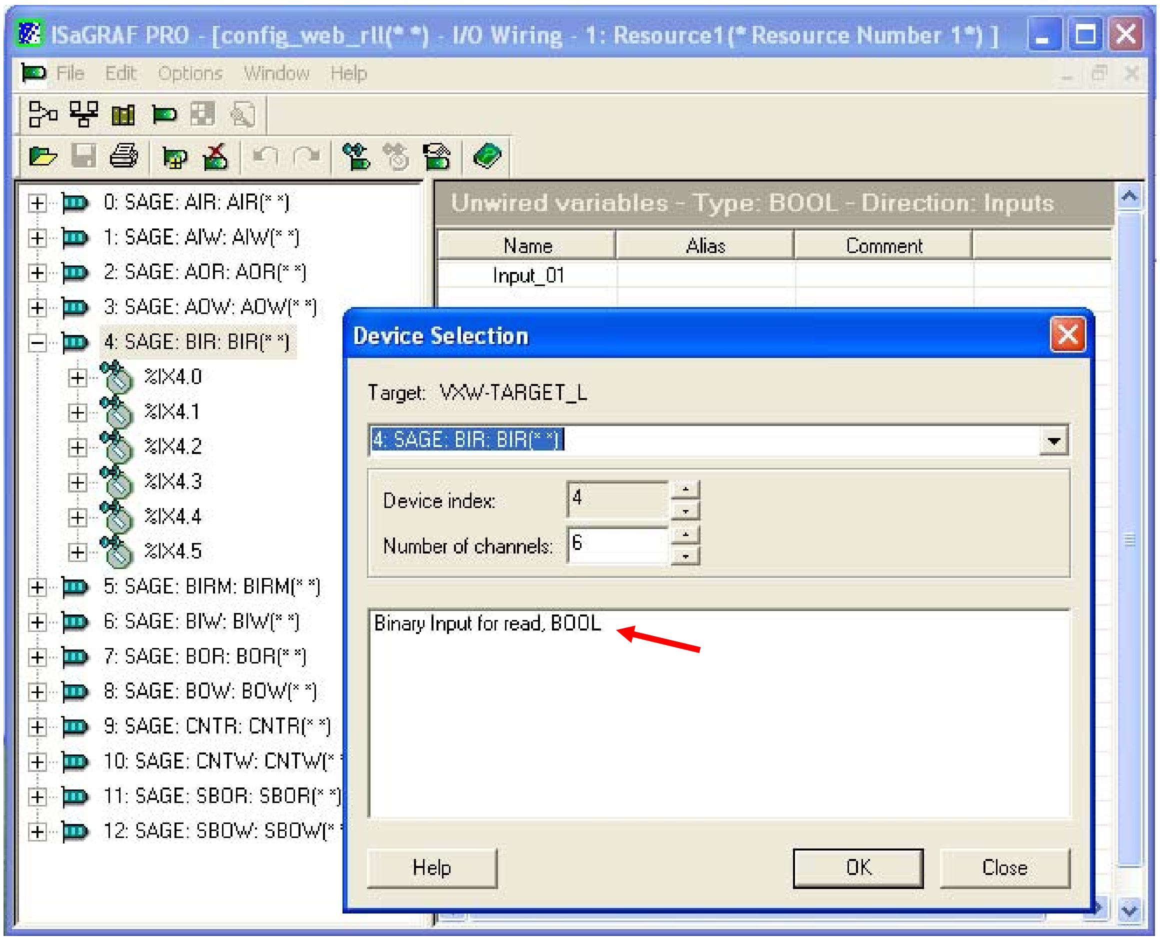


Acronyms in the wiring list have meanings for our variables. For instance, AIR means Analog Input Read. BIR means Binary Input Read. You can see the exact meaning for any of the items in the list when you double click the item as shown in Figure 3-53. The explanation is indicated by the arrow in the figure. Click OK to dismiss the Drive Selection box.

The meanings of the driver acronyms are also spelled out in section “3.4 Drivers”.

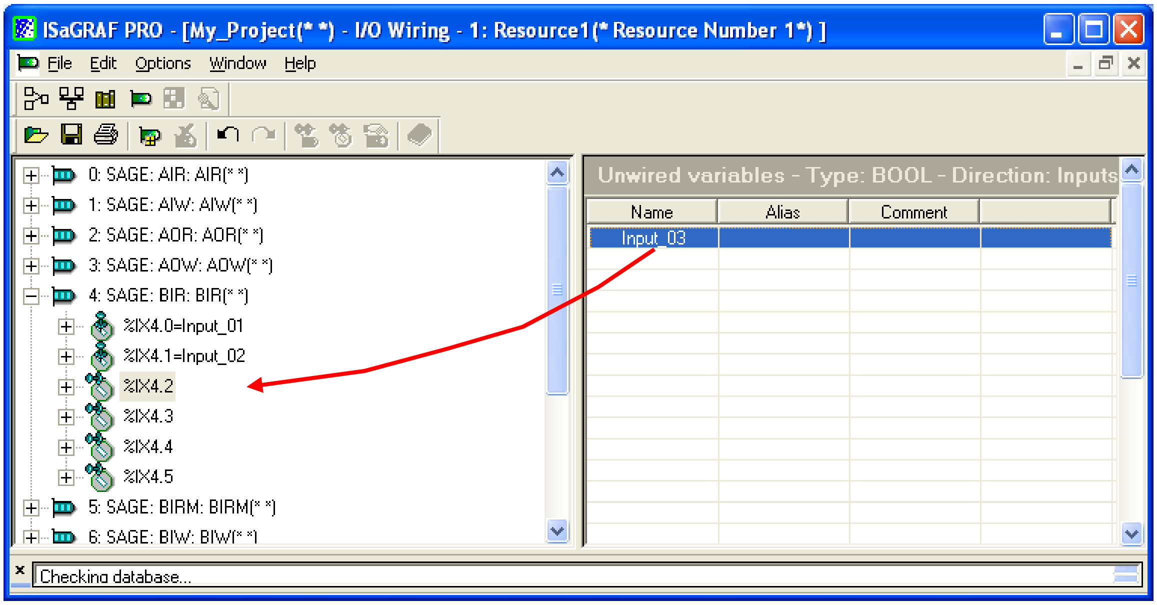
Expand and select the SAGE: BIR wiring list as shown below. Notice that the Input variable appears in this list.

Figure 3-53 The SAGE: BIR Wiring List



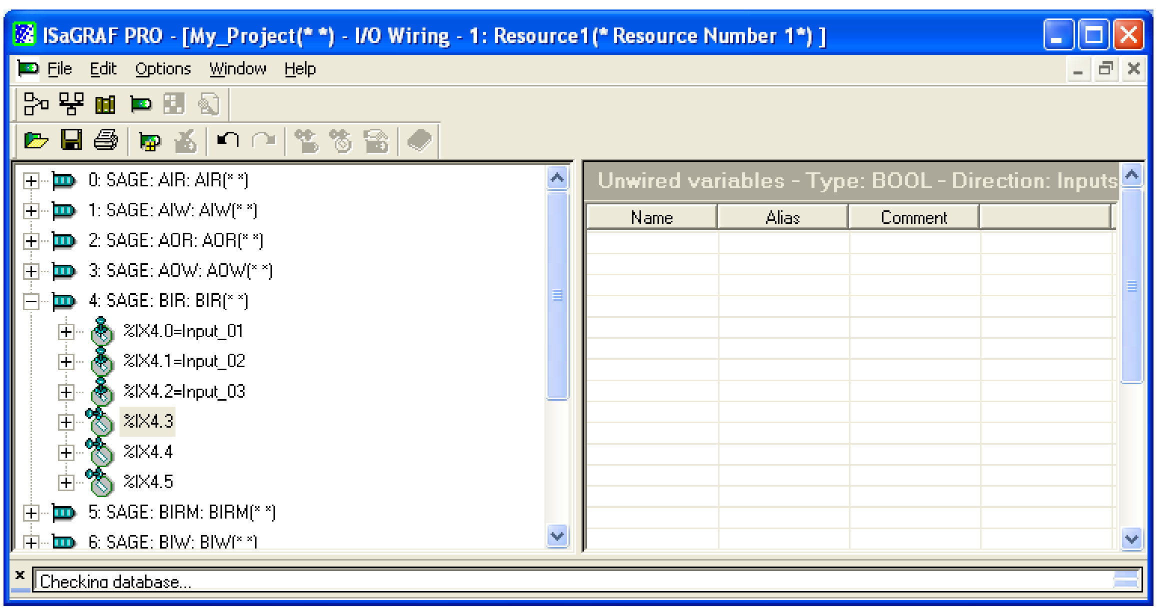
The input variables must now be associated with the wiring drivers. Drag and drop the input variables to the first available slots under SAGE: BIR as shown below. BIR means Binary Input Read. This type of driver corresponds to the attributes we assigned to these variable.

Figure 3-54 Connecting Input Variable to Wiring Driver



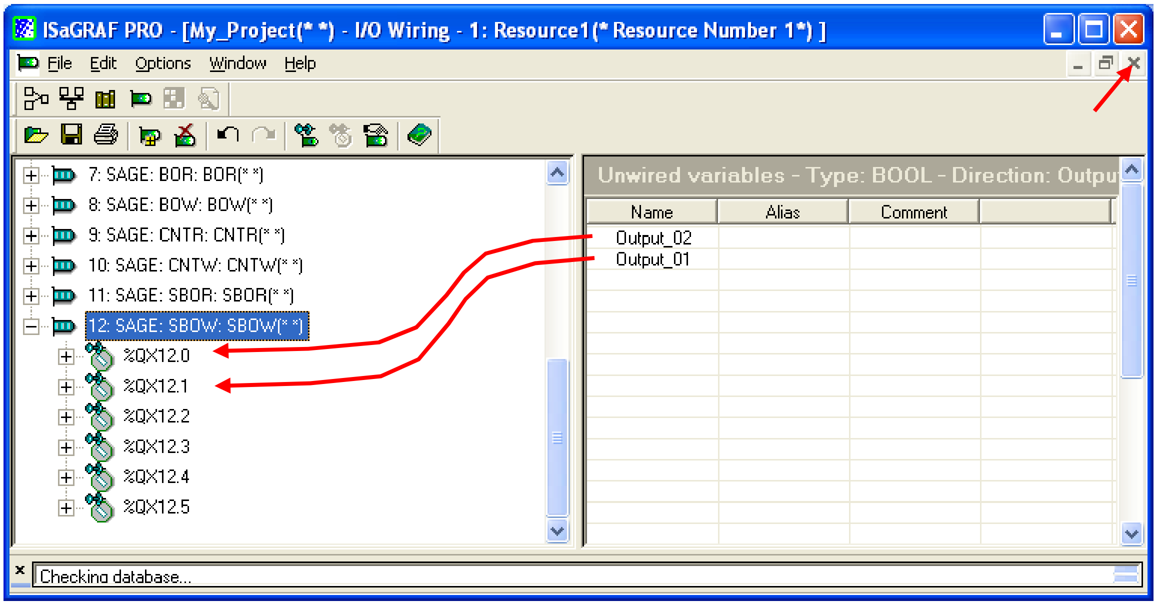
The result is as shown below.

Figure 3-55 The Wired Input Variables



Repeat this process with the output variables. In this case, the wiring point is SAGE: SBOW as shown below. SBOW means SBO for write. This type of driver corresponds to the attributes we assigned to this variable.

Figure 3-56 Connecting Input Variable to Wiring Driver



Save after wiring your variables, then close the I/O Wiring function by clicking on the X at upper right. Don't click the top X, for this would close the ISaGRAF program.

Note: If you accidentally click the top X (which is easy to do), the ISaGRAF program will close. This is not a disaster. If you didn't save, it will warn you to save before closing. Simply reopen the program and navigate to this screen to continue.

3.2.8 Compiling the Project and Downloading to the RTU

In the Link Architecture view:

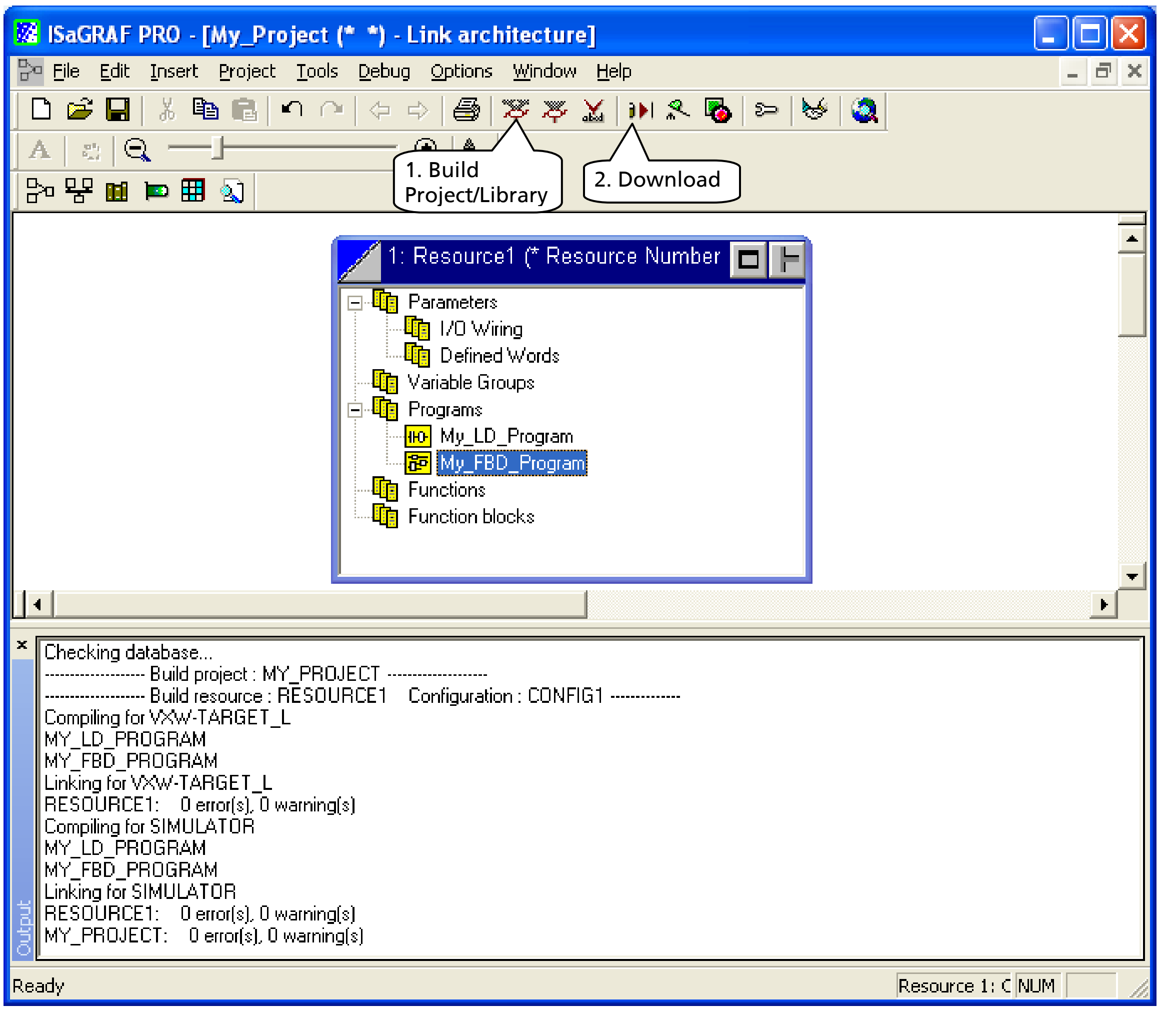
1. Click the icon for building the Project/Library as shown below. This action compiles your program and reports errors, if any.

Note: The two icons to the right of Build Project Library are Build Resource and Build Program. Normally, you will never use these functions. Any time you make a change in the program or any changes in the project, always select Build Project/Library. This insures that all changes will be compiled.

2. Download. This action begins the download process.

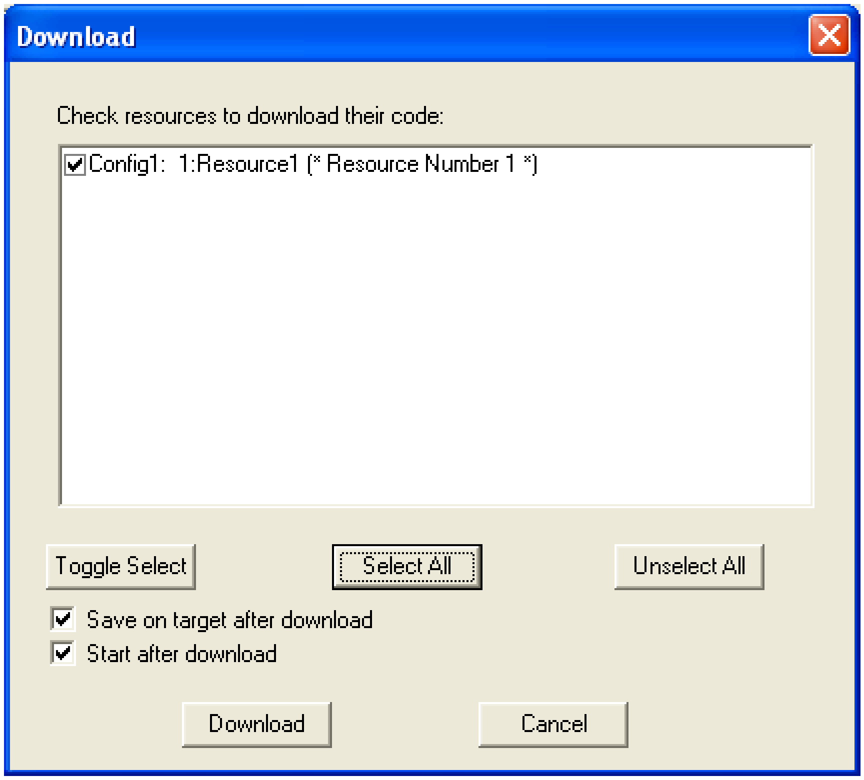
Note: Your PC must be connected to the RTU by Ethernet (TCP/IP).

Figure 3-57 Building the Project/Library



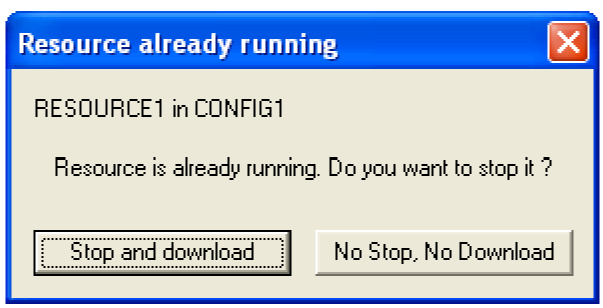
When you click the download icon, you get the dialog box shown below. Click Select All, then click Download.

Figure 3-58 Download Dialog Box



If there is already a config@WEB RLL program running in the RTU, you will get the following warning. Click Stop and download.

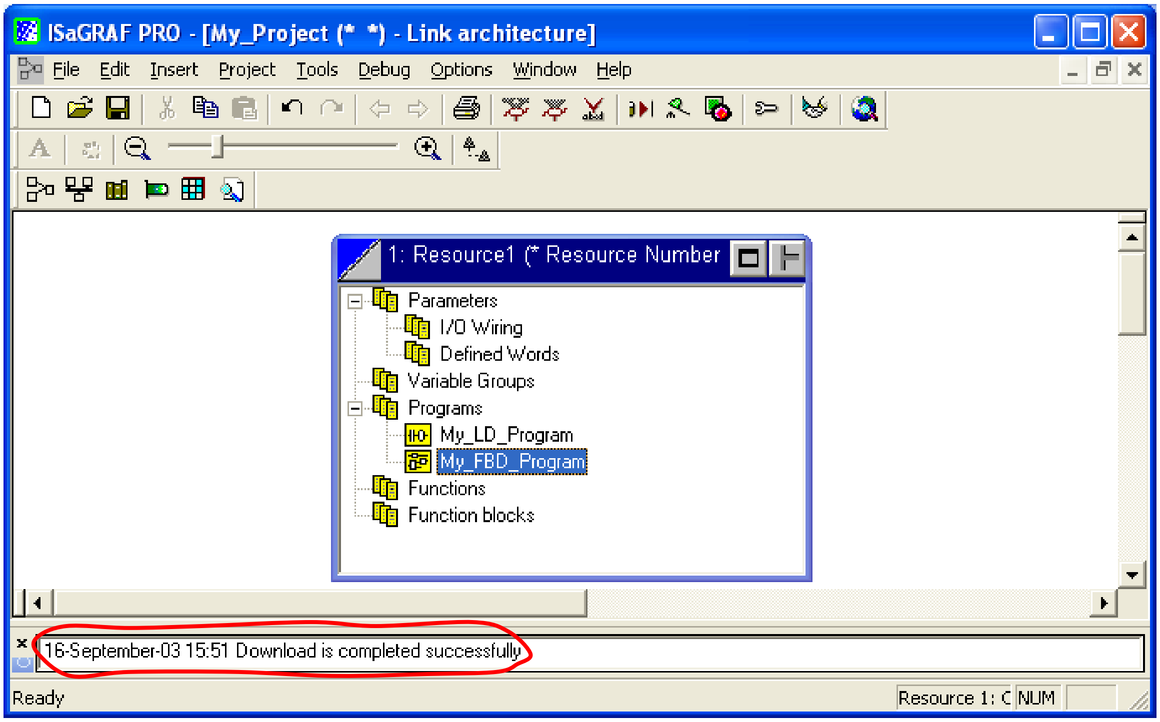
Figure 3-59 Resource Already Running



Note: Depending on your PC OS, the above message could have a different format. For 98SE, click Continue.

After the download completes successfully, you will get the following message.

Figure 3-60 Download Completed Successfully

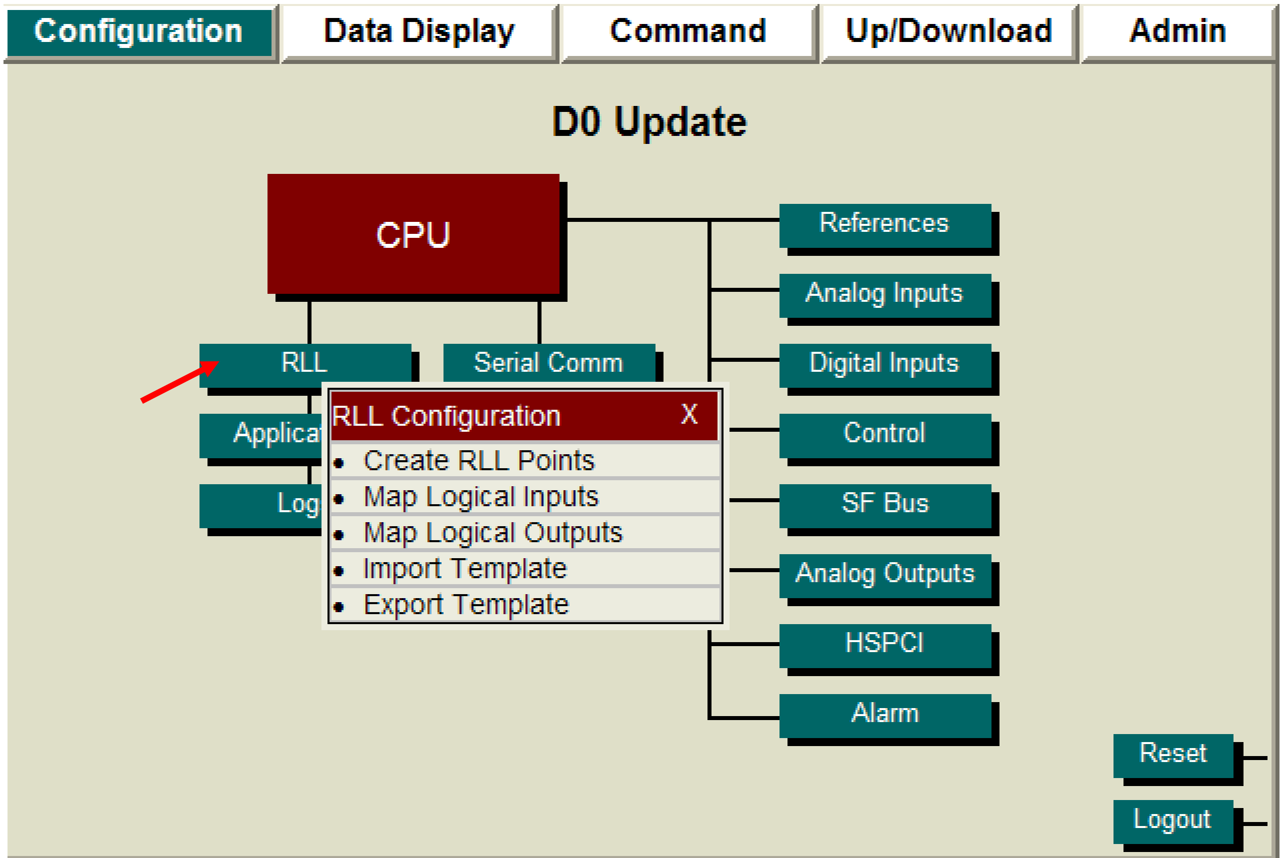


Note: Leave ISaGRAF running as you go through the next section on setting up RTU points. You will use ISaGRAF in the sections on Debugging and Simulation.

3.2.9 Configuring the RTU for RLL

Open IE with the config@WEB interface. Make sure that status inputs and SBO outputs are assigned in hardware (right side). Click on Configuration – RLL as shown. You must map points in the RTU that correspond to the simple LD program.

Figure 3-61 Configuration – RLL



Create RLL Points

This function creates pseudo points.

Map Logical Inputs

This function maps input points corresponding to input points in the RLL program

Map Logical Outputs

This function maps Output points corresponding to output points in the RLL program

Import Template

See the section on Import/Export Templates.

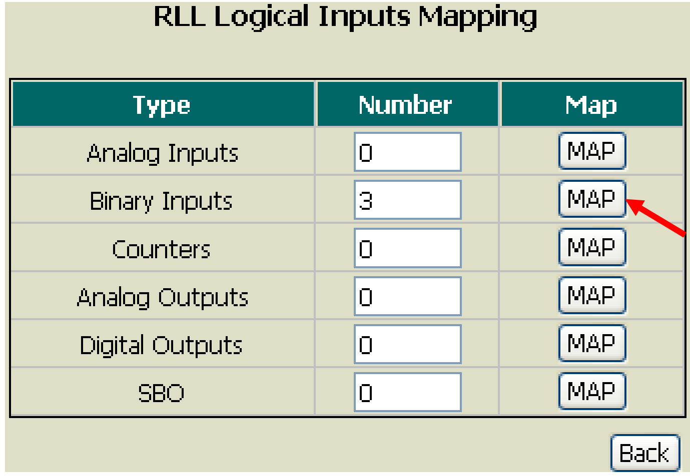
Export Template

See the section on Import/Export Templates.

For our simple RLL programs that we have downloaded to the RTU, we need to map three input points and two output points. (Recall that we created an LD program with one input, one output, and an FBD program with two inputs, one output. We nee to map a total of three inputs and two outputs.) Click Map Logical Inputs.

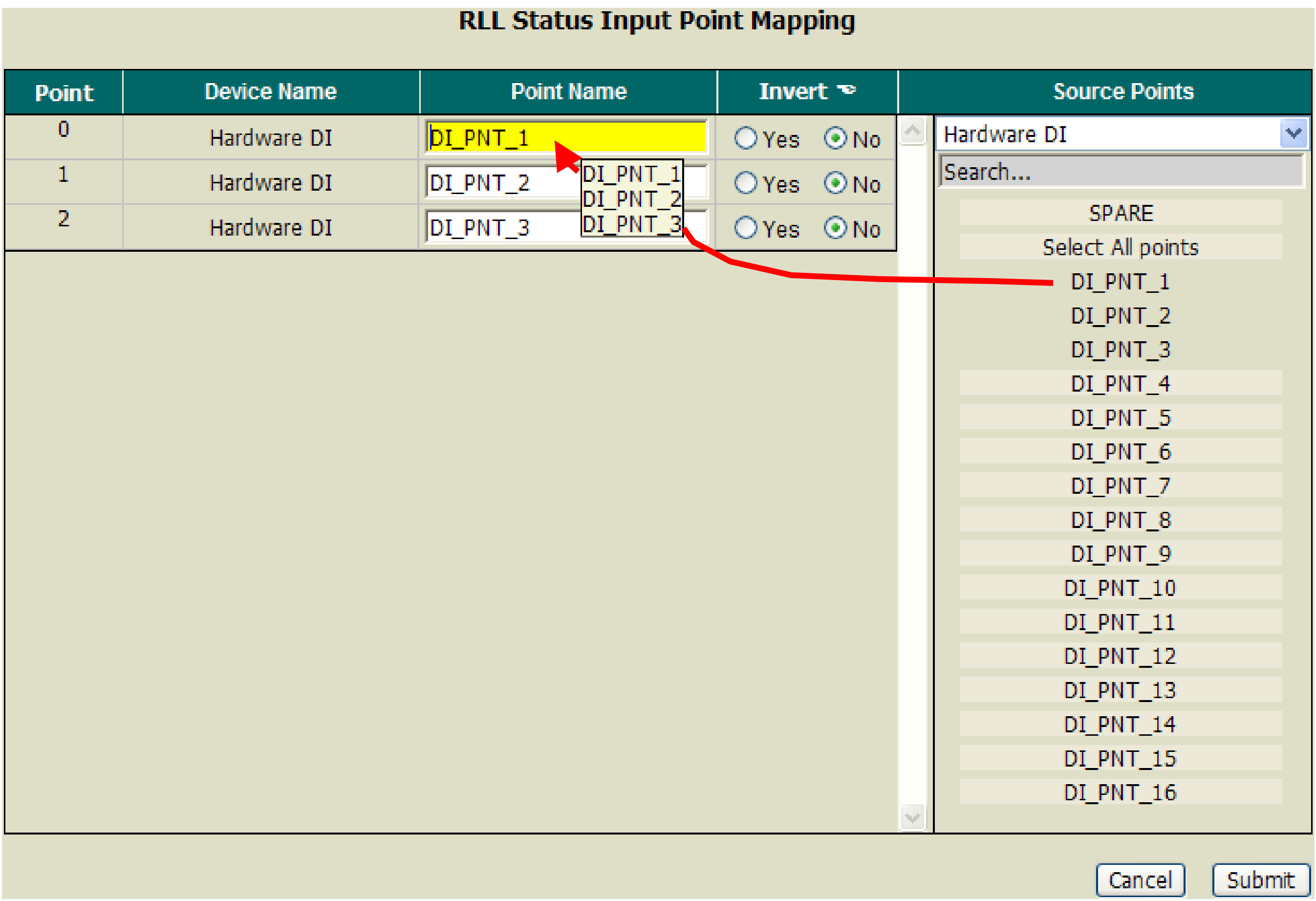
Enter three points for Binary Inputs, then click MAP, as shown below.

Figure 3-62 Mapping the Binary Input Point



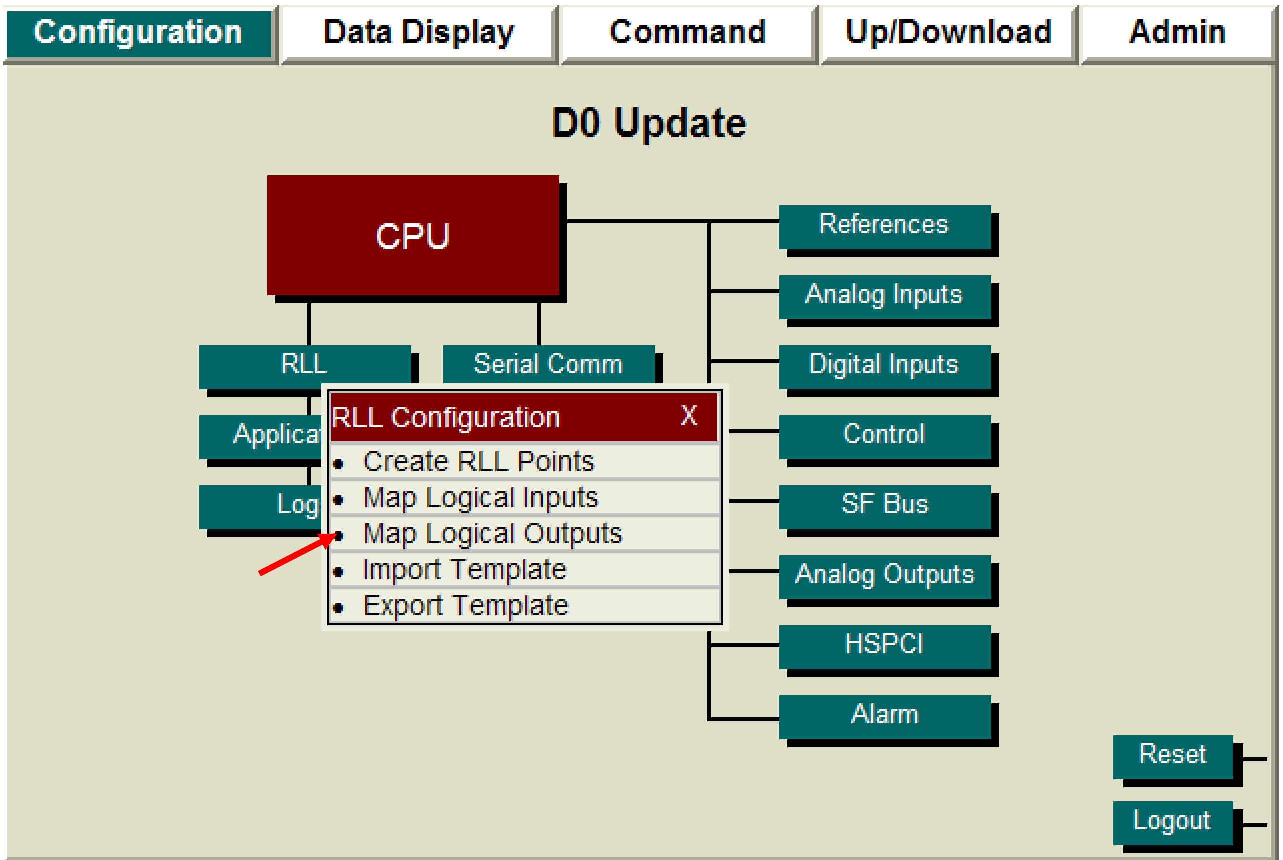
Select the first three hardware points and map them to the first three points as shown.

Figure 3-63 Mapping the Binary Input Points



Submit and return to the Configuration screen. Select Map Logical Outputs from the RLL button as shown.

Figure 3-64 Map Logical Outputs



Enter two points for SBOs as shown, then click MAP.

Figure 3-65 Map One SBO Point

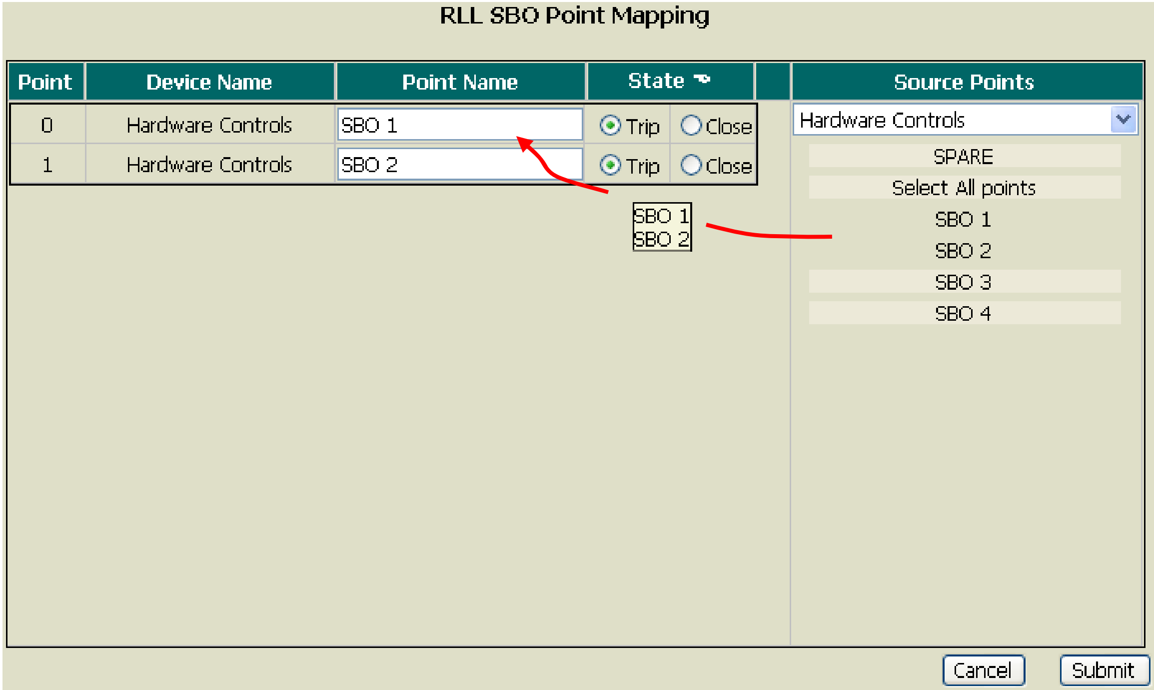
RLL Logical Outputs Mapping

Type	Number	Map
Analog Inputs	0	MAP
Binary Inputs	0	MAP
Counters	0	MAP
Analog Outputs	0	MAP
Digital Outputs	0	MAP
SBO	2	MAP

Back

Map two SBO hardware points as shown.

Figure 3-66 Mapping One SBO Point



Submit and return to the Configuration screen. Reset the RTU.

3.3 Testing Your Programs

Remember, we have two programs running simultaneously. We set them up to use non-interfering status points and SBOs.

LD Program

Exercise status point 1. SBO 1 Trip should engage repeatedly as long as the status point is closed.

FBD Program

Exercise either status point 2 (OR) status point 3. Either status point, or both, should Trip SBO 2.

3.3.1 Multiple Programs in the RTU

As demonstrated above, ISaGRAF for config@WEB can support multiple programs running simultaneously in the RTU. This gives the user great freedom and great responsibility. If you create and download multiple programs, be sure they either run in harmony with each other, or do not interfere with each other. If you wish to clear the RTU of a particular program, delete the particular program in ISaGRAF Workbench (that is, on your PC), then compile and download the new project (without the offending program). If you wish to clear the RTU of all programs, create an empty project (no programs), compile it, and download it.

3.4 Drivers

3.4.1 Types and Number of Drivers

Drivers allow the program variables to connect to points in the RTU. There are two versions of ISaGRAF: limited number of I/O-points, and unlimited I/O points.

In the Telvent implementation, there are seven possible types of drivers for inputs to logic and six possible types of drivers for outputs from logic. Each type of driver has a maximum number of channels – refer to "Points Supported" in Chapter One. Of course, if you have the limited I/O point version of ISaGRAF, the total number of connections is according to the license. The default config_web_rll template is set for six channels for each type of driver.

Your application probably won't need six each of all the various driver types. For your real-world program, you will probably want to assign only the types of drivers needed in order to conserve driver resources.

Drivers may be deleted (and easily recovered later, if needed) to gain extra driver channels for your particular application. See section 3.4.2 Removing/Adding/Modifying Drivers.

Note: Plan ahead. Determine what driver types and how many channels of each type you will need before you wire your variables to the drivers. Any changes to the wiring (driver changes) will un-wire the variables. While it is easy to re-wire variables to drivers, planning ahead is even easier.

3.4.1.1 Drivers for Inputs to Logic

air (Analog Input Read)

This function is used to read analog input data (RTU analog inputs).
32 bit input value scaled by information entered when point is mapped.

aor (Analog Output Read)

This function is used to read analog output data (RTU analog outputs).
32 bit input value scaled by information entered when point is mapped.

bir (Binary Input Read)

This function is used to read binary inputs (RTU status inputs).
Boolean input represents current state of the input.

birm (Binary Input Read MCD)

This function is used to read change of state data that happens at least once during the ISaGRAF cycle time from binary inputs. For instance, one could use birm to read the change of state of a recloser that cycled faster than the ISaGRAF cycle time. The default cycle time of ISaGRAF is 1 second, but this time can be adjusted down to approximately 200 milliseconds, depending on the complexity of your program.

1. True only for 1 cycle.
2. Any rising-edge change of state sets the bit.

bor (Binary Output Read)

This function is used to read binary outputs (RTU digital outputs).
Boolean input represents that at least one rising-edge change of state has occurred.

cntr (Counter Input Read)

This function is used to read counter inputs (RTU accumulator inputs).
32 bit counters.

sbor (SBO Read)

This function is used to read SBO input values from points allocated by the ISaGRAF package. The values allowable are Trip and Close.

1. Only valid for points that are allocated by ISaGRAF task.
2. True only for 1 cycle.

3.4.1.2 Drivers for Outputs from Logic

aiw (Analog Input Write)

This function is used to write analog input data (RTU analog inputs).
32 bit input value scaled by information entered when point is mapped.
Must write to only points created by logic.

aow (Analog Output Write)

This function is used to write analog output data (RTU analog outputs).
32 bit input value scaled by information entered when point is mapped.

biw (Binary Input Write)

This function is used to write binary inputs (RTU status inputs).
Boolean input represents current state to write to the input.
Must write to only points created by logic.

bow (Binary Output Write)

This function is used to write binary outputs (RTU digital outputs).
Boolean input represents output state.

cntw (Counter Write)

This function is used to write counter inputs (RTU accumulator inputs).
32 bit counters.
Must write to only points created by logic.

sbow (SBO Write)

This function is used to write SBO values.

3.4.2 Removing/Adding/Modifying Drivers

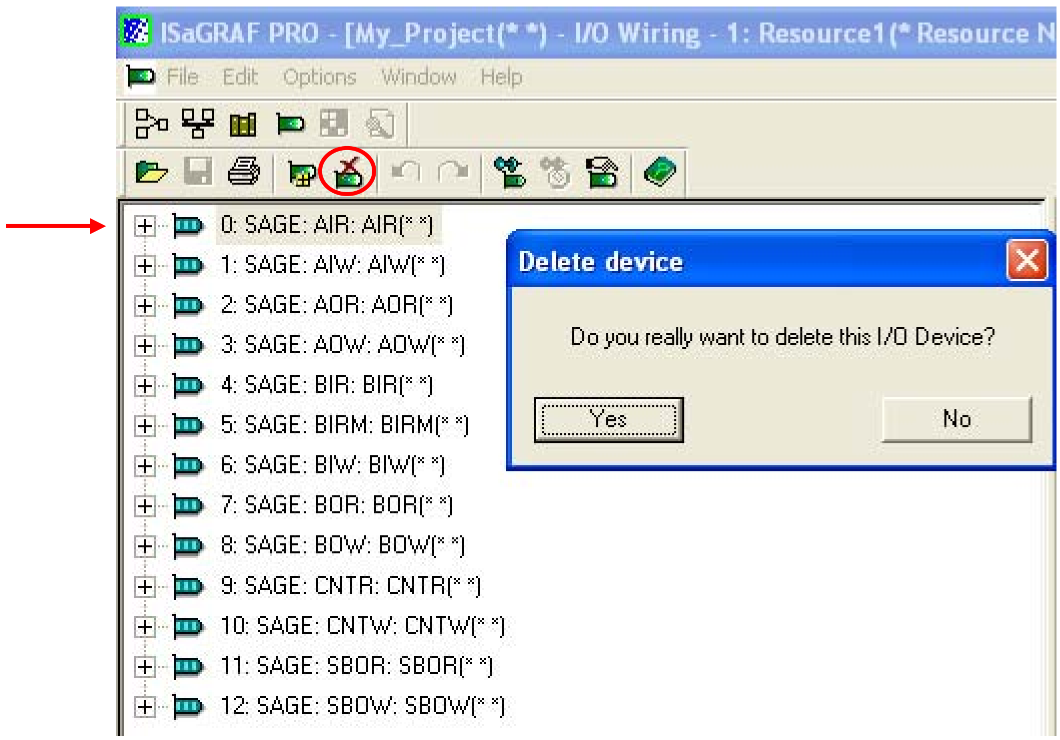
Each of the drivers must occur only one time in the list.

Note: The :"Gain", "Offset", "Direct", or "Conversion" fields are not used in any of the drivers.

3.4.2.1 Removing a Driver

Select the driver (arrow) and click on the Delete Device icon. You will get a warning message. Click Yes. You may later add the deleted driver back in.

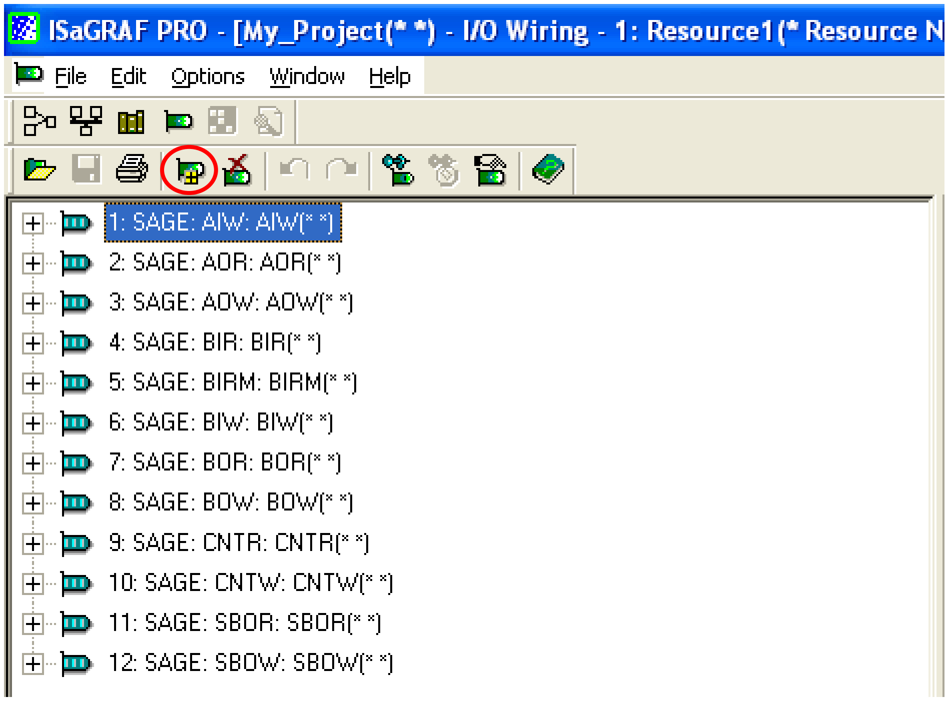
Figure 3-67 Removing a Driver



3.4.2.2 Adding a Driver

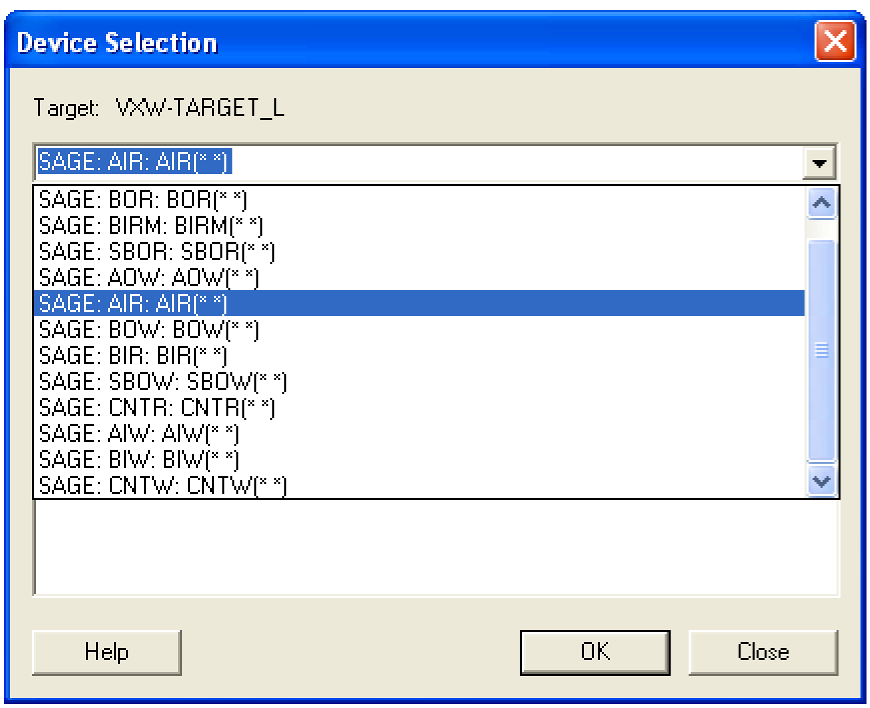
Click on the Add a Driver icon (circled).

Figure 3-68 Adding a Driver



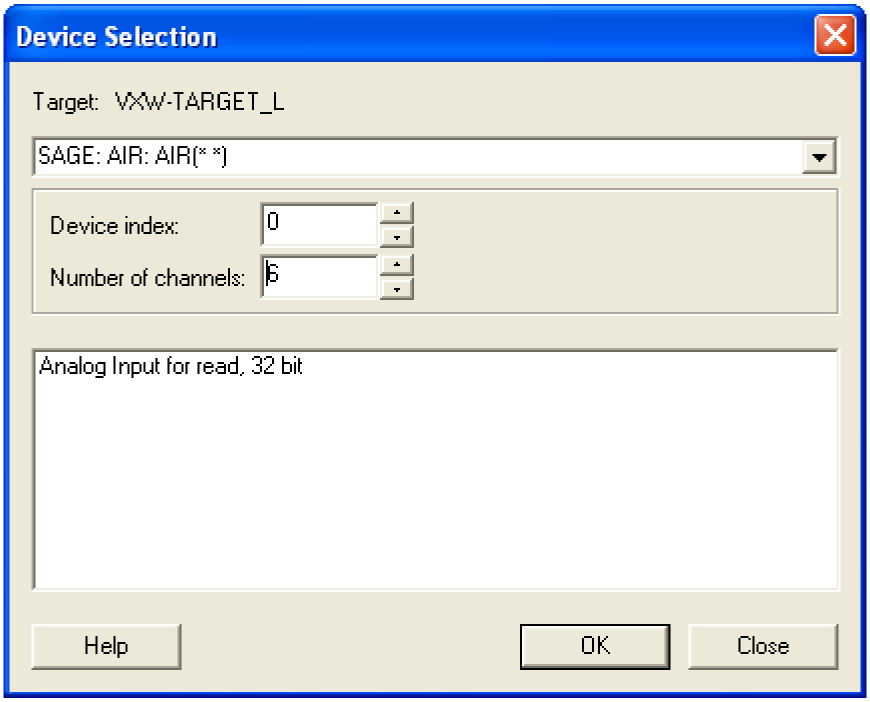
Select the driver from the drop-down scroll list as shown below.

Figure 3-69 Adding a Driver



Decide where in the driver list you want the new driver to appear. This is the Device Index. In the example below, it is set at 0, so it will appear as the first driver. Decide how many channels for the driver. You can have up to the maximum per driver type (but remember, if you have a limited I/O point version of ISaGRAF, the total is the maximum allowed by the license for all drivers). The example is 6. Click OK

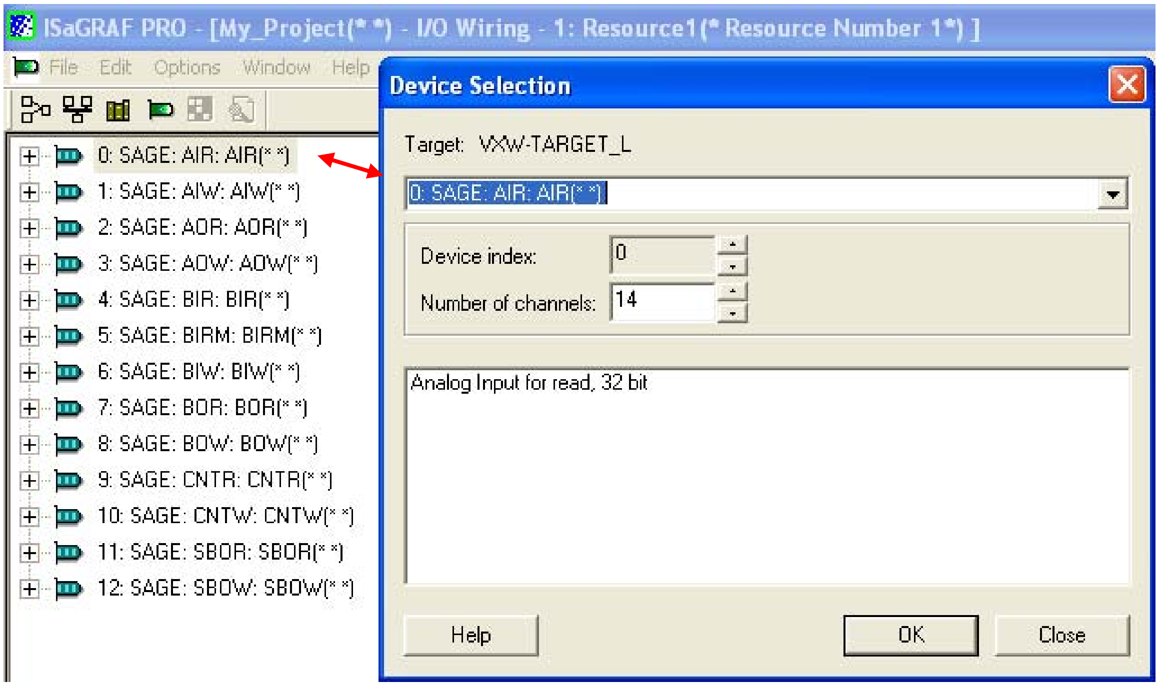
Figure 3-70 Device Index and Number of Channels



3.4.2.3 Modifying a Driver

You may change the number of channels for any driver type by double-clicking on that type, as shown below. In the example, the number of channels was changed to 14.

Figure 3-71 Modifying a Driver

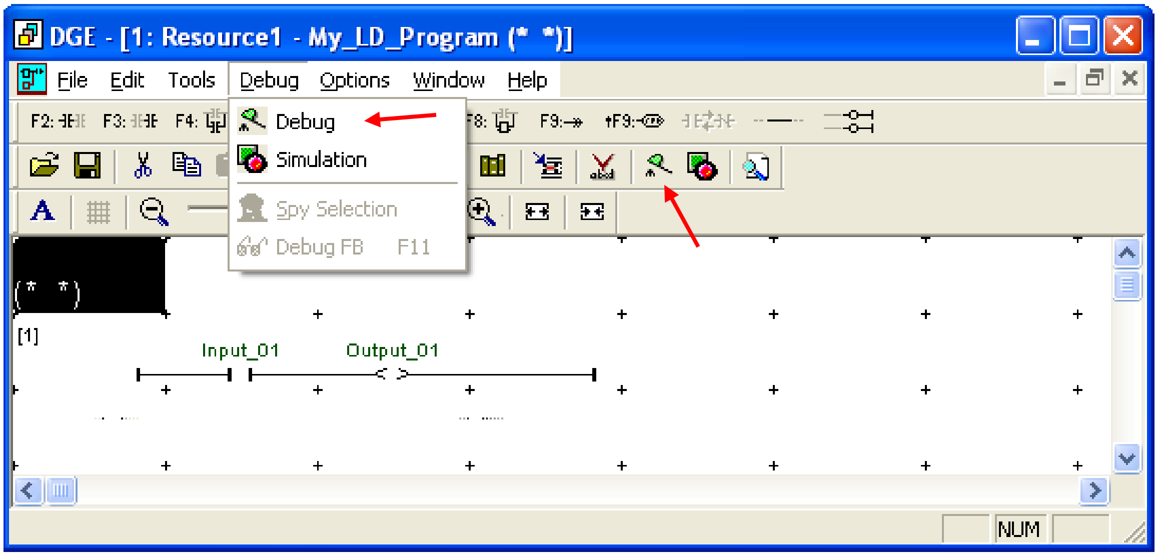


3.5 Program Debug

You can debug your program while the program is running in the RTU. If you prefer to debug your program off-line, see section 3.6 Program Simulation.

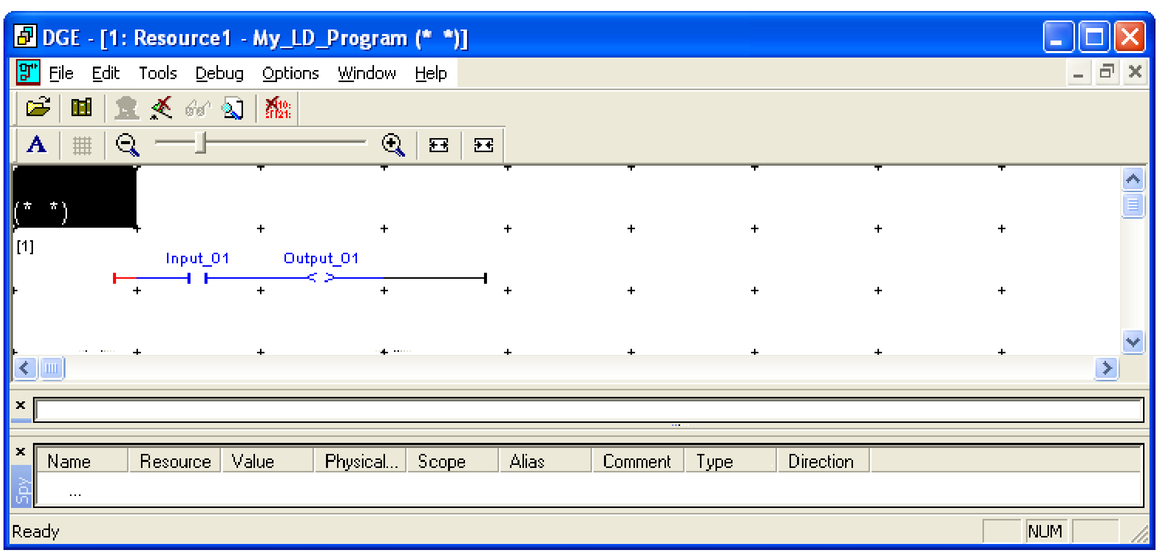
From the Program view, click on Debug either from the Debug drop-down menu or from the Debug icon on the toolbar as shown below.

Figure 3-72 Starting Debugging



After a few seconds, the debug mode will begin. You will get a screen similar to the one below. Notice that the program elements are colored blue. The line is red on the left side of Input_01. Red signifies True, blue signifies False.

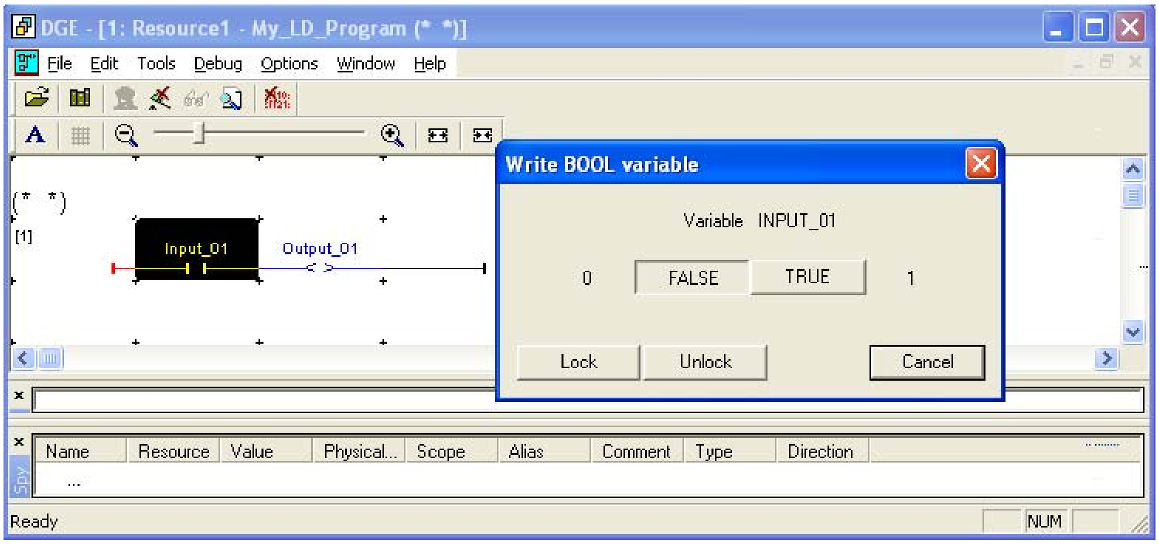
Figure 3-73 Debugging Mode



To begin a real-time debugging simulation, double-click on the first element, Input_01, as shown below. A dialog box will appear. The "Lock" button puts the program in simulation mode (the real-world input is "locked" out). The "Unlock" button returns the program to the real-world input.

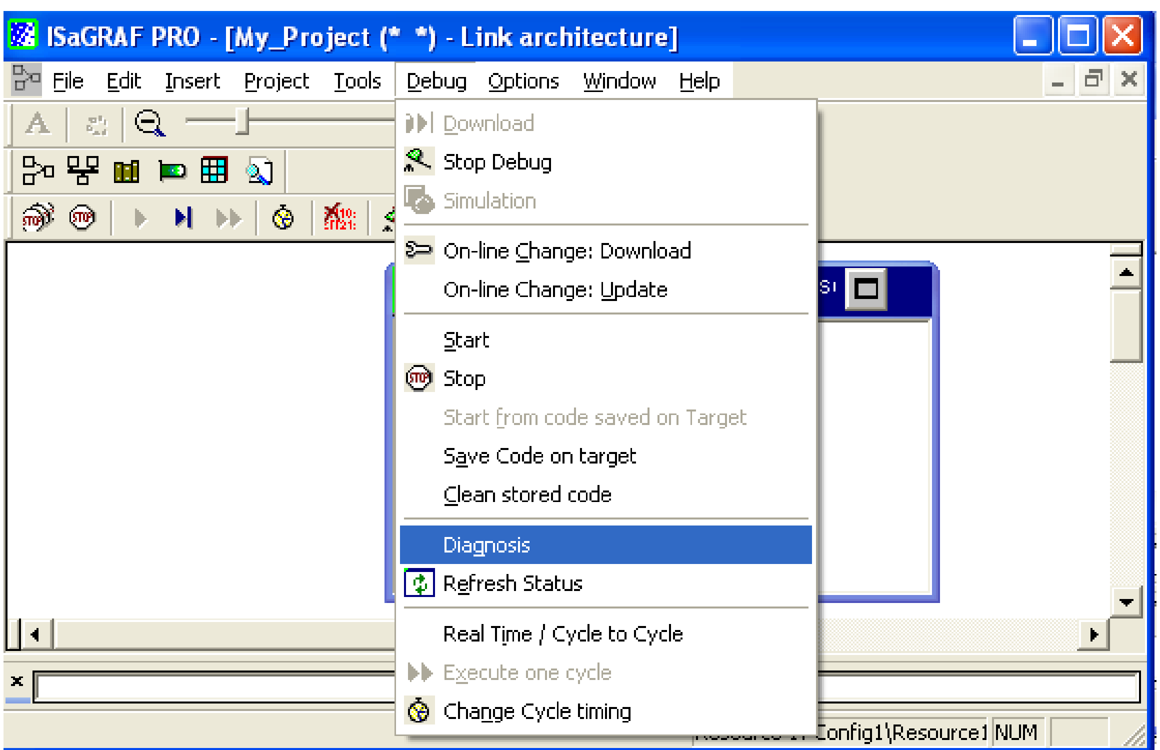
For debug simulation, click the Lock button. The dialog box will disappear after you do this. Just double-click on Input_01 again. You may now click on the “True” button to force the input to close. This will trip SBO 1. Notice that the color of Input_01 and Output_01 elements changes to red, or True. Double-click on Input_01 again. Click the False button to open Input_01. Click the Unlock button to return to the real-world input.

Figure 3-74 Debugging Mode



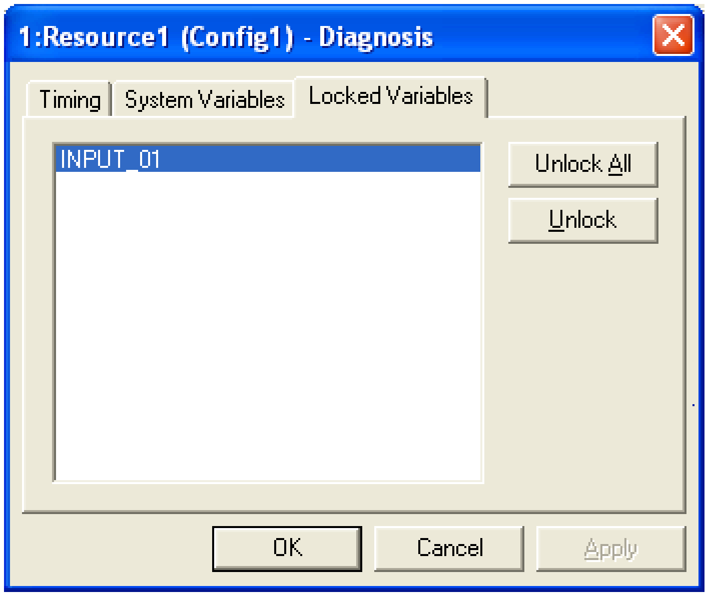
Unfortunately, the Lock and Unlock buttons do not show the current state. To see the current state, go to the Debug menu on the Link Architecture view and open the Diagnosis box as shown below.

Figure 3-75 Opening the Diagnosis Dialog Box



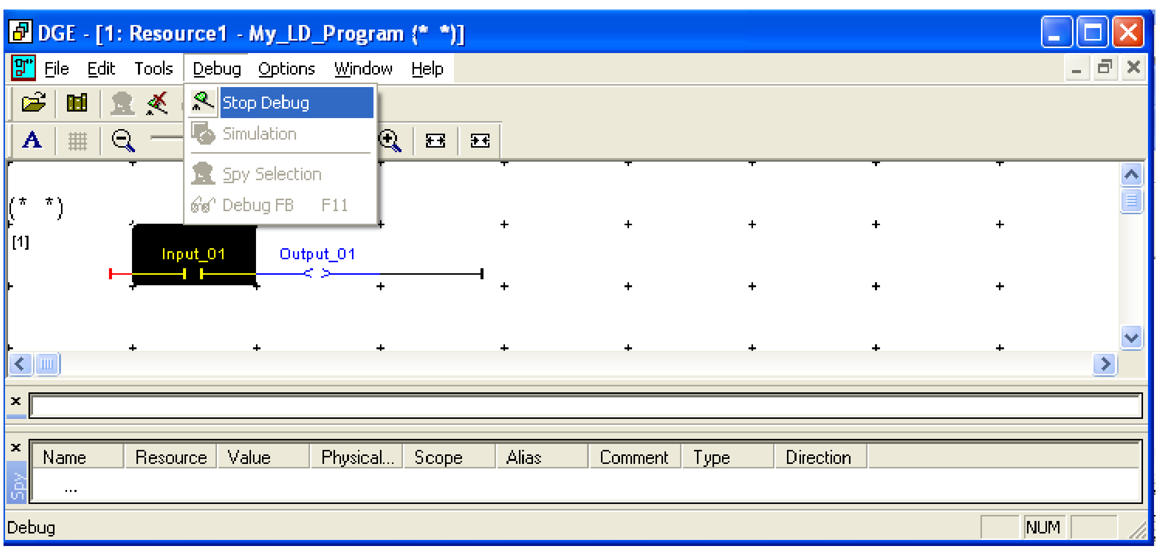
As shown below, the Locked Variables tab will show whatever variables are locked. You can unlock from this dialog box or simply use this box as an indication.

Figure 3-76 Diagnosis Dialog Box



Back at the Program display, you can stop debugging mode by selecting Stop Debug in the Debug drop-down menu as shown below.

Figure 3-77 Stopping the Debug Mode

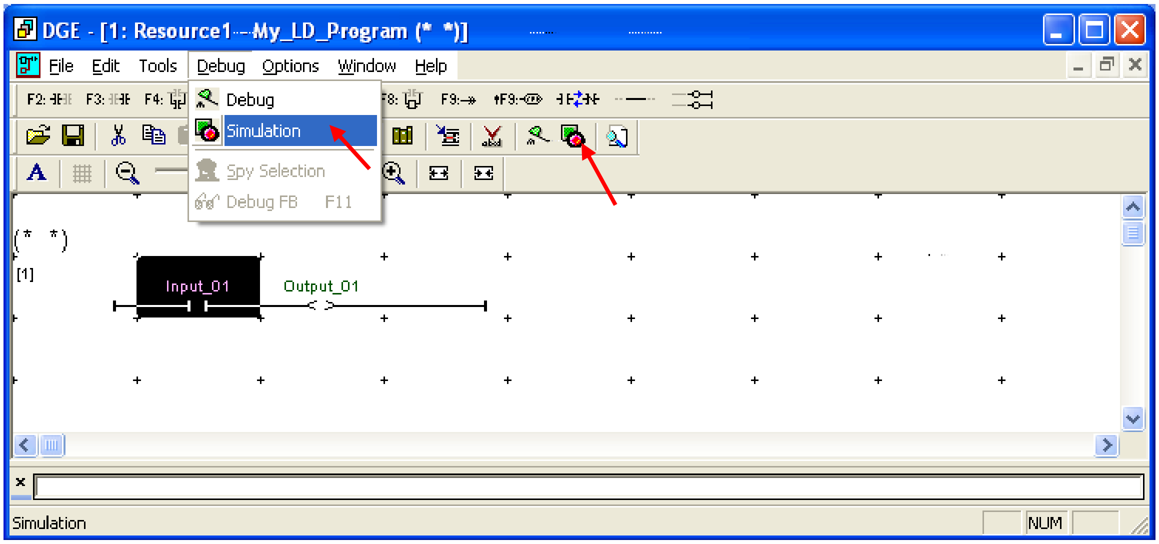


3.6 Program Simulation

Simulation gives you a way to test your program as it runs on your PC. Simulation will not change any real-world values in the RTU.

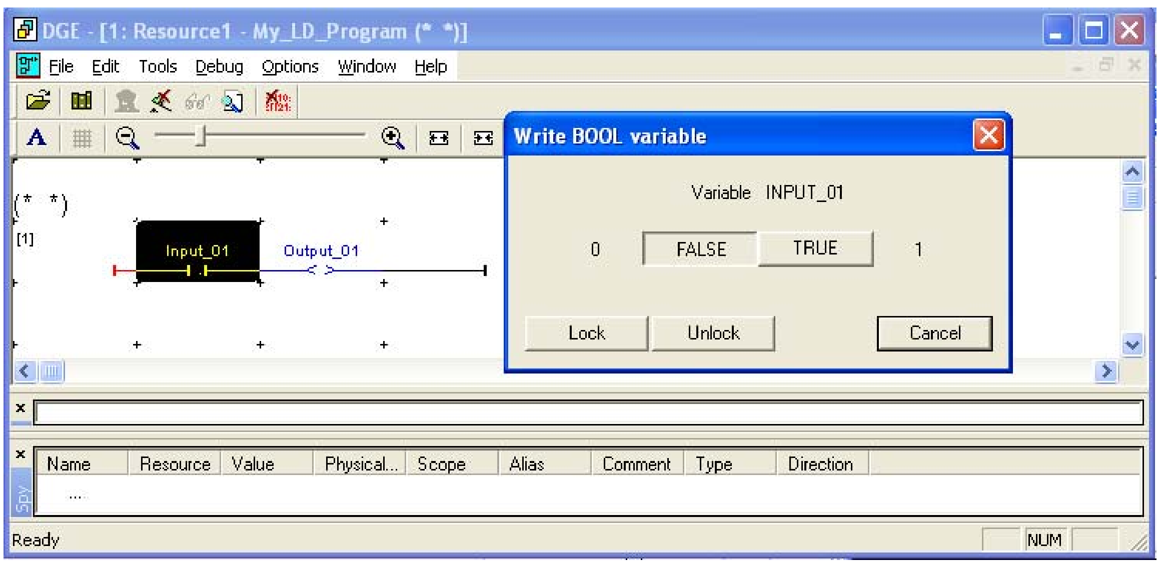
From the Program view, click on Simulation either from the Debug drop-down menu or from the Simulation icon on the toolbar as shown below.

Figure 3-78 Starting Simulation



Double-click on Input_01 to get the dialog box shown below.

Figure 3-79 Simulation Mode

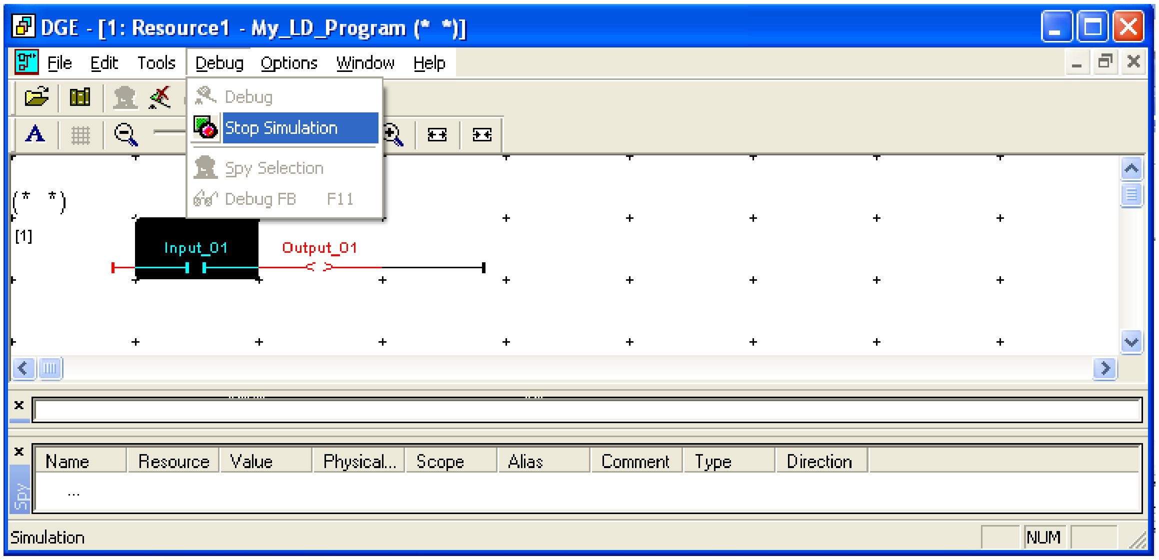


For the Simulation mode, the "Lock" – "Unlock" buttons have no meaning. The True and False buttons are the effective controls. They will change variables in the program on your PC. If you want to change variables on the program running in the RTU, see the Program Debug section.

As in the case with Debug, red is true, blue is false. When you change Input_01 to True, Output_01 turns red to show it has been activated.

You may stop Simulation mode by selecting Stop Simulation from the Debug drop-down menu as shown below.

Figure 3-80 Stop Simulation

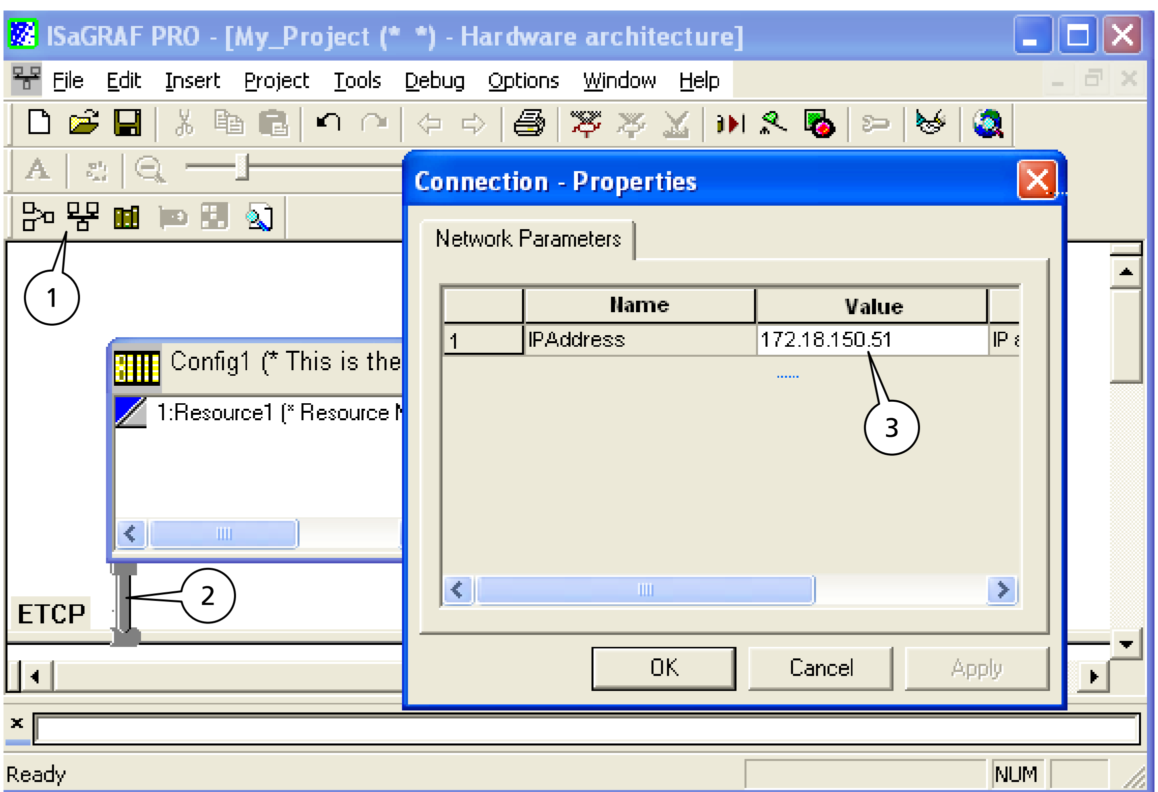


3.7 Changing the IP Address

The IP address that was set up in the beginning of this chapter is easy to change:

1. Click on Hardware Architecture from the Link Architecture view
2. Click on the bar connecting the ETCP network to the Config1 box
3. Enter the new IP address

Figure 3-81 Changing the IP Address



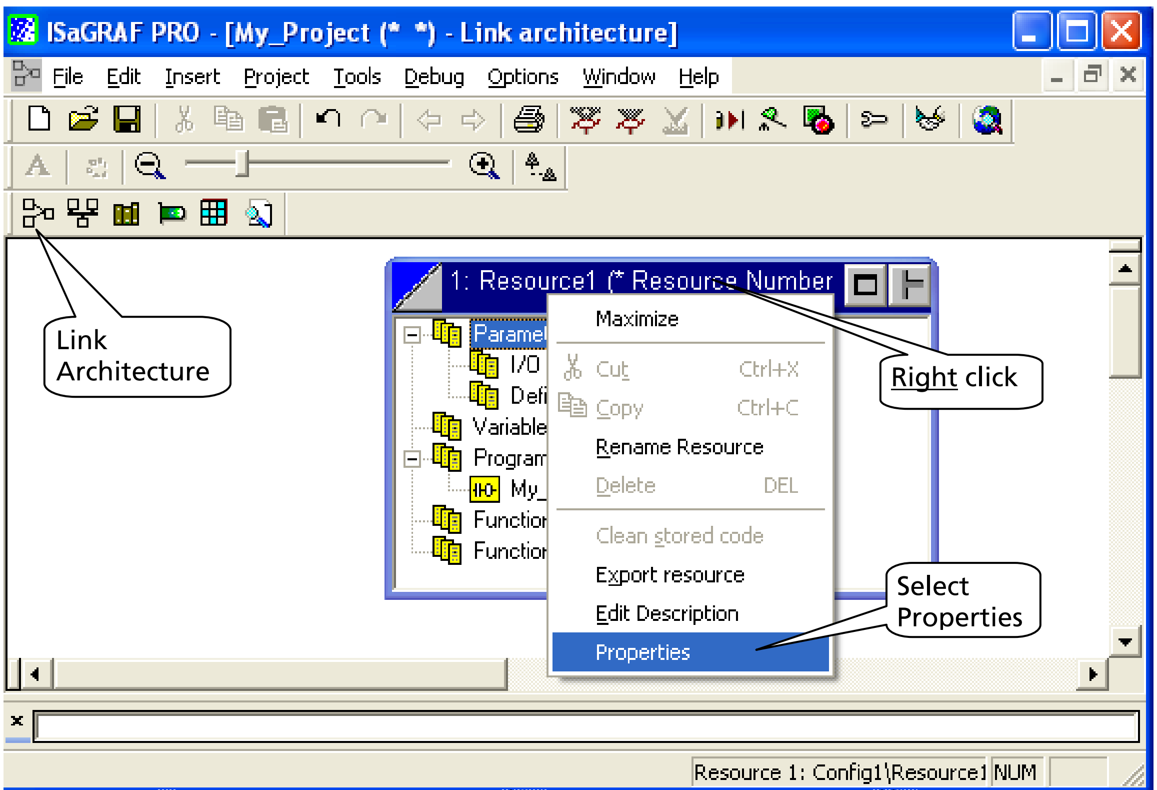
3.8 Changing the Program Cycle Time

The config_web_rll template determines the initial cycle time of the program. The default cycle time is set at 1000 milliseconds. Some applications may require a faster cycle time. Follow the directions below to change the cycle time.

Caution: If the RLL cycle time is too fast, the RTU can reset and cause the RTU to go into crash recovery mode. There are many factors that affect RTU performance, but keep in mind that a cycle time of less than 200 milliseconds is not recommended.

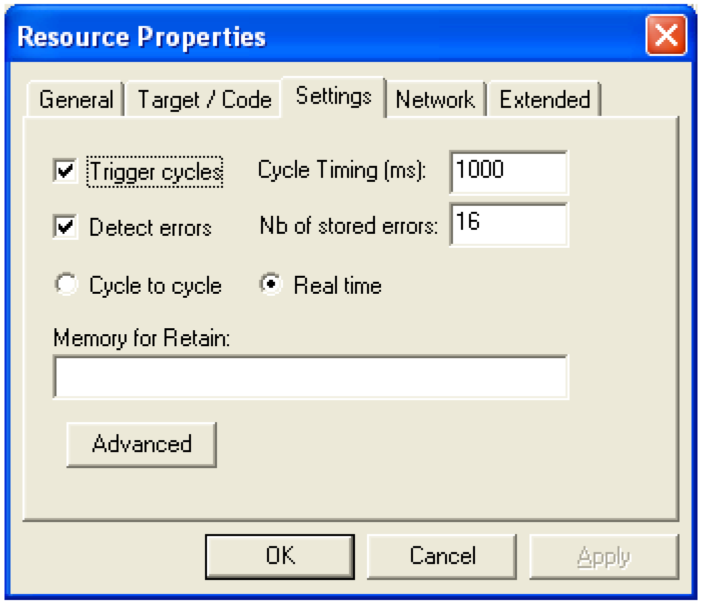
From the Link Architecture window, right-click on the Resource1 header and select Properties as shown below.

Figure 3-82 Finding the Cycle Time



Select the Settings tab. Enter a new Cycle Timing in milliseconds.

Figure 3-83 Changing the Cycle Time

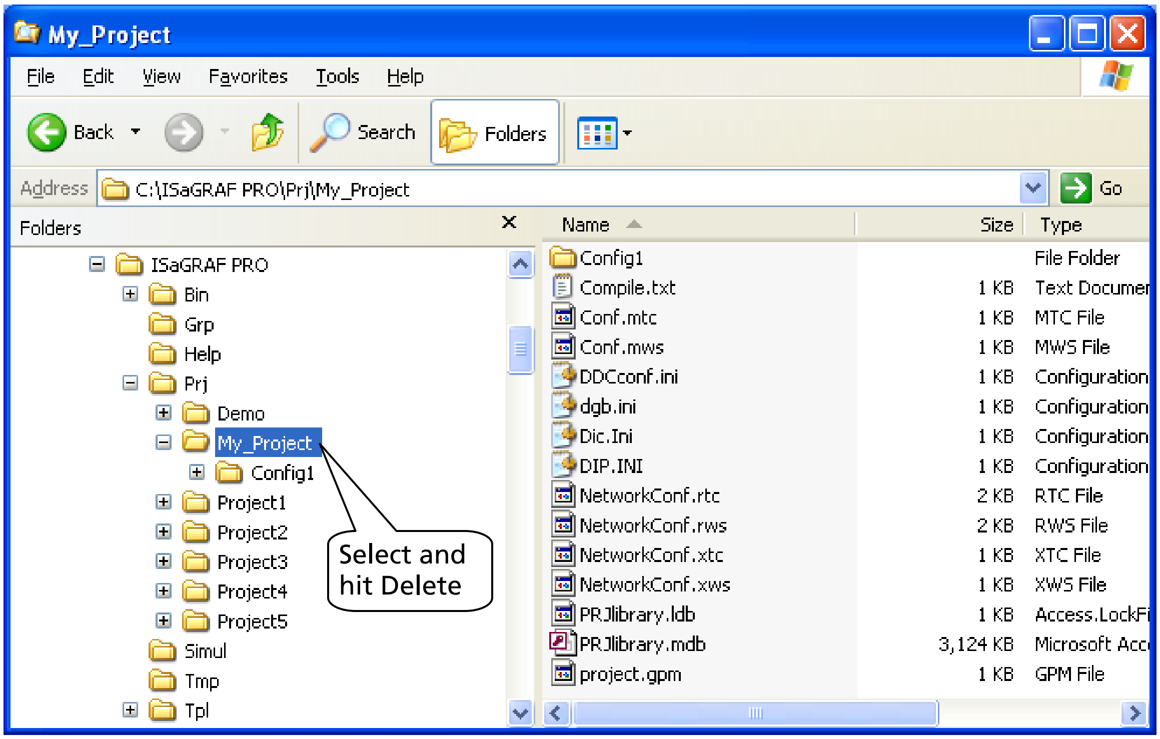


3.9 Managing Multiple Programs for Different RTUs

Create a different project for every program you need. Just make sure all projects are created under the config_web_rll template.

If you have projects you no longer need, you can delete them by navigating Windows Explorer to the ISaGRAF PRO directory, opening the Prj directory, selecting the project you wish to delete, and deleting it, as shown below.

Figure 3-84 Deleting a Project



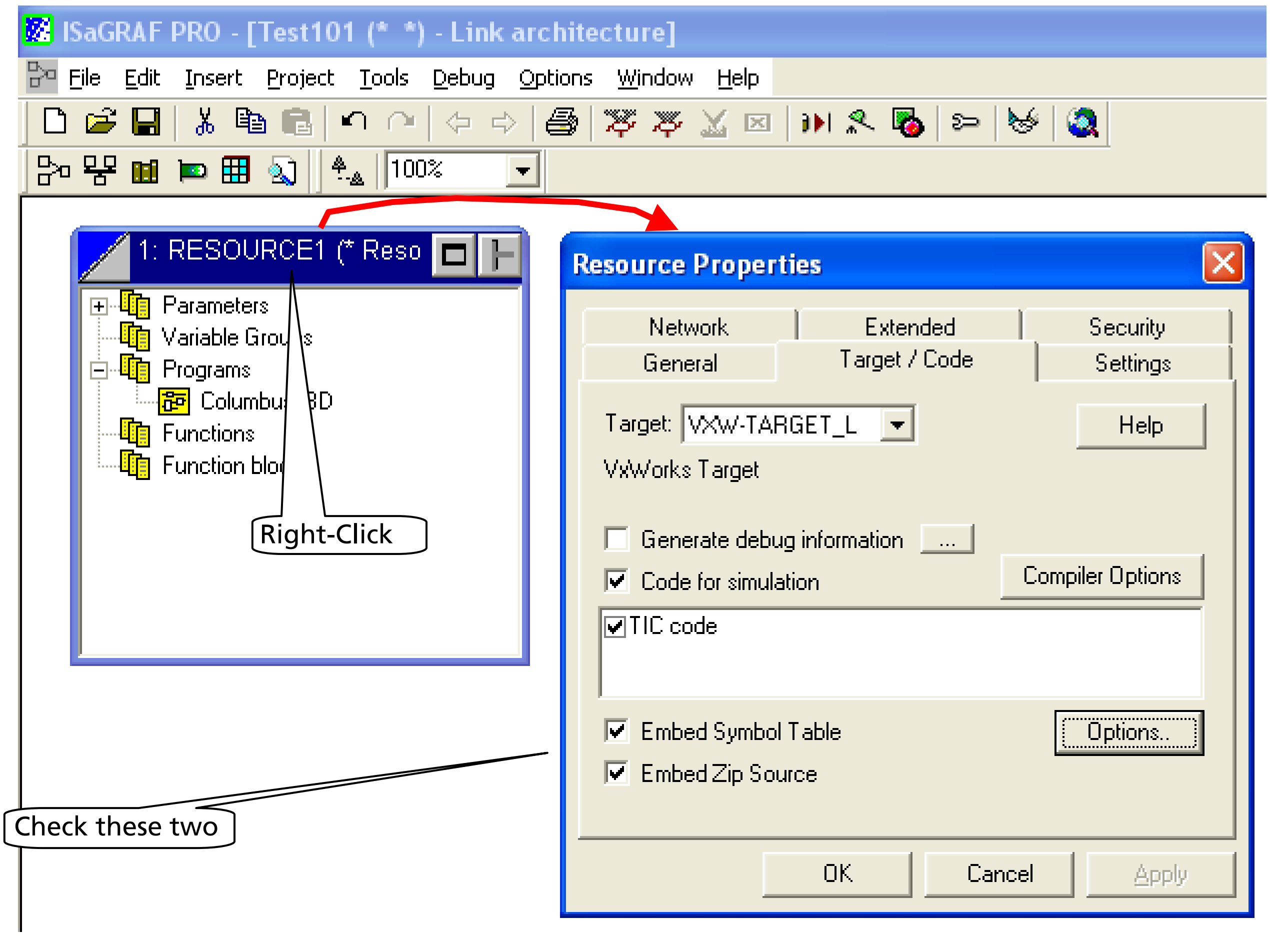
3.10 Downloading & Recovering Code to/from Target

This section describes how to download ISaGRAF source code to your RTU, then later recover the source code to your computer. There is a drawback to this technique and that is that before the source code is recovered to your computer, you must delete the resource in the ISaGRAF project. Therefore, before attempting this procedure, you should back up your project under a different name to be safe.

3.10.1 Downloading Code to Target

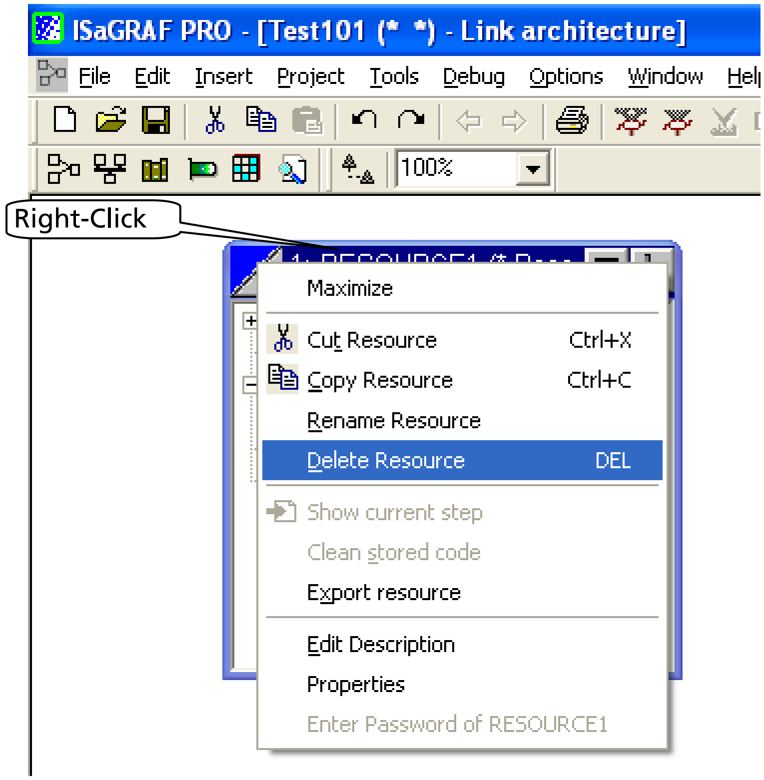
Follow the sequence described below.

Figure 3-85 Embedding Code to Target (RTU)



After the above procedure, Compile & Download to Target, as usual, then delete the Resource as shown in the next Figure.

Figure 3-86 Deleting the Resource



After deleting the Resource, **Save**.

3.10.2 Recovering Code from Target

To recover the code from the Target, go to File, select Import, then Resource

Figure 3-87 Importing the Resource

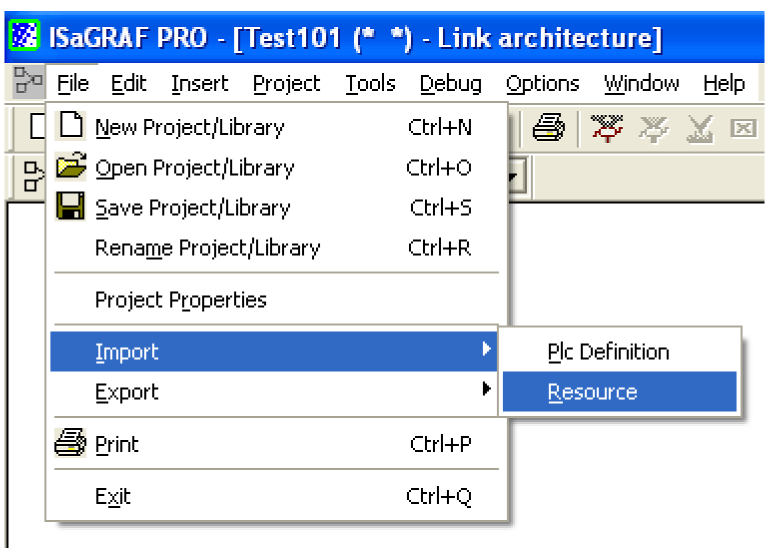


Figure 3-88 Importing the Resource

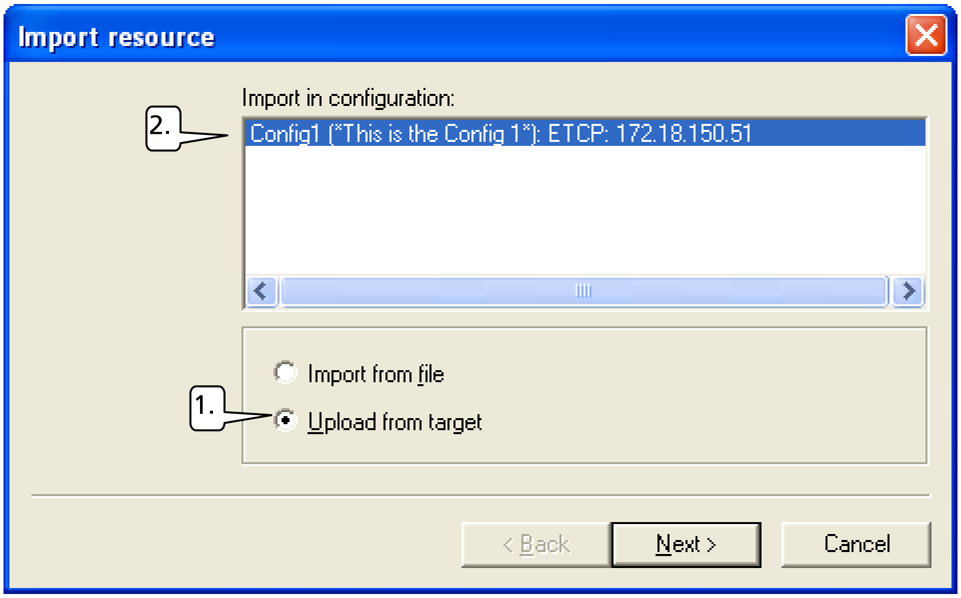


Figure 3-89 Warning Box

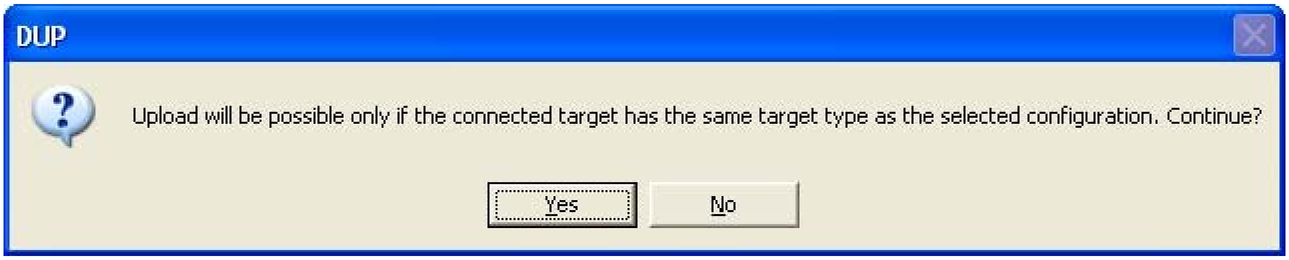


Figure 3-90 Importing the Resource

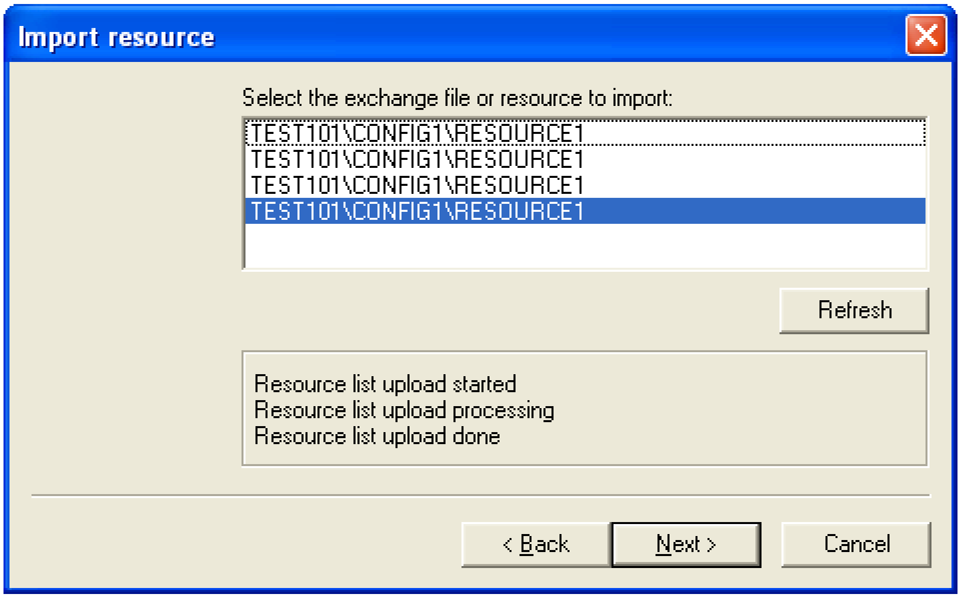


Figure 3-91 Importing the Resource

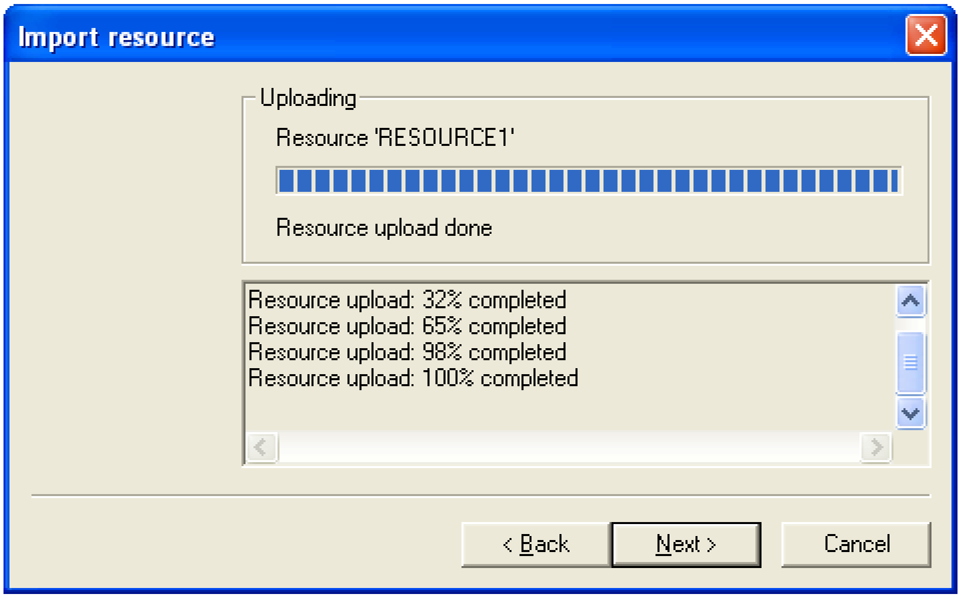
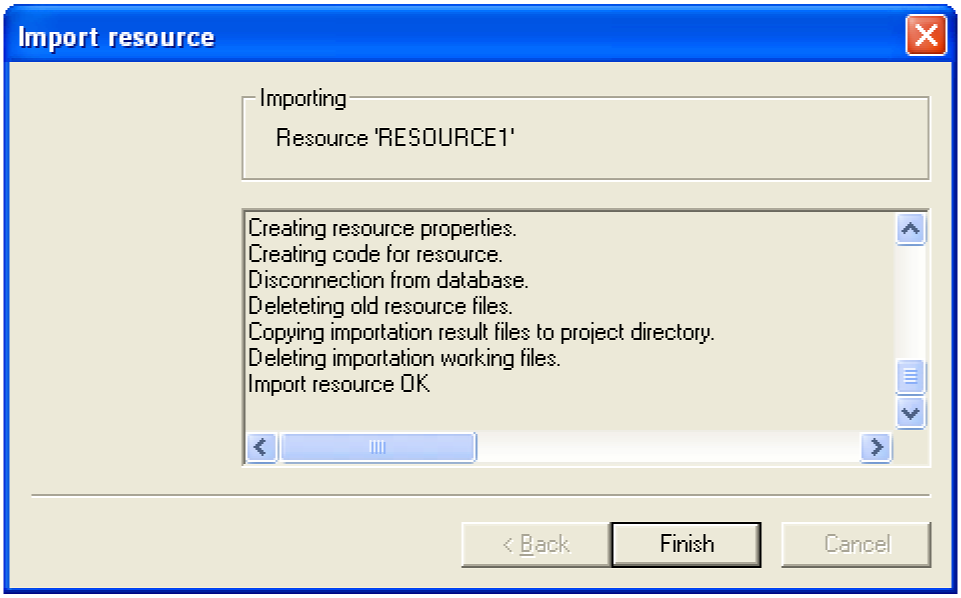
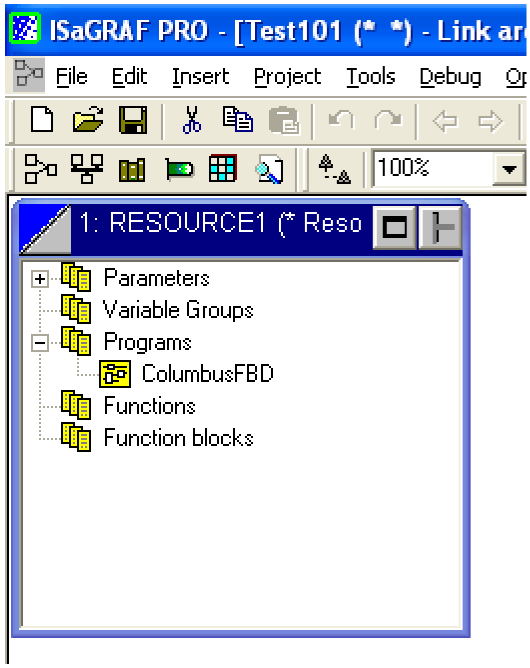


Figure 3-92 Importing the Resource



By looking at the view shown below, you will see that you have uploaded the source code from the RTU

Figure 3-93 The Resource has been Recovered from the Target RTU



3.11 ISaGRAF Program Maintenance

3.11.1 Clean Stored Code

Elsewhere in this manual, it is suggested that one way to stop a project from running on your target RTU after it has been downloaded, is to download a blank or "null" project. That technique works, but there is another option, as detailed below: "Clean Stored Code."

If you have downloaded a resource with the "Save" option checked in the Download dialog box, the resource's code is stored on the target system. Then if the target system restarts, it will load this code and start a virtual machine to run this code.

Note: If you want to clean (i.e. remove) this code from the target and avoid restarting on it, from the Debug menu, choose Clean Stored Code.

3.11.2 Cleaning Projects

The "Clean Projects" command gets rid of extraneous files from the last build. Unfortunately, ISaGRAF does not have a recommended period to perform this command. Yet, from customer experience, it is a good idea to perform this command occasionally.

The "Clean Project" or "Clean Resource" commands (on the Project menu of the Project Manager) simulate a modification of all the project's (or resource's) programs, so that they are all verified during the next "Build Project" or "Build Resource" operation.

Note: These commands actually delete all files that have been generated during the last "Build" command.

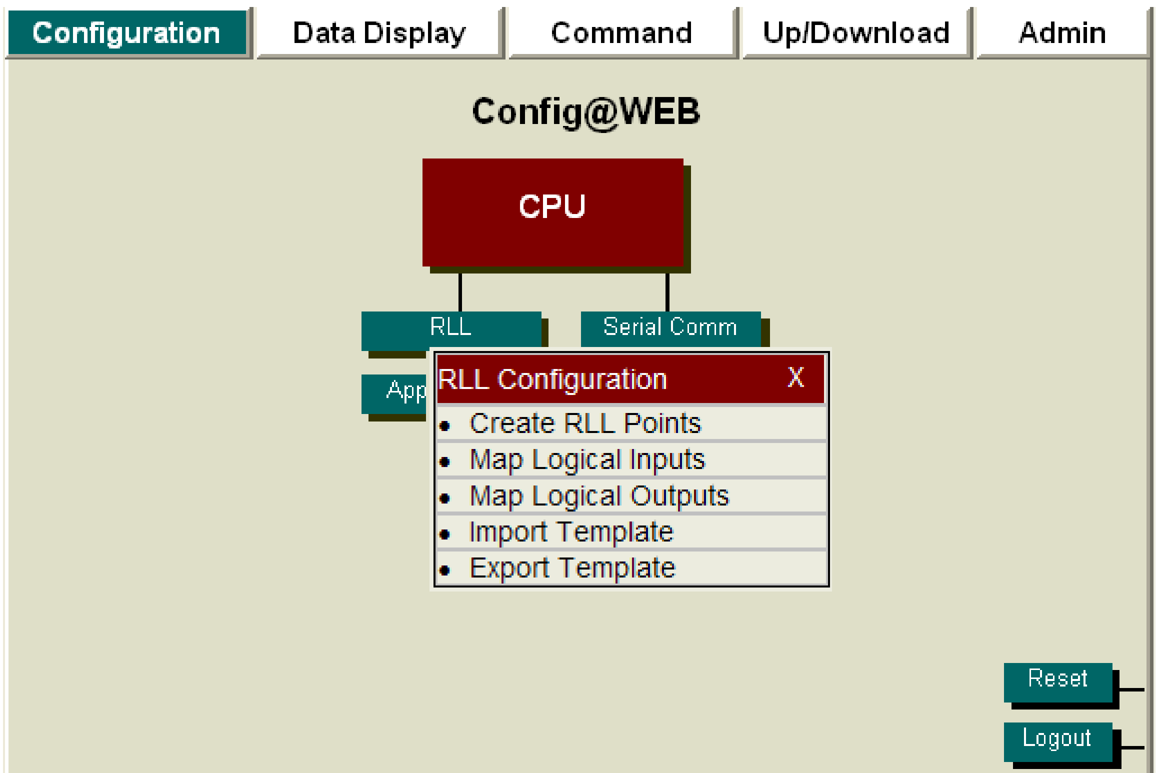
3.12 Reference Material - RLL Configuration

RLL Configuration is used to map points produced by ISaGRAF PRO, Telvent's Relay Ladder Logic programming package. ISaGRAF PRO (pronounced "is a graph pro") is supported in config@WEB firmware beginning with A8. The package contains six different programming languages, four of which are easy-to-use graphical languages. Please see the following manuals for detailed information on the operation of Telvent RTUs:

SAGE 1210 Operation & Maintenance Manual, Part # S1210-AAA-00001
SAGE 1230 Operation & Maintenance Manual, Part # S1230-AAA-00001
SAGE 2200 Operation & Maintenance Manual, Part # S2200-AAA-00001

The RLL Configuration is divided into three parts, as shown below. "Create RLL Points" allows you to create pseudo points, usually to map these points to a master station or an IED, either as commands (SBO, DO, AO) or as readable inputs (DI, AI, ACC). "Map Logical Inputs" and "Map Logical Outputs" allow you to connect the RLL program to hardware points.

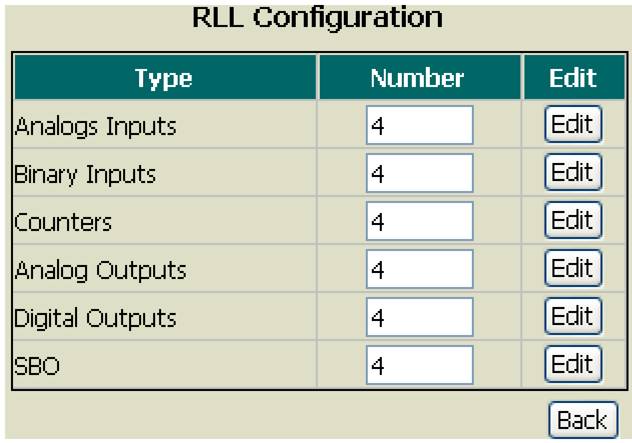
Figure 3-94 RLL Configuration



3.12.1 Create RLL Points

The RLL Configuration screen allows you to create RLL (pseudo) points. These are usually used to map logically derived points to a master station. One example would be to use the RLL logic to sum two hardware analogs and create an RLL analog point as the result to be read by the master station.

Figure 3-95 RLL Configuration



RLL Configuration

Type	Number	Edit
Analogs Inputs	4	Edit
Binary Inputs	4	Edit
Counters	4	Edit
Analog Outputs	4	Edit
Digital Outputs	4	Edit
SBO	4	Edit

Back

Type

The type of point

Number

The number of the specific type of point

Note: You must click the Edit button after entering a number or the entered number will not be retained.

Edit

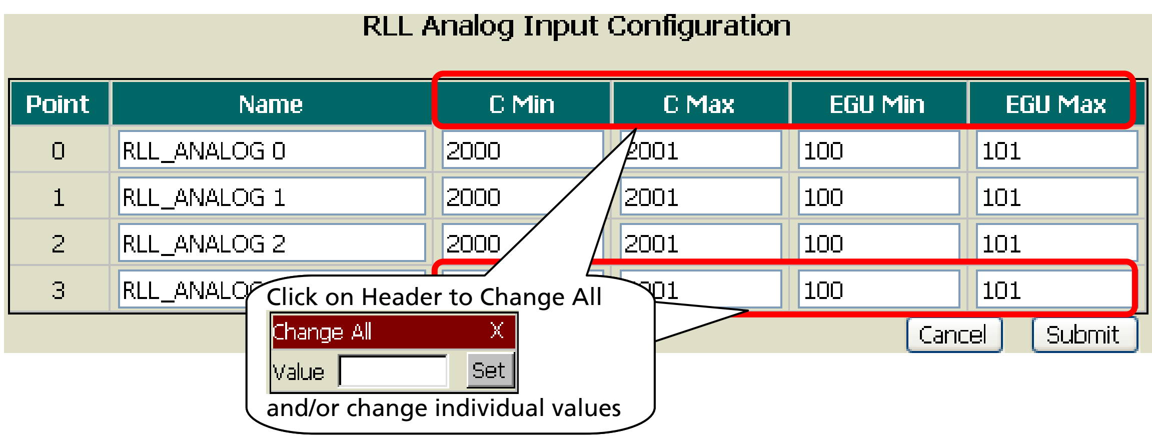
Click here to edit the specific parameters for the type of point.

Navigation

Click the Back button to go to the previous screen without changes.

3.12.1.1 Analog Inputs

Figure 3-96 RLL Analog Input Configuration



RLL Analog Input Configuration

Point	Name	C Min	C Max	EGU Min	EGU Max
0	RLL_ANALOG 0	2000	2001	100	101
1	RLL_ANALOG 1	2000	2001	100	101
2	RLL_ANALOG 2	2000	2001	100	101
3	RLL_ANALO		01	100	101

Point

RLL logical point number. This number cannot be changed.

Name

Enter the name of the point (or accept the default name).

C Min

Enter the Min count number. All entries in this column may be changed at once by clicking on the header.

C Max

Enter the Max count number. All entries in this column may be changed at once by clicking on the header.

EGU Min

Enter a minimum engineering unit value for the point. All entries in this column may be changed at once by clicking on the header.

EGU Max

Enter a maximum engineering unit value for the point. All entries in this column may be changed at once by clicking on the header.

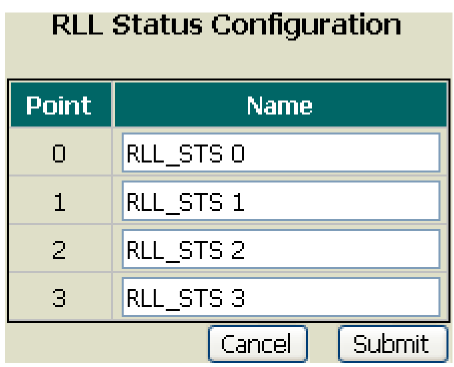
Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.1.2 Binary Inputs

Figure 3-97 RLL Status Configuration



Point

RLL logical point number. This number cannot be changed.

Name

Enter the name of the point (or accept the default name).

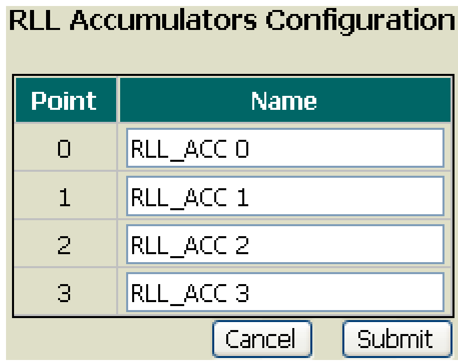
Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.1.3 Counters

Figure 3-98 RLL Accumulators Configuration



Point

RLL logical point number. This number cannot be changed.

Name

Enter the name of the point (or accept the default name).

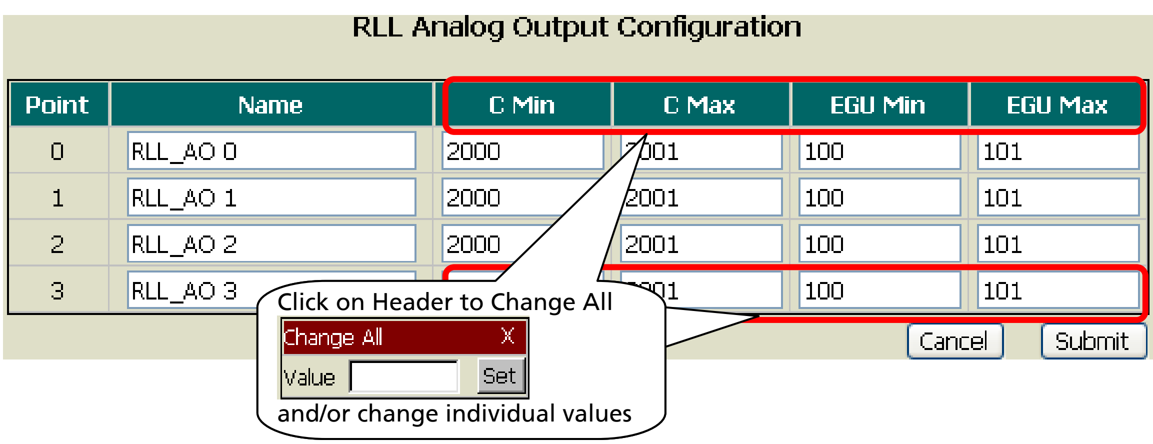
Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.1.4 Analog Outputs

Figure 3-99 RLL Analog Output Configuration



Point

RLL logical point number. This number cannot be changed.

Name

Enter the name of the point (or accept the default name).

C Min

Enter the Min count number. All entries in this column may be changed at once by clicking on the header.

C Max

Enter the Max count number. All entries in this column may be changed at once by clicking on the header.

EGU Min

Enter a minimum engineering unit value for the point. All entries in this column may be changed at once by clicking on the header.

EGU Max

Enter a maximum engineering unit value for the point. All entries in this column may be changed at once by clicking on the header.

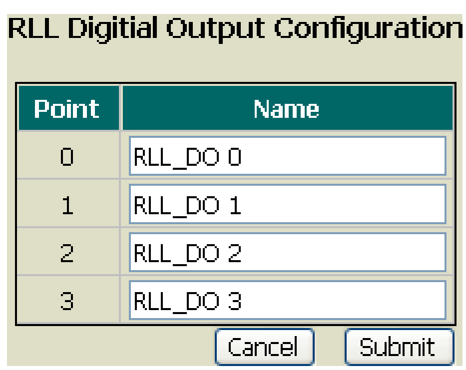
Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.1.5 Digital Outputs

Figure 3-100 RLL Digital Output Configuration



Point

RLL logical point number. This number cannot be changed.

Name

Enter the name of the point (or accept the default name).

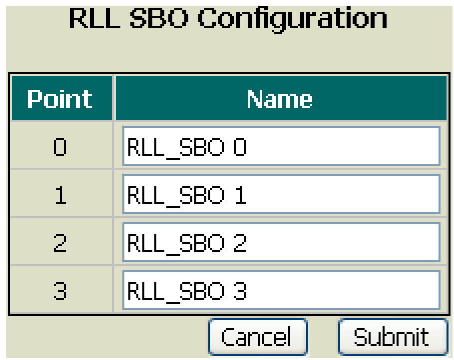
Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.1.6 SBO

Figure 3-101 RLL SBO Configuration



Point

RLL logical point number. This number cannot be changed.

Name

Enter the name of the point (or accept the default name).

Navigation

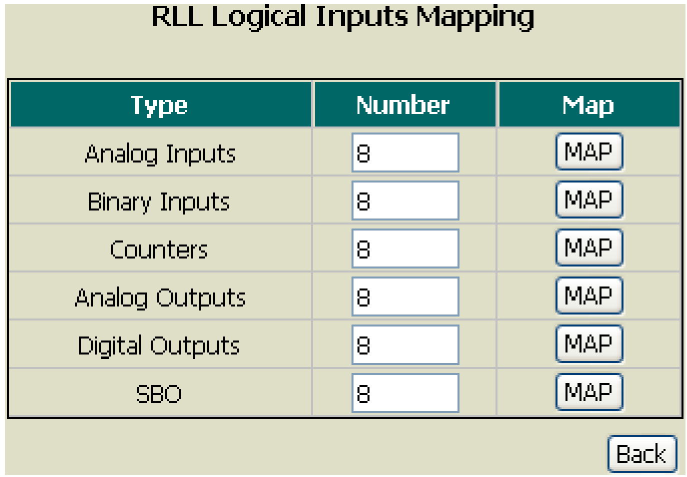
Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.2 Map Logical Inputs

The RLL Configuration screen allows you to map inputs for RLL points.

Figure 3-102 RLL Logical Inputs Mapping



Type

The type of point

Number

The number of the specific type of point

> **Note:** You must click the Map button after entering a number or the entered number will not be retained.

Map

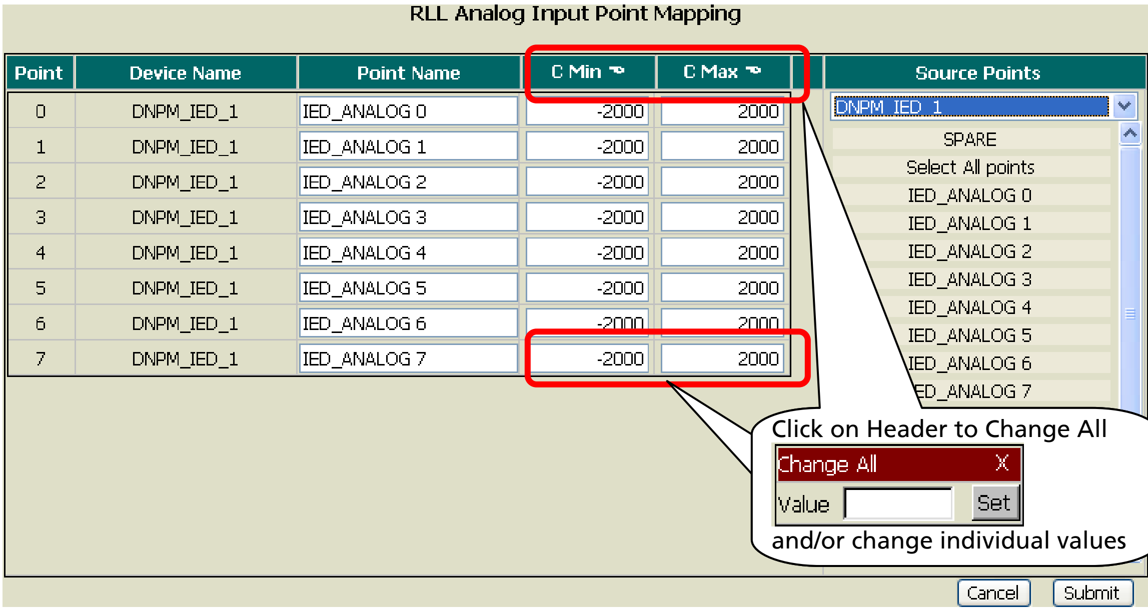
Click here to map points.

Navigation

Click the Back button to go to the previous screen without changes.

3.12.2.1 Analog Inputs

Figure 3-103 RLL Analog Input Configuration



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

C Min

Enter the minimum counts required or accept the default counts. Default setting is -2000.

C Max

Enter the maximum counts required or accept the default counts. Default setting is 2000.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

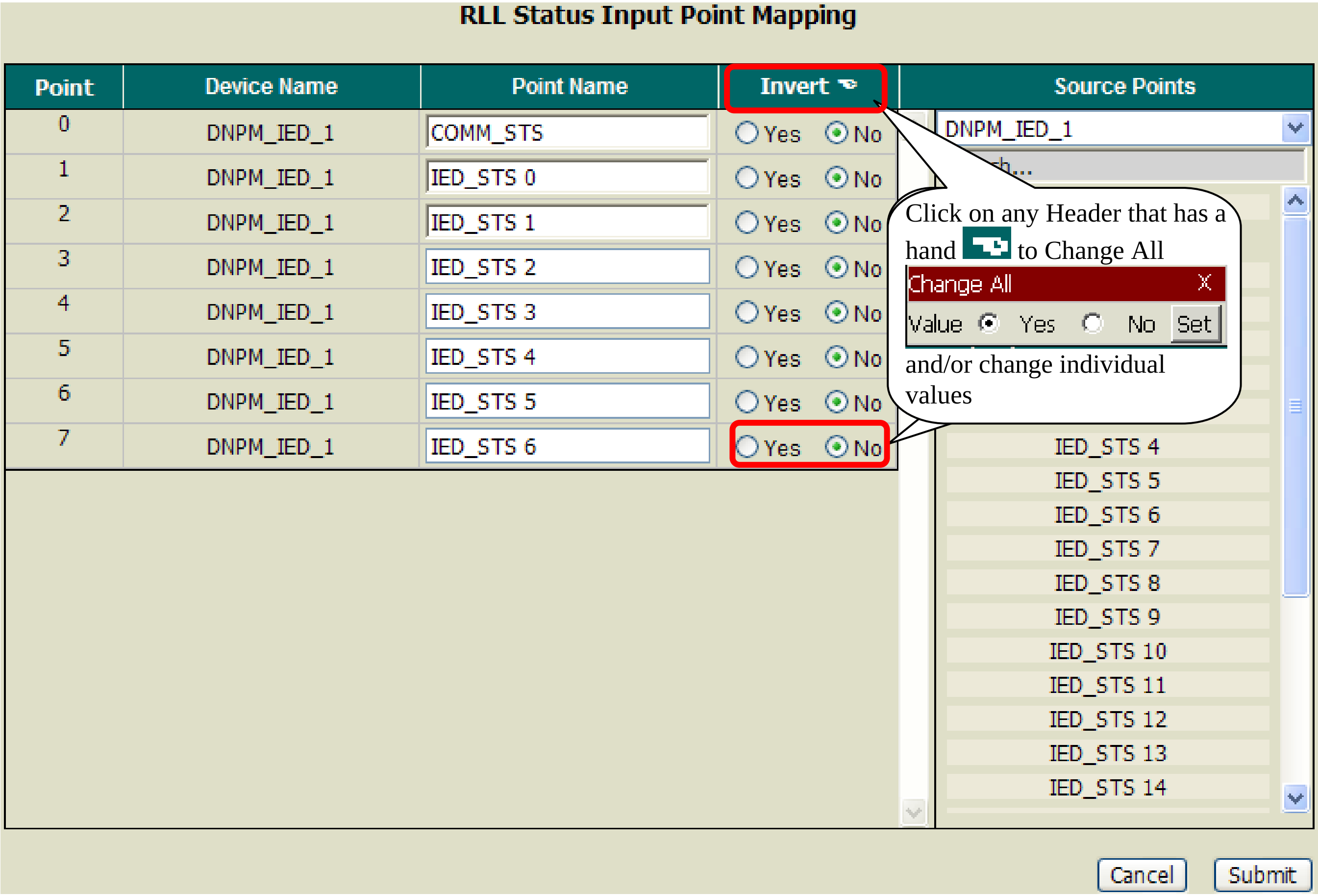
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.2.2 Binary Inputs

Figure 3-104 RLL Status Input Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

Invert

Click Yes to invert the point. The default is No.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

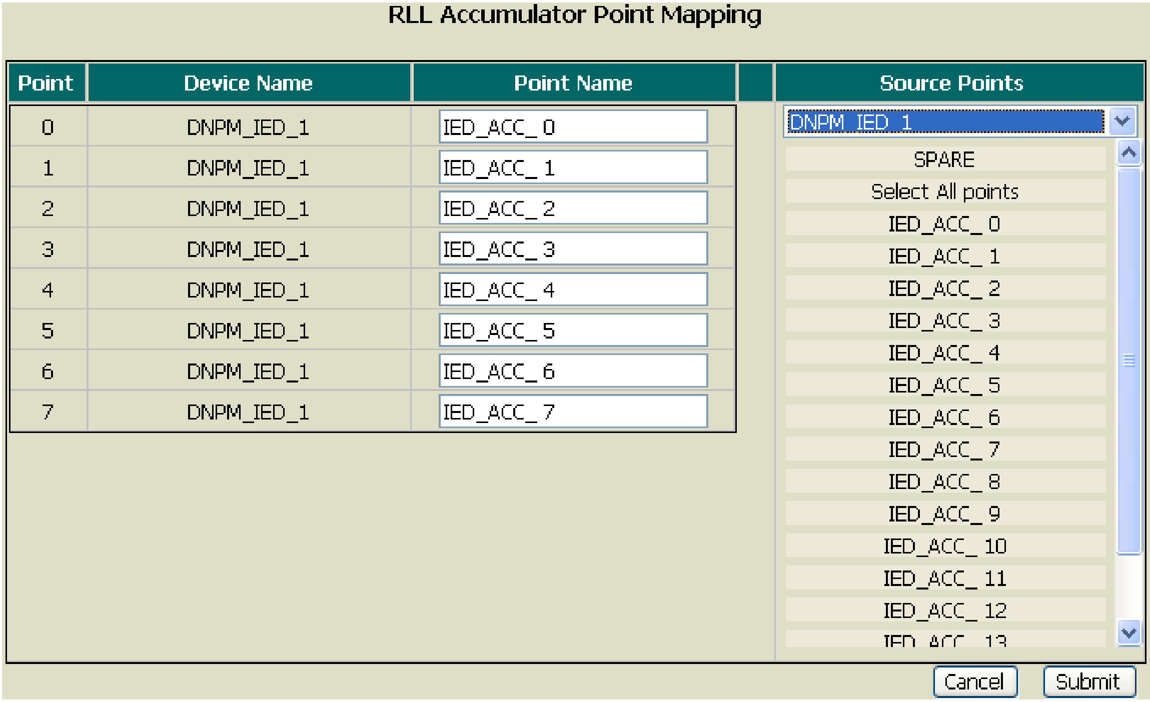
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.2.3 Counters

Figure 3-105 RLL Accumulator Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

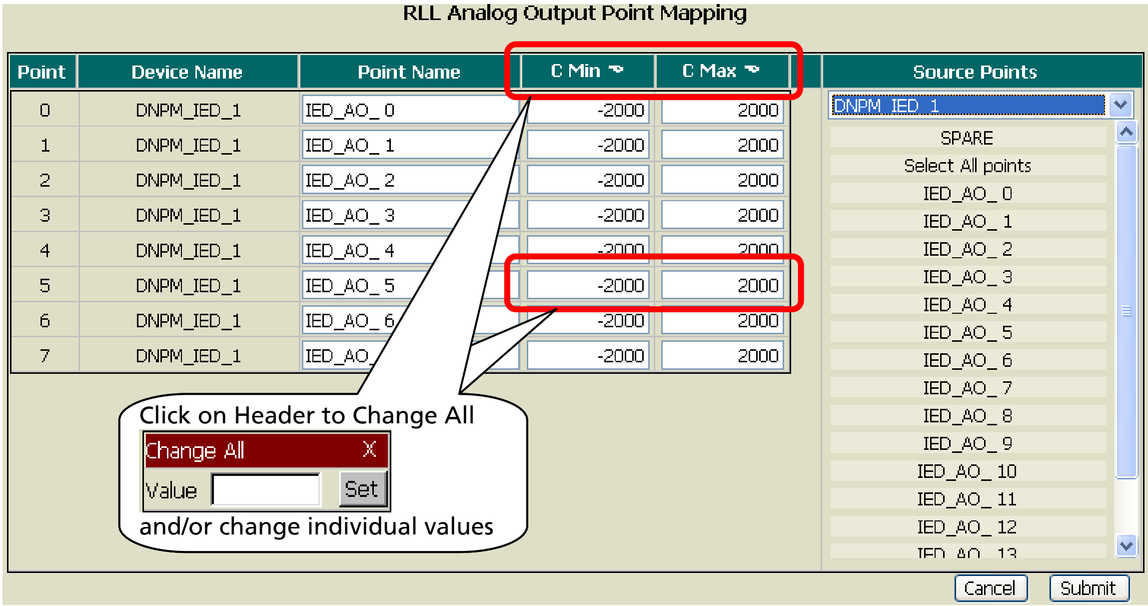
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.2.4 Analog Outputs

Figure 3-106 RLL Analog Output Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

C Min/C Max

Enter the counts required, or accept the default counts.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

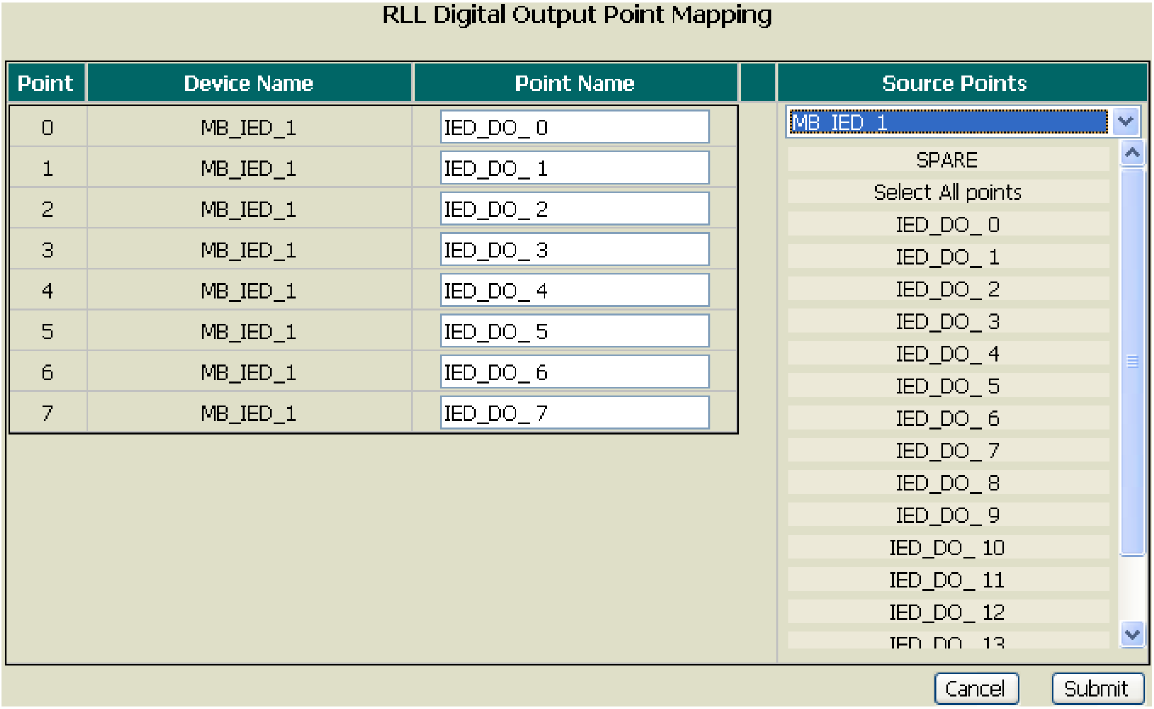
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.2.5 Digital Outputs

Figure 3-107 RLL Digital Output Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

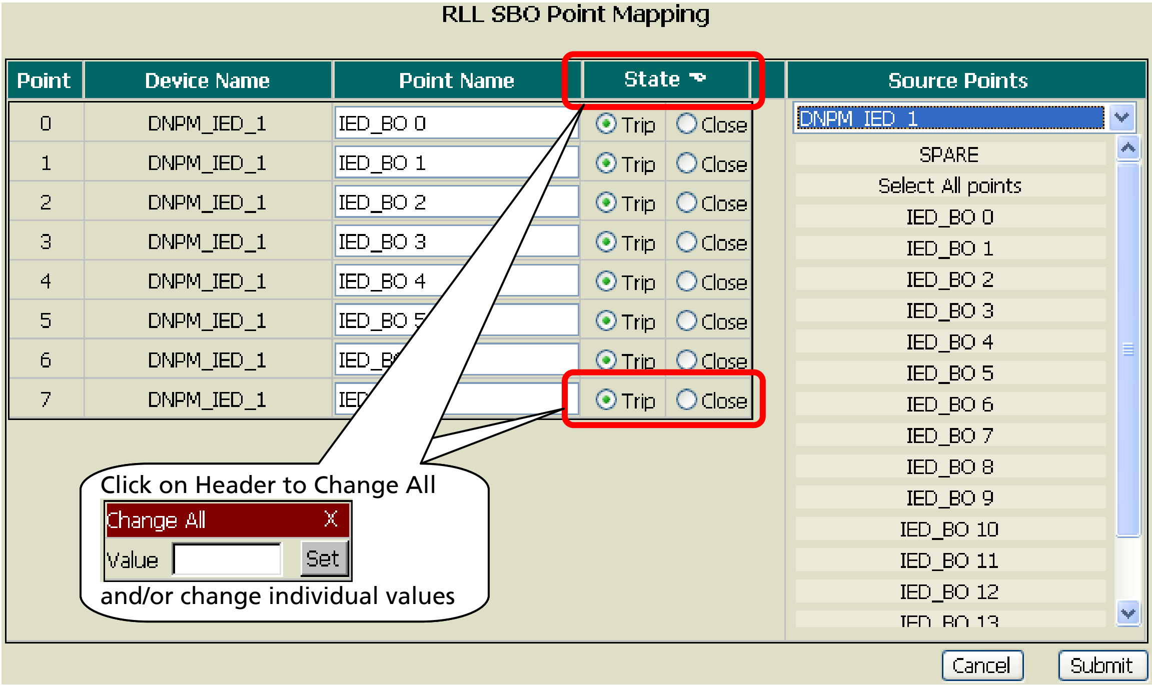
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.2.6 SBO

Figure 3-108 RLL SBO Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

State

Select Close or accept the default of Trip.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

Navigation

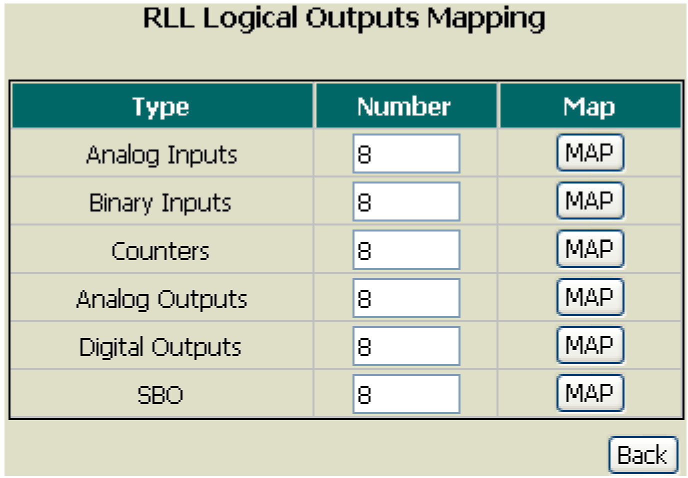
Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.3 Map Logical Outputs

The RLL Configuration screen allows you to map Outputs for RLL points.

Figure 3-109 RLL Logical Outputs Mapping



Type

The type of point

Number

The number of the specific type of point

Note: You must click the Map button after entering a number or the entered number will not be retained.

Map

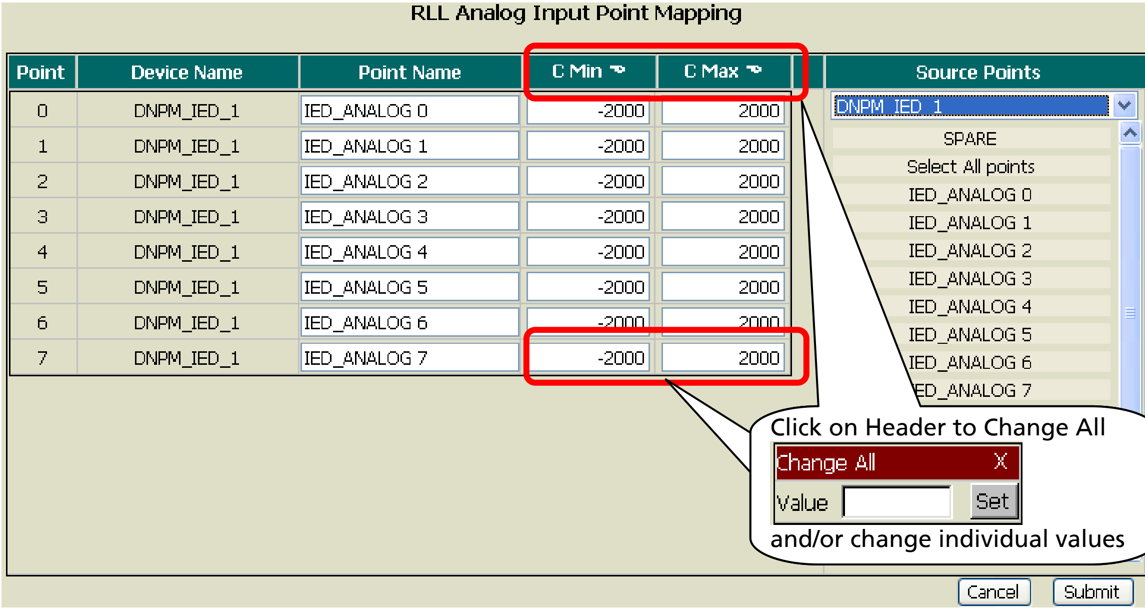
Click here to map points.

Navigation

Click the Back button to go to the previous screen without changes.

3.12.3.1 Analog Inputs

Figure 3-110 RLL Analog Input Configuration



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

C Min

Enter the minimum counts required or accept the default counts. Default setting is -2000.

C Max

Enter the maximum counts required or accept the default counts. Default setting is 2000.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

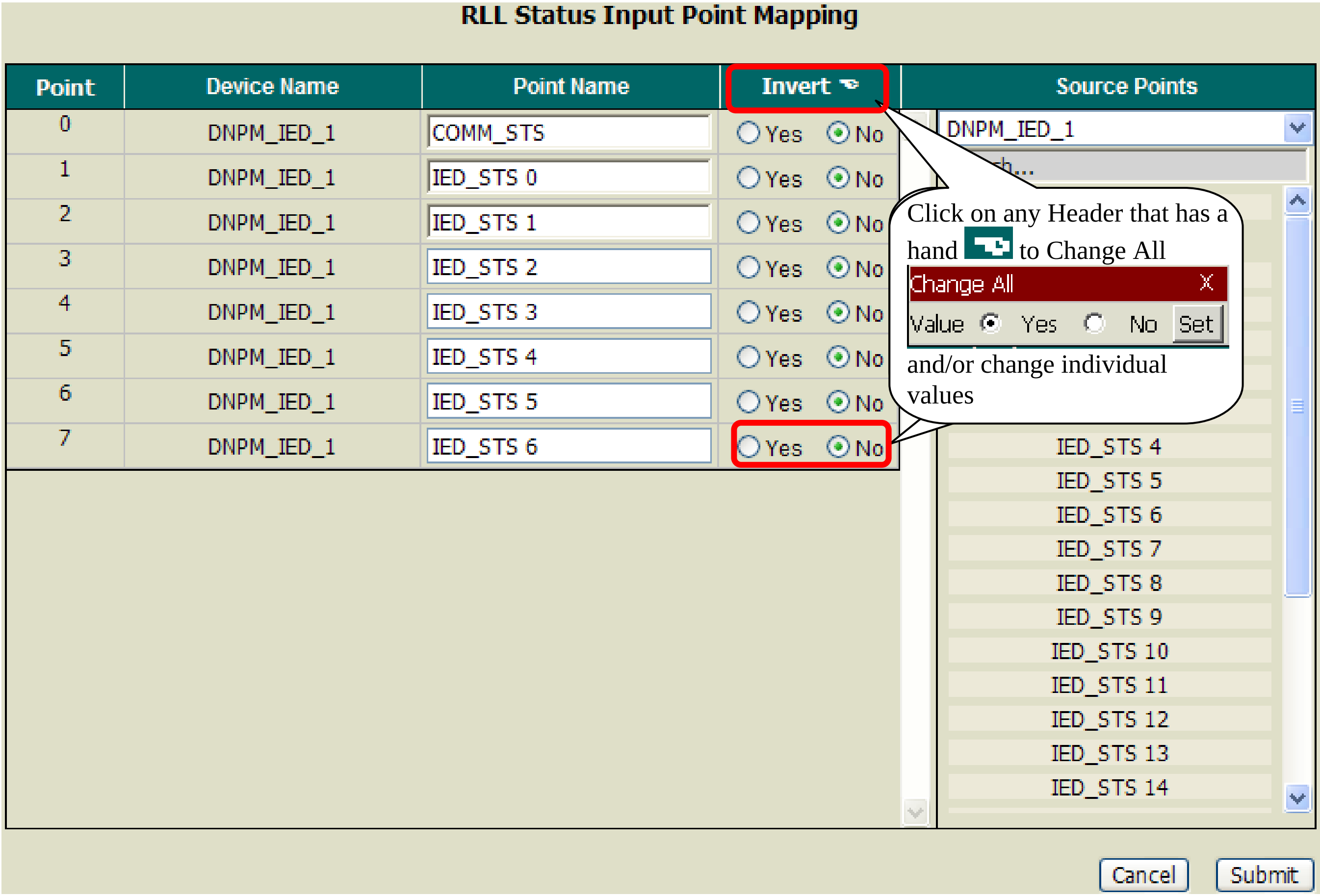
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.3.2 Binary Inputs

Figure 3-111 RLL Status Input Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

Invert

Click Yes to invert the point. The default is No.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

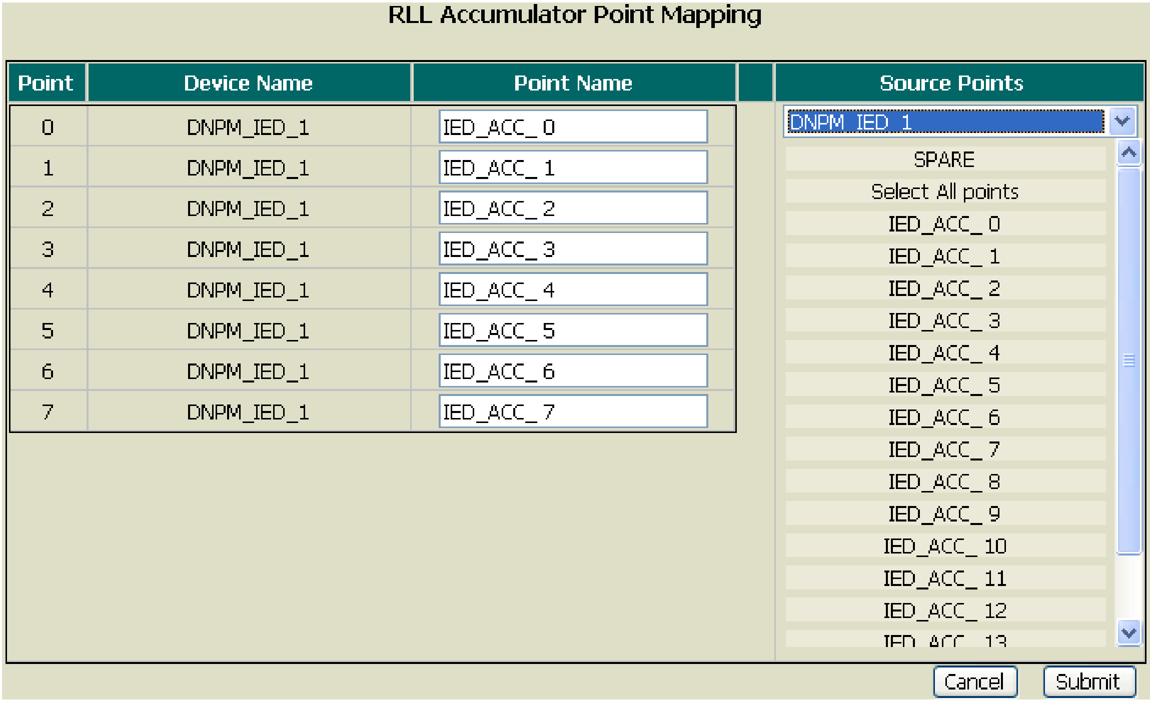
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.3.3 Counters

Figure 3-112 RLL Accumulator Point Mapping



Point	Device Name	Point Name
0	DNPM_IED_1	IED_ACC_ 0
1	DNPM_IED_1	IED_ACC_ 1
2	DNPM_IED_1	IED_ACC_ 2
3	DNPM_IED_1	IED_ACC_ 3
4	DNPM_IED_1	IED_ACC_ 4
5	DNPM_IED_1	IED_ACC_ 5
6	DNPM_IED_1	IED_ACC_ 6
7	DNPM_IED_1	IED_ACC_ 7

Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

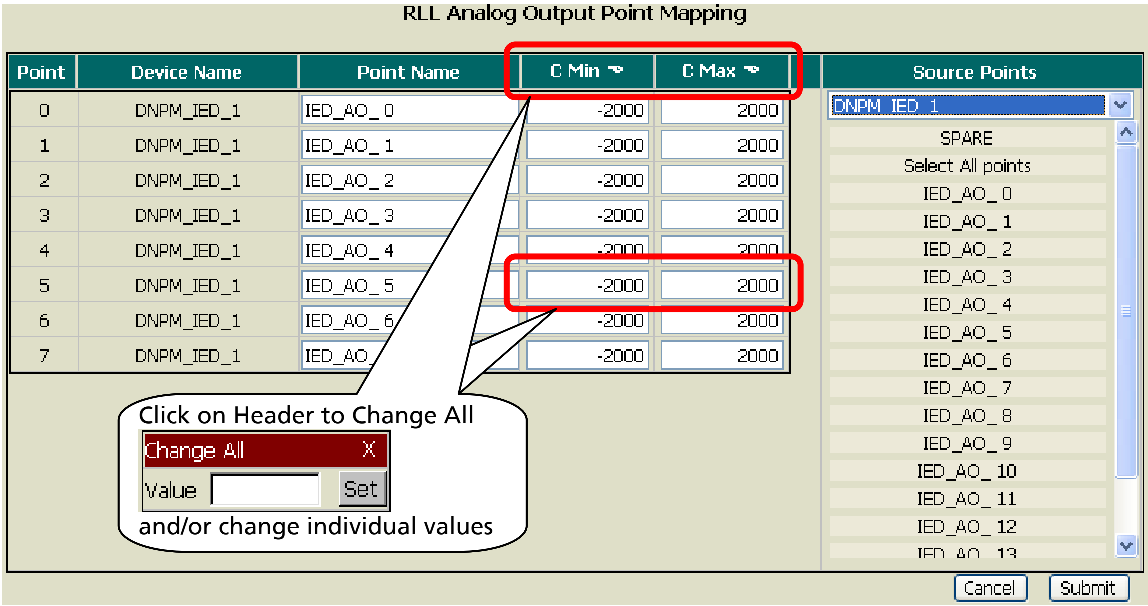
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.3.4 Analog Outputs

Figure 3-113 RLL Analog Output Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

C Min/C Max

Enter the counts required, or accept the default counts.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

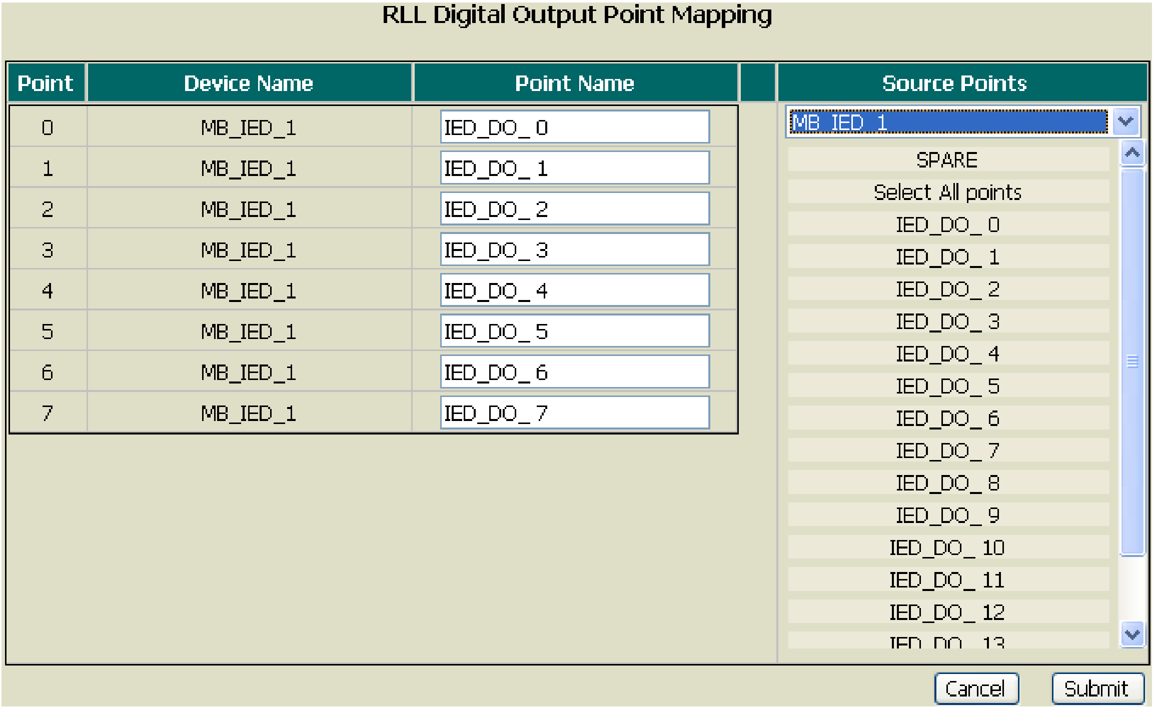
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.3.5 Digital Outputs

Figure 3-114 RLL Digital Output Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

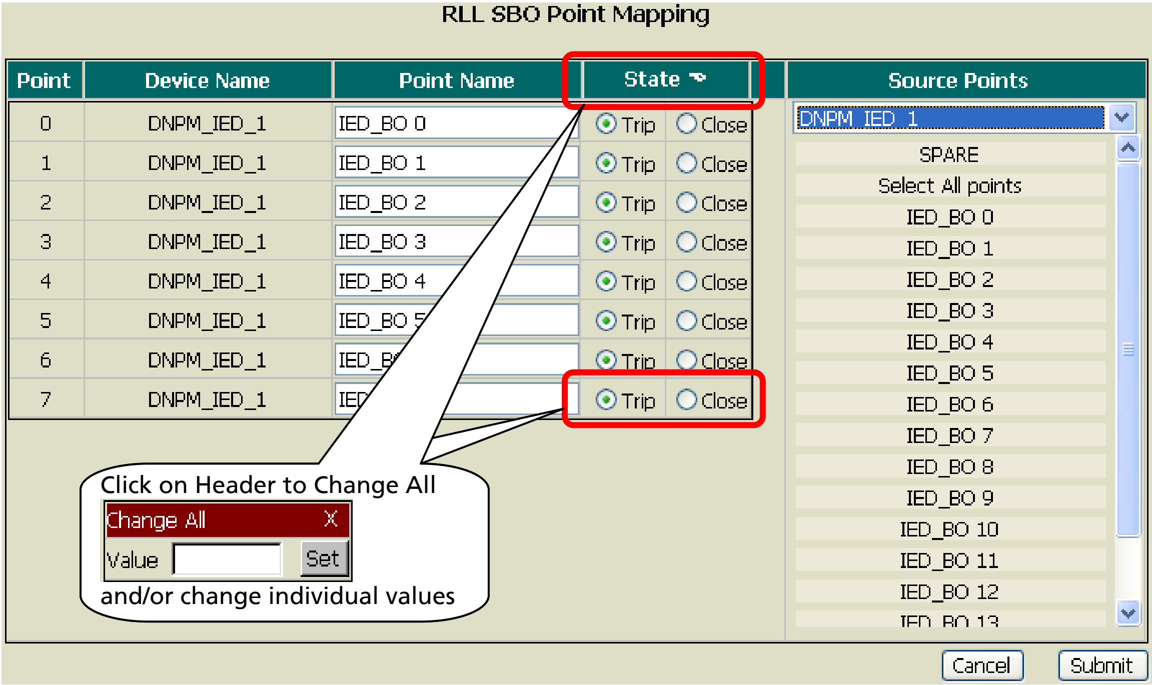
Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

3.12.3.6 SBO

Figure 3-115 RLL SBO Point Mapping



Point

RLL logical point number. This number cannot be changed.

Device Name

The name of the source device for the mapped point.

Point Name

The name of the mapped point.

State

Select Close or accept the default of Trip.

Source Points

Select the source points to place under Point Name from the drop-down list. Single points, or all points, or spare, may be selected.

Navigation

Click the Cancel button to discard changes. Click the Submit button to accept the changes.

Please note: No configuration changes take effect until the RTU is reset.

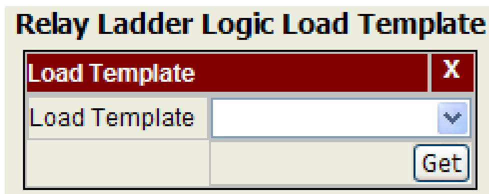
3.12.4 Import/Export Templates

The purpose of the functions Import Template and Export Template in RLL is to be able to save an RLL configuration to be used on another RTU (or within the same RTU for another ISaGRAF program) without having to load the entire RTU configuration.

Note: Saving a template for RLL configuration does not save the associated configuration for I/O points outside RLL.

3.12.4.1 Import Template

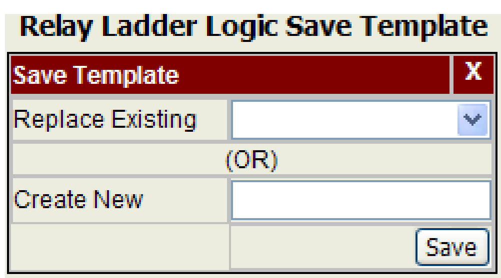
The Import function imports an RLL configuration in xml format as shown below. Choose from one of the existing files (if present) shown in the pull-down menu. If a new file has been created under Export, that file will also show up in the pull-down menu. When you set up another RTU, choose the Up/Download tab, select Templates from the File Type drop-down menu, and click “Send” (send files to RTU). The template you saved in the first RTU will be downloaded to the second RTU. Proceed to RLL under Configuration, select Import Template, then click Get as shown below.



3.12.4.2 Export Template

The Export function copies everything in the RLL configuration to an xml file. The Exp button exports a configuration in xml format from the RTU as a template. This template is stored in the RTU. When you choose Up/Download tab, select Templates from the File Type drop-down menu, and click on “Get” (get Templates from RTU), you will transfer these templates to your PC.

Choose from one of the existing file types (if present), or create a new xml file type. Click Save after your selection.



3.13 Reference Material - RLL Data Display

The RLL data display shows all the RLL points assigned under the Configuration tab. When you select the Display tab, then click on the RLL block, you will get a screen similar to Figure 3-116.

Figure 3-116 RLL Data Display

RLL Data Display

Type	Number	View
Analogs Inputs	4	View
Binary Inputs	4	View
Counters	4	View
Analog Outputs	4	View
Digital Outputs	4	View
SBO	4	

Back

Click on View to see the various point types that have been configured for RLL. The following section show the displays for all RLL point types.

3.13.1 Analog Inputs

Figure 3-117 RLL Analog Inputs Display

RLL Analog Inputs Display

Page1 of 1 Go To [ ] Go

Point	Point Name	Point Status	Point Value	Point Counts
0	RLL_ANALOG 0	F	100.000	null
1	RLL_ANALOG 1	F	100.000	null
2	RLL_ANALOG 2	F	100.000	null
3	RLL_ANALOG 3	F	100.000	null
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

Back

Point

The RLL point number.

Point Name

The point name assigned (or the default name accepted) during Configuration.

Point Status

The optional letter code(s) in the Point Status column means:

F analog point is marked failed (point is offline)

Point Value

The engineering unit value based on the EGU Min and EGU Max scaling assigned during Configuration

Point Counts

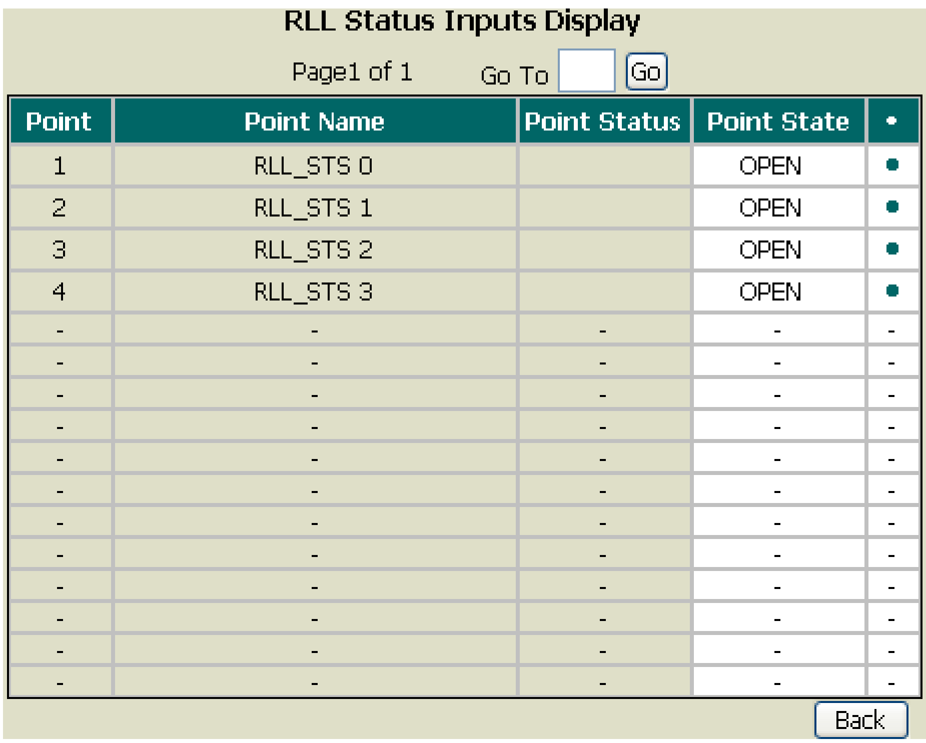
Counts are based on the C Min and C Max assigned during Configuration

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Data Display screen.

3.13.2 Binary Inputs

Figure 3-118 RLL Status Inputs Display



RLL Status Inputs Display

Page1 of 1 Go To Go

Point	Point Name	Point Status	Point State	•
1	RLL_STS 0		OPEN	•
2	RLL_STS 1		OPEN	•
3	RLL_STS 2		OPEN	•
4	RLL_STS 3		OPEN	•
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

Back

Point

The RLL point number.

Point Name

The point name assigned (or the default name accepted) during Configuration.

Point Status

The optional letter code(s) in the Point Status column means:

F analog point is marked failed (point is offline)

Point State

This will be either CLOSED or OPENED.

●

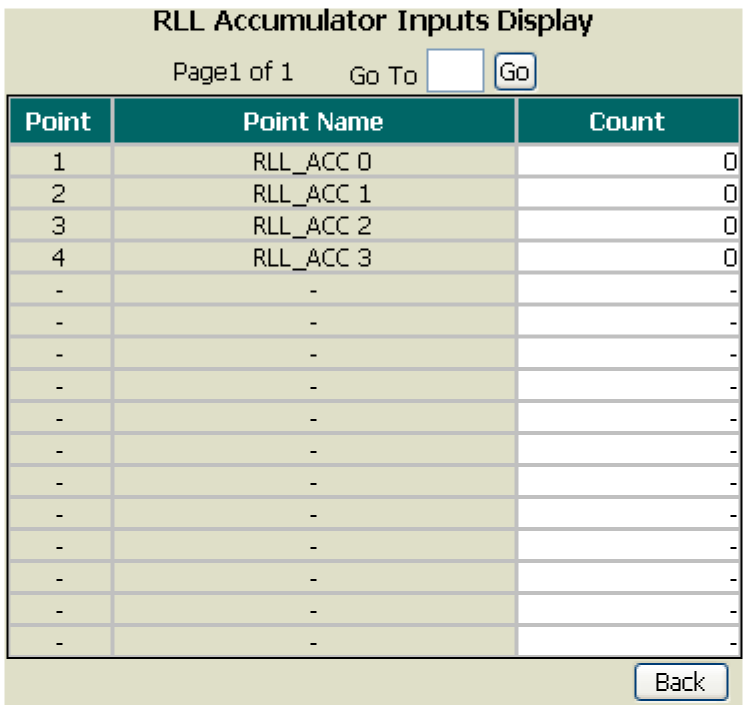
Displays a green dot for OPEN and a red dot for CLOSED.

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Data Display screen.

3.13.3 Counters

Figure 3-119 RLL Accumulator Inputs Display


RLL Accumulator Inputs Display

Page1 of 1 Go To Go

Point	Point Name	Count
1	RLL_ACC 0	0
2	RLL_ACC 1	0
3	RLL_ACC 2	0
4	RLL_ACC 3	0
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-
-	-	-

Back

Point

The RLL point number.

Point Name

The point name assigned (or the default name accepted) during Configuration.

Count

The maximum value is 4,294,967,295. The next count will force a rollover to zero.

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Data Display screen.

3.13.4 Analog Outputs

Figure 3-120 RLL Analog Outputs Display

RLL Analog Outputs Display

Page1 of 1 Go To [ ] Go

Point	Point Name	Point Status	Point Value
0	RLL_AO 0		100.000
1	RLL_AO 1		100.000
2	RLL_AO 2		100.000
3	RLL_AO 3		100.000
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-

Back

Point

The RLL point number.

Point Name

The point name assigned (or the default name accepted) during Configuration.

Point Status

The optional letter code(s) next to the point value means:

F - analog output point is marked failed (point is offline)

Point Value

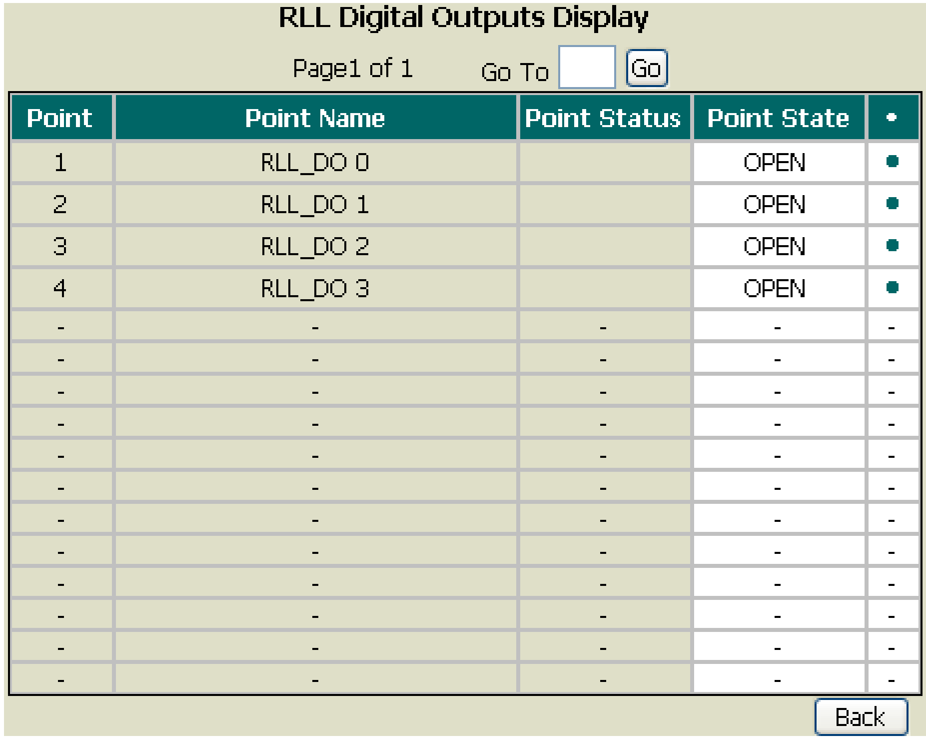
The engineering unit value based on the EGU Min and EGU Max scaling assigned during Configuration

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Data Display screen.

3.13.5 Digital Outputs

Figure 3-121 RLL Digital Outputs Display


RLL Digital Outputs Display

Page1 of 1 Go To [ ] Go

Point	Point Name	Point Status	Point State	•
1	RLL_DO 0		OPEN	•
2	RLL_DO 1		OPEN	•
3	RLL_DO 2		OPEN	•
4	RLL_DO 3		OPEN	•
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

Back

Point

The RLL point number.

Point Name

The point name assigned (or the default name accepted) during Configuration.

Point Status

The optional letter code(s) in the Point Status column means:

F analog point is marked failed (point is offline)

Point State

This will be either CLOSED or OPENED.

●

Displays a green dot for OPEN and a red dot for CLOSED.

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Data Display screen.

3.13.6 SBO

There is no display for SBOs.

3.14 Reference Material - RLL Command Output

RLL Analog Outputs, RLL Digital Outputs, and RLL SBOs may be commanded. Select the Command tab, then click on the RLL block. You will get a display as shown in Figure 3-122. Each of the three Command screens are shown in the following sections.

Figure 3-122 RLL Command

RLL Command

Type	Number	Command
Analogs Inputs	4	
Binary Inputs	4	
Counters	4	
Analog Outputs	4	Command
Digital Outputs	4	Command
SBO	4	Command

Back

3.14.1 Analog Outputs

Figure 3-123 RLL Analog Outputs Command

RLL Analog Outputs Command

Page 1 of 1 Go To Go

Point	Name	Range	Value	Operation
0	RLL_AO 0	100.000 to 101.000	100.000	Execute
1	RLL_AO 1	100.000 to 101.000	100.000	Execute
2	RLL_AO 2	100.000 to 101.000	100.000	Execute
3	RLL_AO 3	100.000 to 101.000	100.000	Execute

RLL_AO 0 : Success Back

Point

The RLL point number.

Name

The point name assigned (or the default name accepted) during Configuration.

Range

The EGU range as determined by the values chosen in the Configuration.

Value

Enter a value within the Range to exercise the point.

Operation

Click the Execute button to execute the command.

Status

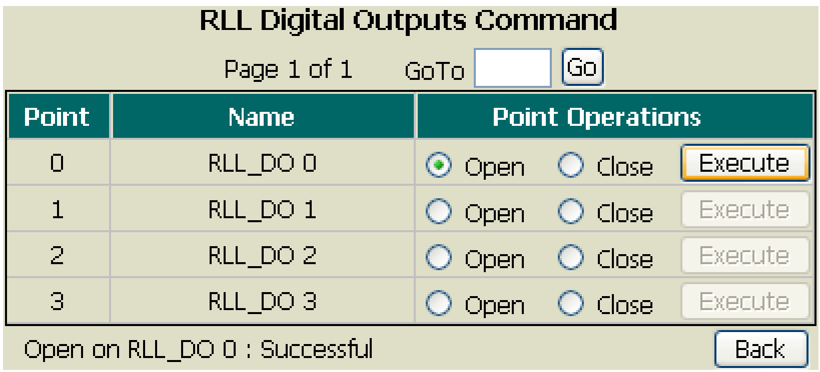
The Status message at the lower left will show the result of your command as shown in Figure 3-123.

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Command screen.

3.14.2 Digital Outputs

Figure 3-124 RLL Digital Outputs Command



Point

The RLL point number.

Name

The point name assigned (or the default name accepted) during Configuration.

Point Operations

Trip

Click the Trip button to select Trip.

Close

Click the Close button to select Close.

Execute

The Execute button will be active only if either the Trip or the Close has been selected. Once it is active, clicking the button will execute the action.

Status

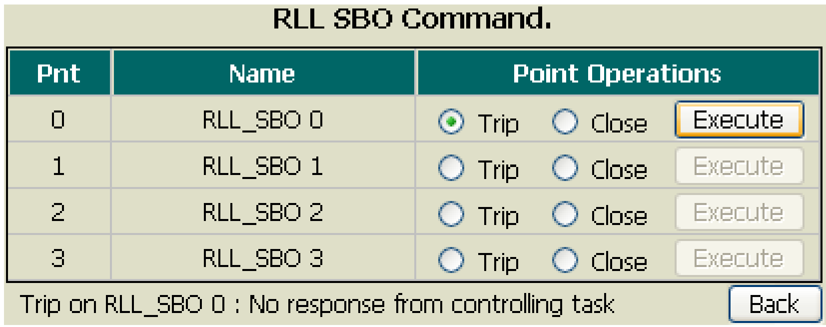
The Status message at the lower left will show the result of your command as shown in Figure 3-124.

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Command screen.

3.14.3 SBO

Figure 3-125 RLL SBO Command



Point

The RLL point number.

Name

The point name assigned (or the default name accepted) during Configuration.

Point Operations

Trip

Click the Trip button to select Trip.

Close

Click the Close button to select Close.

Execute

The Execute button will be active only if either the Trip or the Close has been selected. Once it is active, clicking the button will execute the action.

Status

The Status message at the lower left will show the result of your command as shown in Figure 3-125.

Navigation

Click <<Prev to navigate to the previous 16 points, if applicable. Page *n* of *n* tells you which page (of a total number of pages) you are on. Go to a specific page by typing in the page number, then click the Go button. Click Next>> to go to the next 16 points, if applicable. Click the Back button to go back to the Command screen.

CHAPTER 4

Programming Principles & Examples

4.1 Introduction

The following examples are in the form of Function Block Diagram (FBD) programs. Although the ISaGRAF program supports several other programming languages, Telvent supports only Ladder Diagram (LD) and Function Block Diagram (FBD). A simple example of the LD program is shown in Chapter 3, Operation.

The following programs are only examples. Telvent makes no warranty, either expressed or implied, to the usefulness or fitness of these examples for any purpose.

4.2 MTU/RTU Programming Model

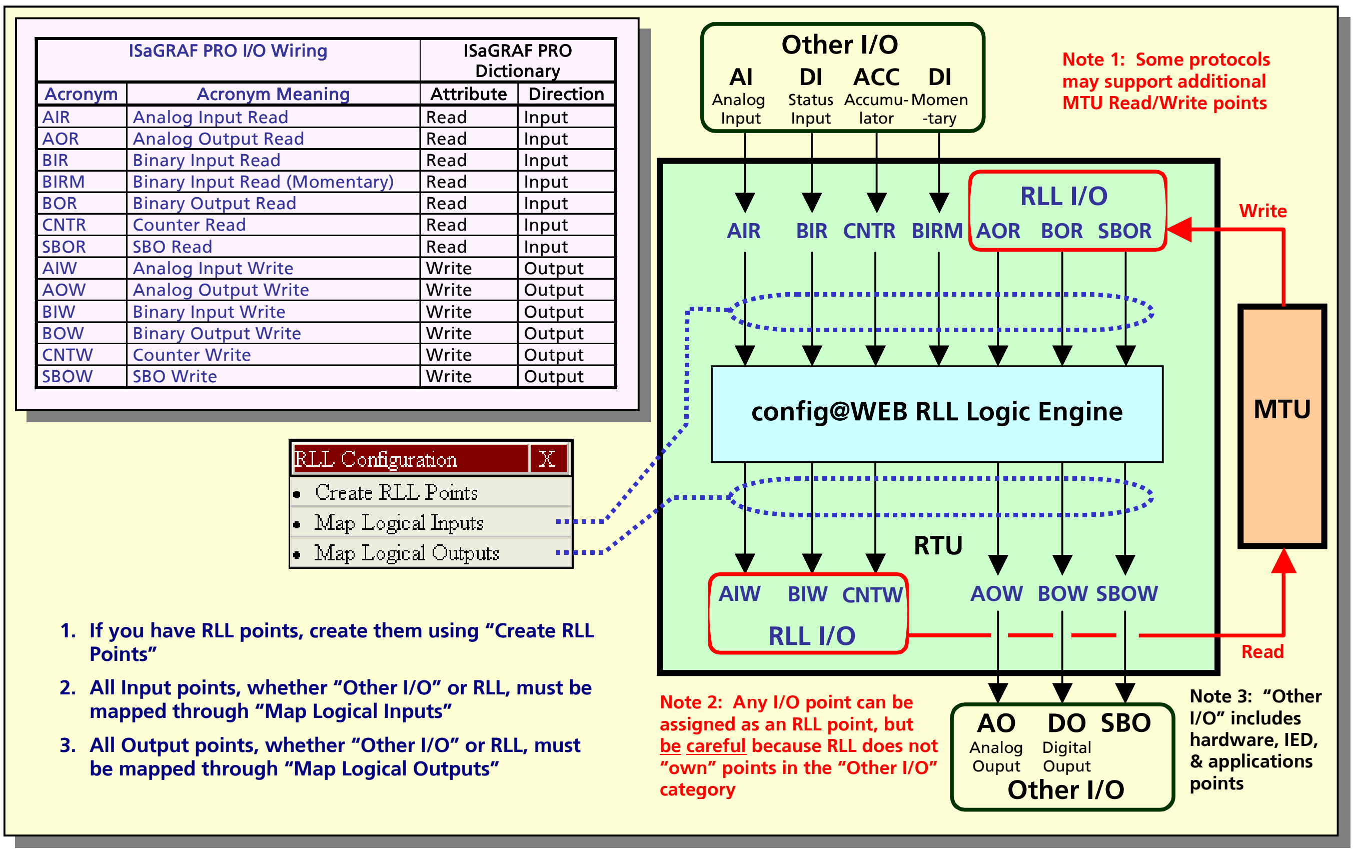
Figure 4-1 shows how the MTU can read certain points from RLL and write certain points to the RLL program. Notice that points must be "Write" (AIW, BIW, CNTW) for the MTU to read those points. Conversely, the points must be "Read" (AOR, BOR, SBOR) for the MTU to write to those points.

Proceed as follows:

1. If you have RLL points, create them using "Create RLL Points"
2. All Input points, whether "Other I/O" or RLL, must be mapped through "Map Logical Inputs"
3. All Output points, whether "Other I/O" or RLL, must be mapped through "Map Logical Outputs"

The next few Figures elaborate on these steps.

Figure 4-1 MTU/RTU Programming Model



ISaGRAF PRO I/O Wiring		ISaGRAF PRO Dictionary	
Acronym	Acronym Meaning	Attribute	Direction
AIR	Analog Input Read	Read	Input
AOR	Analog Output Read	Read	Input
BIR	Binary Input Read	Read	Input
BIRM	Binary Input Read (Momentary)	Read	Input
BOR	Binary Output Read	Read	Input
CNTR	Counter Read	Read	Input
SBOR	SBO Read	Read	Input
AIW	Analog Input Write	Write	Output
AOW	Analog Output Write	Write	Output
BIW	Binary Input Write	Write	Output
BOW	Binary Output Write	Write	Output
CNTW	Counter Write	Write	Output
SBOW	SBO Write	Write	Output

Figure 4-2 Mapping RLL Points

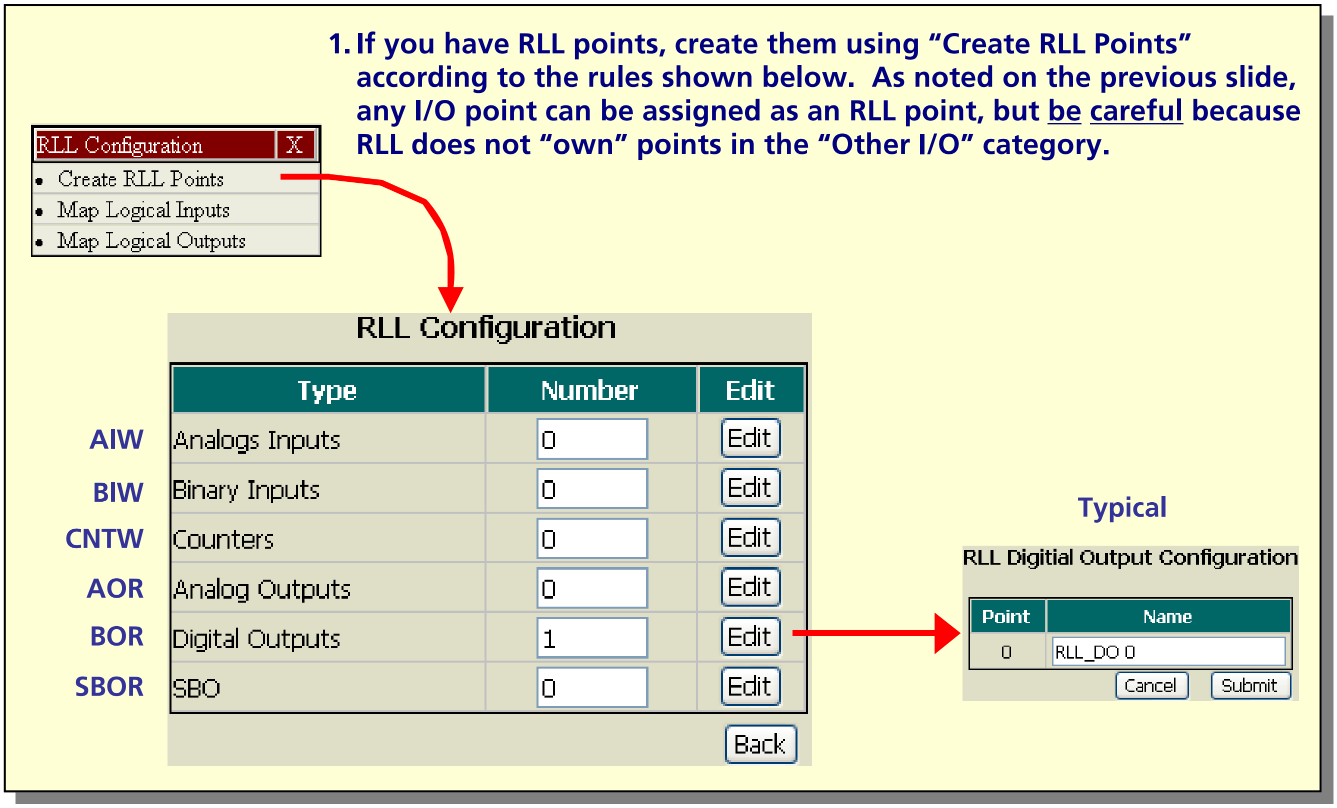
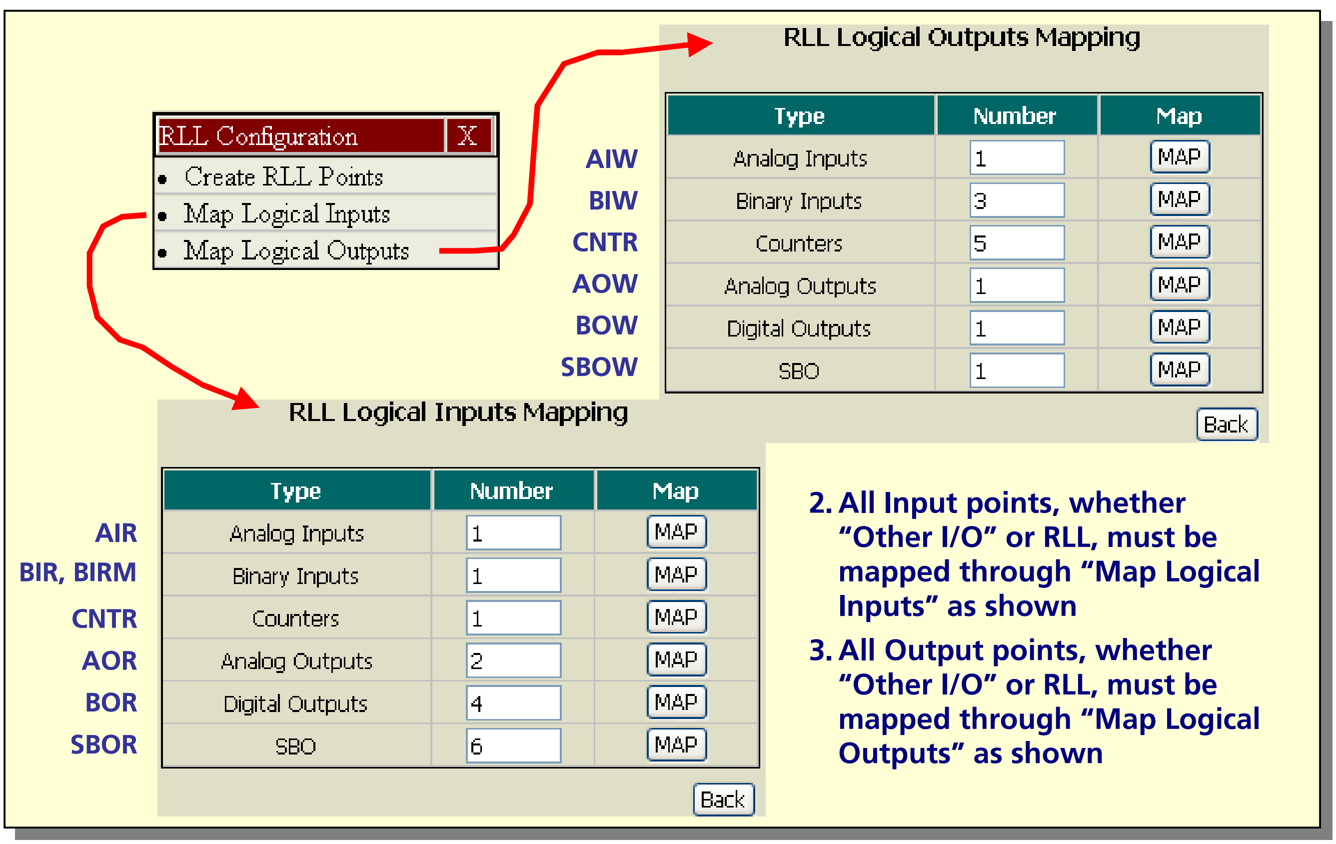


Figure 4-3 Mapping All I/O Points

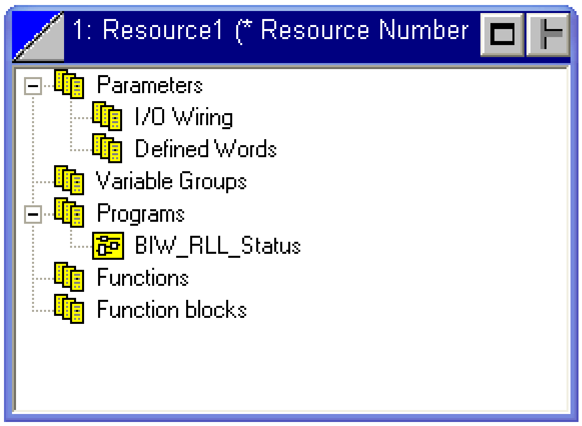


4.2.1 BIRM, BIR Notes

As shown in Figure 4-3, both BIRM and BIR must be mapped to the same point. For example, if you created a program that needs to "see" if a status point changes during the ISaGRAF cycle time (1 second by default), you would create a MCD (Momentary Change Detect) variable, then wire that variable to both a BIR and a BIRM in order to have both a static status point and a momentary status point. When you map a single Binary Input point on the RLL Logical Inputs Mapping (see Figure 4-3), that status input to your program will be used for both MCD and regular status.

4.3 Hardware AI to RLL (Pseudo) Points

Figure 4-4 Link Architecture View

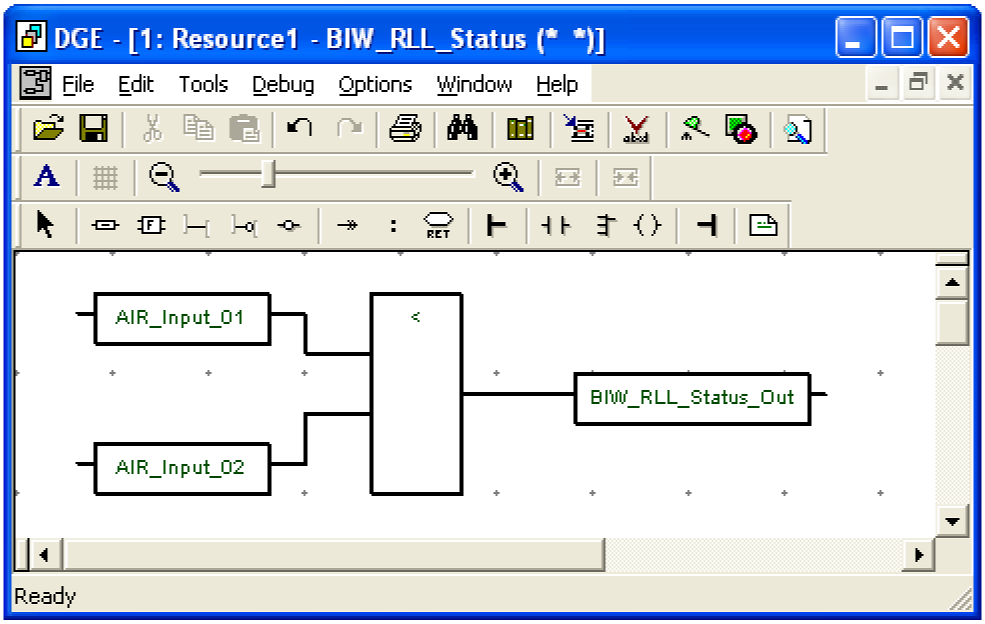


4.3.1 Program

This example shows how to create an RLL (pseudo) status point that can be read by the MTU. The program compares two analogs. The logic block is a "less than." When AIR_Input_02 equals or exceeds AIR_Input_01, the output turns True and BIW_RLL_Status_Out changes from Open to Close. BIW_RLL_Status_Out can be read by the MTU.

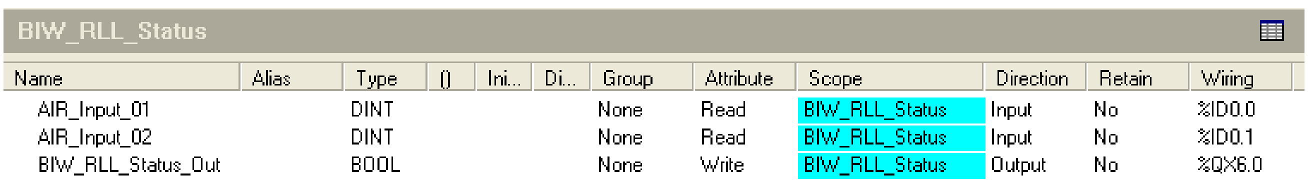
As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-5 BIW_RLL_Status



4.3.2 Variables

Figure 4-6 BIW_RLL_Status Program Variables


BIW_RLL_Status

Name	Alias	Type	()	Ini...	Di...	Group	Attribute	Scope	Direction	Retain	Wiring
AIR_Input_01		DINT				None	Read	BIW_RLL_Status	Input	No	%ID0.0
AIR_Input_02		DINT				None	Read	BIW_RLL_Status	Input	No	%ID0.1
BIW_RLL_Status_Out		BOOL				None	Write	BIW_RLL_Status	Output	No	%QX6.0

4.3.3 Wiring

Figure 4-7 BIW_RLL_Status Program Wiring for Analog Inputs

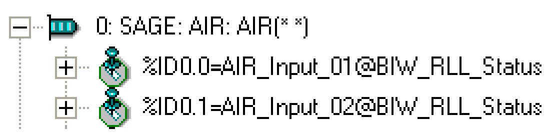
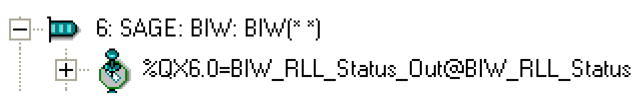


Figure 4-8 BIW_RLL_Status Program Wiring for RLL Status Output



4.3.4 RTU Mapping

Compile the program and download it to the RTU. Next, the RTU must be configured. Figure 4-9 shows how to map the RLL point.

Figure 4-9 RTU Configuration – Create RLL Point

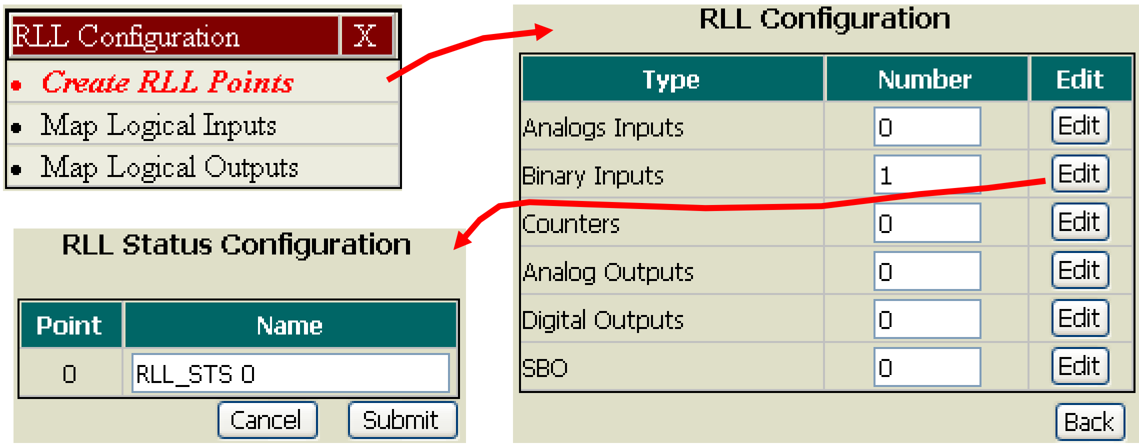


Figure 4-10 show how to map the logical output from the RLL point created in Figure 4-9.

Figure 4-10 RTU Configuration – Map Logical Outputs

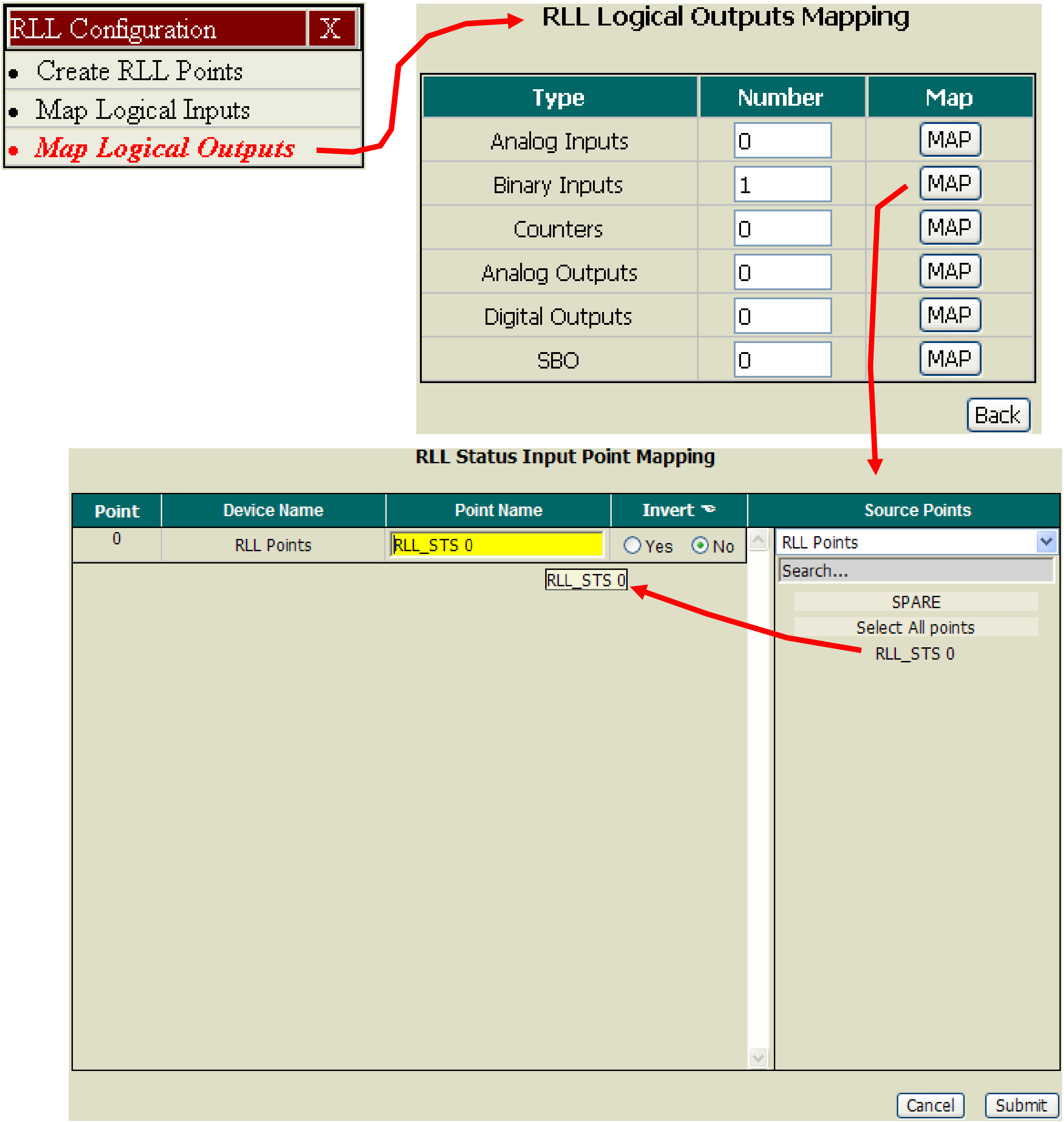
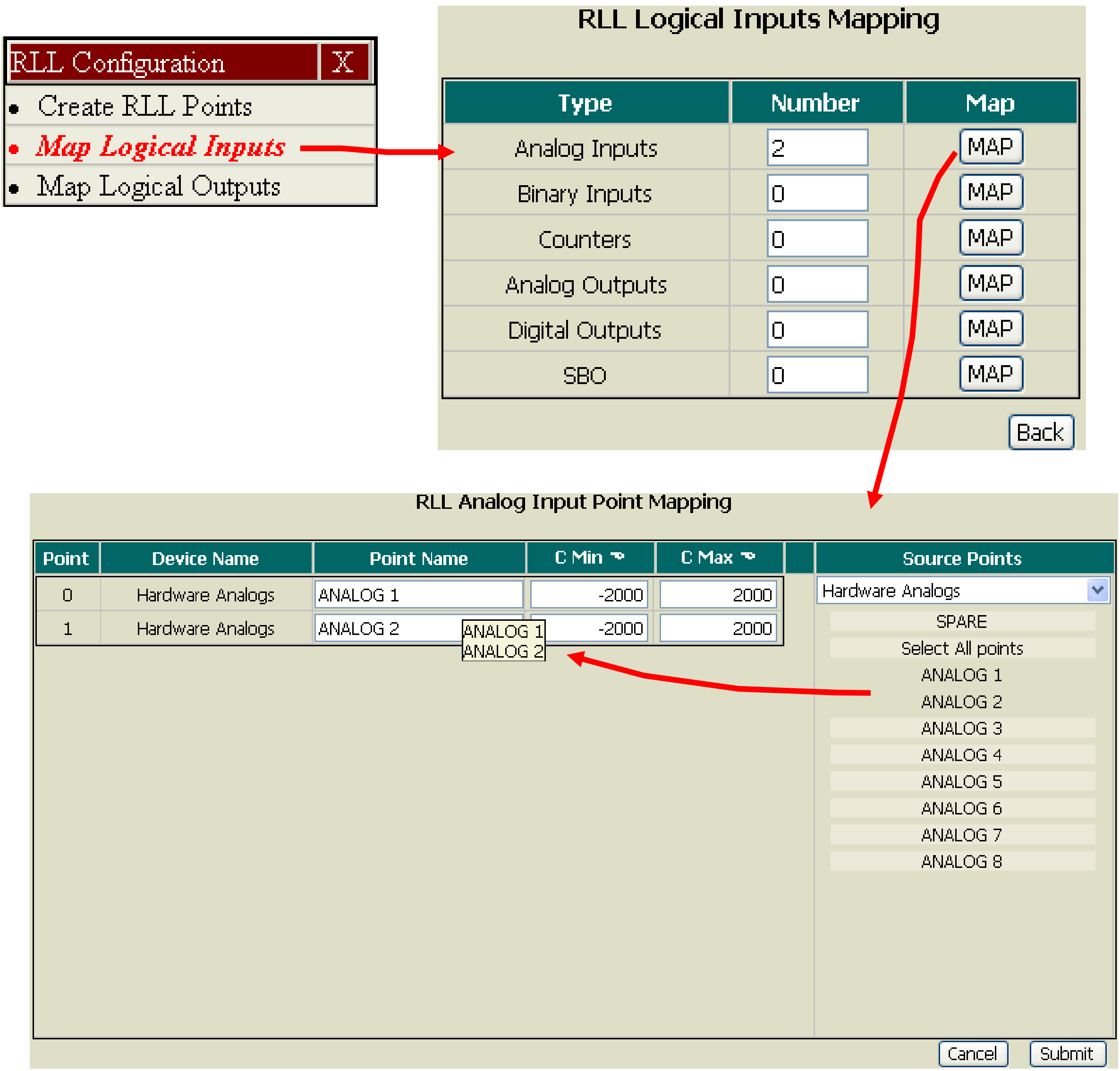


Figure 4-11 Shows how to map the two analog input points from the hardware analog points.

Figure 4-11 RTU Configuration – Map Logical Inputs



This completes the configuration of the RTU. As always, you must reset the RTU before the new configuration takes effect.

4.3.5 RTU Display

To see the results of the BIW_RLL_Status program, display the analogs as shown in Figure 4-12. When Analog 2 equals or surpasses Analog 1, the pseudo Status changes from Open to Close.

Figure 4-12 Monitoring the Analogs

Analog Inputs (AI) Display

Page1 of 1 Go To [ ] Go

Point	Point Name	Point Status	Point Value
1	ANALOG 1		1.891
2	ANALOG 2		1.749
3	ANALOG 3		0.000
4	ANALOG 4		0.000
5	ANALOG 5		0.000
6	ANALOG 6		0.000
7	ANALOG 7		0.000
8	ANALOG 8		0.000
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-

Back

However, the Analog Display shows only the inputs to the program. Display the pseudo status point as shown below. Figure 4-14 shows the results if Analog 2 is equal to, or less than, Analog 1. Figure 4-15 shows the results if Analog 2 is greater than Analog 1.

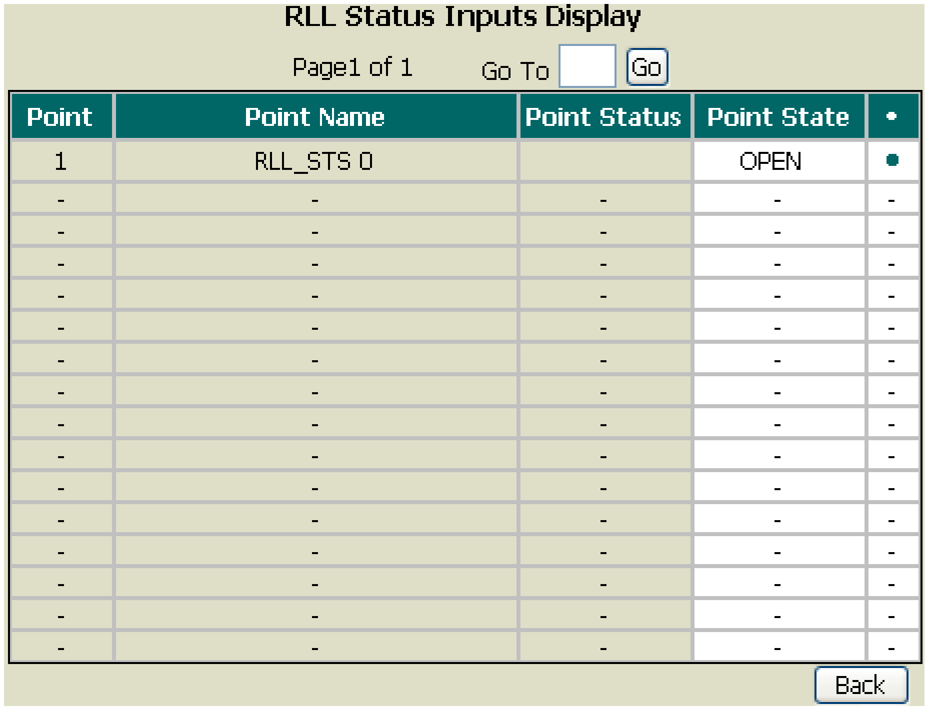
Figure 4-13 RLL Data Display

RLL Data Display

Type	Number	View
Analogs Inputs	0	View
Binary Inputs	1	View
Counters	0	View
Analog Outputs	0	View
Digital Outputs	0	View
SBO	0	

Back

Figure 4-14 Displaying the RLL Status with Analog 2 Less Than Analog 1



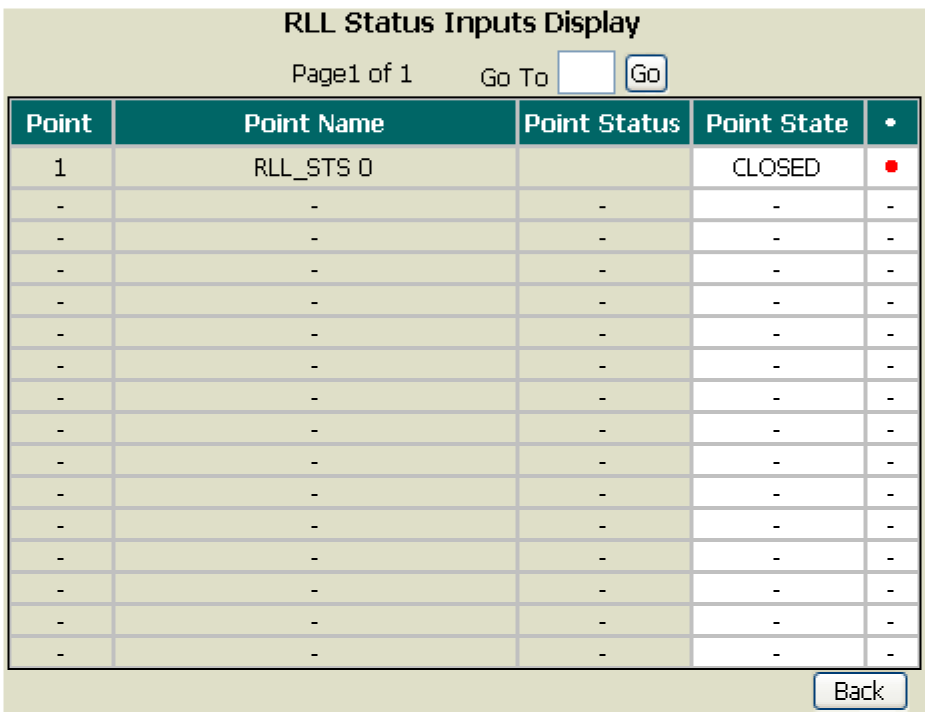
RLL Status Inputs Display

Page1 of 1 Go To [Go]

Point	Point Name	Point Status	Point State	•
1	RLL_STS 0		OPEN	•
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

Back

Figure 4-15 Displaying the RLL Status with Analog 2 Greater Than Analog 1



RLL Status Inputs Display

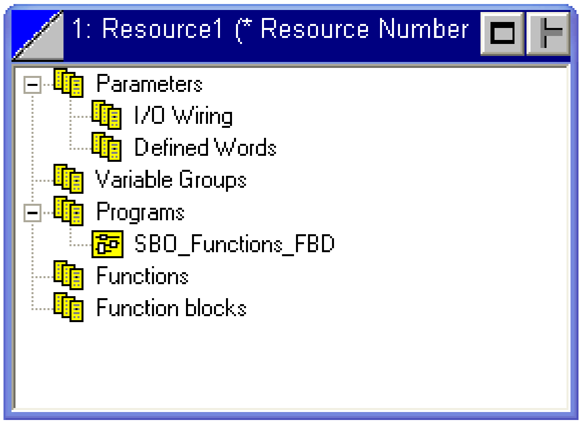
Page1 of 1 Go To [Go]

Point	Point Name	Point Status	Point State	•
1	RLL_STS 0		CLOSED	•
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

Back

4.4 Using Input SBO RLL (Pseudo) Points

Figure 4-16 Link Architecture View



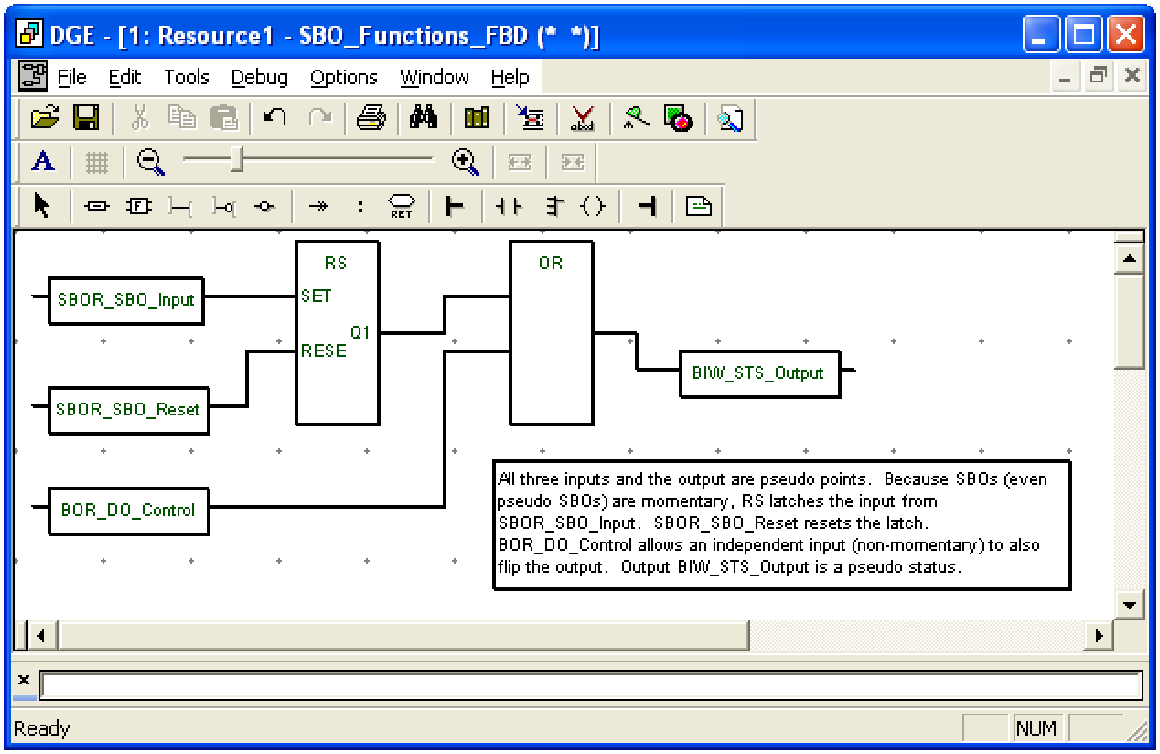
4.4.1 Program

This example shows how to create Input SBO RLL (pseudo) points that can be written to by the MTU. Because SBOs are momentary (even pseudo SBOs), we use a latch (reset dominant bistable). SBOR_SBO_Input sets the latch. The latch stays set until it is reset by SBOR_SBO_Reset. The OR gate adds another logic element, allowing either SBOR_SBO_Input or BOR_DO_Control to change the output. The logic output is a pseudo status point, BIW_STS_Output.

All three inputs may be written to by an MTU and the output may be read by an MTU.

As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-17 SBO_Functions_FBD Program



4.4.2 Variables

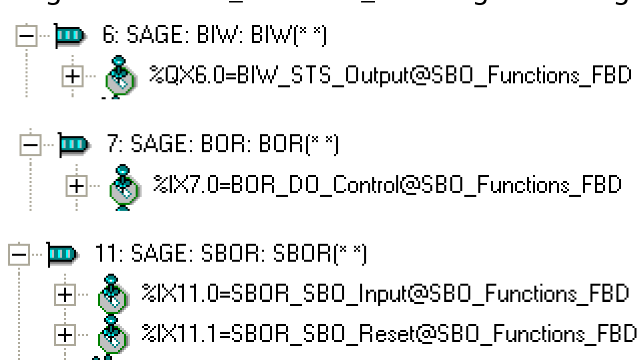
Figure 4-18 SBO_Functions_FBD Program Variables

SBO_Functions_FBD

Name	Alias	Type	()	Ini...	Di...	Group	Attribute	Scope	Direction	Retain	Wiring
SBOR_SBO_Input		BOOL				None	Read	SBO_Functions_FBD	Input	No	%IX11.0
BIW_STS_Output		BOOL				None	Write	SBO_Functions_FBD	Output	No	%QX6.2
SBOR_SBO_Reset		BOOL				None	Read	SBO_Functions_FBD	Input	No	%IX11.1
BOR_DO_Control		BOOL				None	Read	SBO_Functions_FBD	Input	No	%IX7.0

4.4.3 Wiring

Figure 4-19 SBO_Functions_FBD Program Wiring



4.4.4 RTU Mapping

Figure 4-20 SBO_Functions_FBD Program RTU RLL Point Mapping

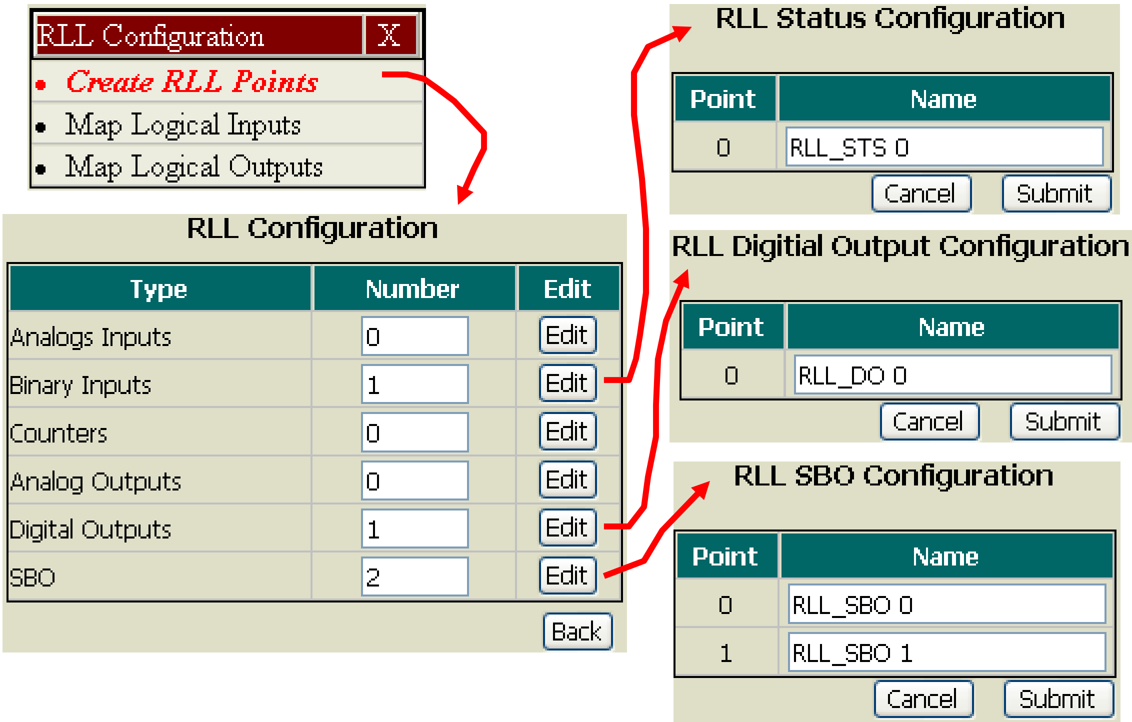


Figure 4-21 SBO_Functions_FBD Program RTU Input Logic Point Mapping

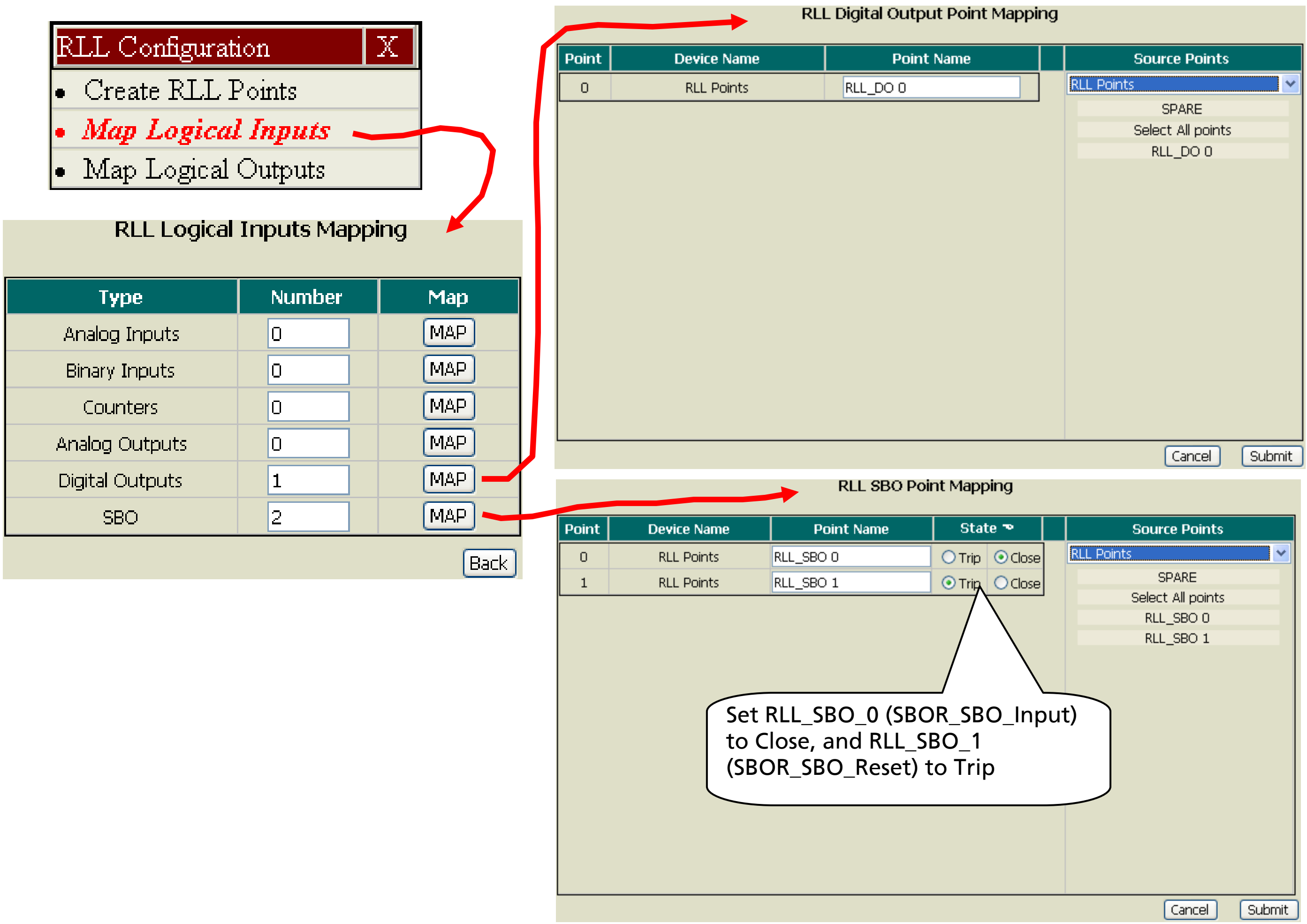
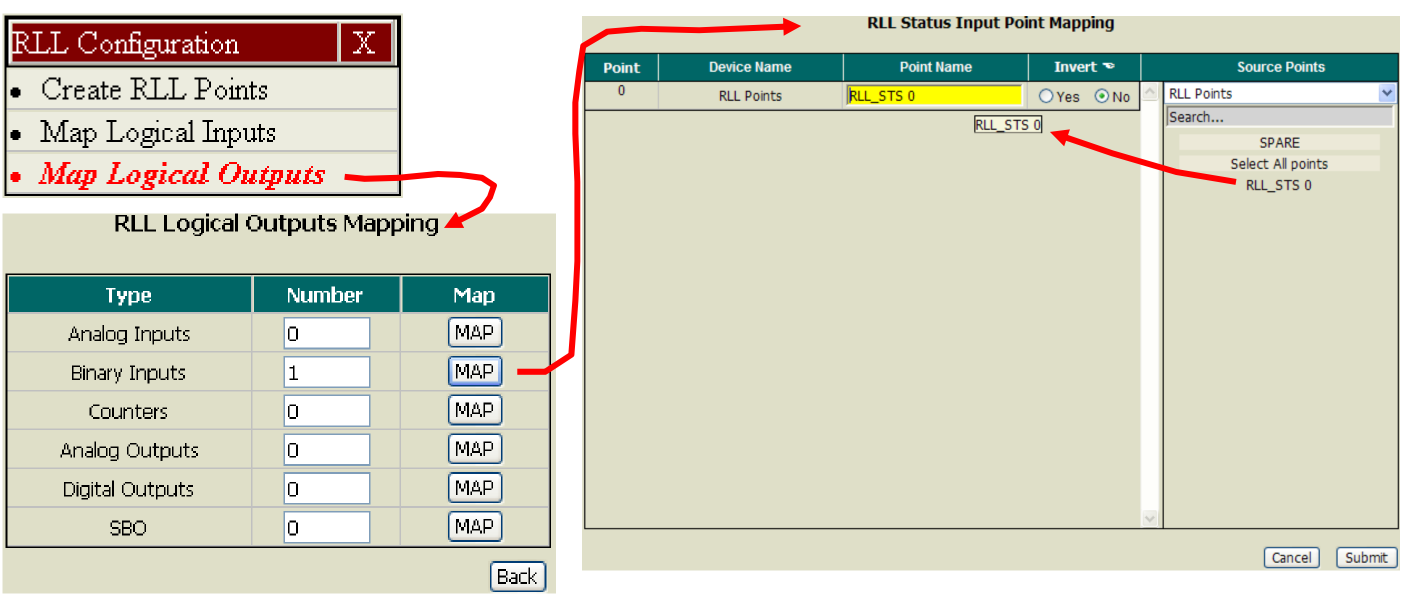


Figure 4-22 SBO_Functions_FBD Program RTU Output Logic Point Mapping



This concludes the configuration of the SBO_Functions_FBD Program. To see the program in operation you must Command the various inputs (DO & SBO), then observe the RLL Status output.

4.4.5 RTU Display

Figure 4-23 Commanding SBOR_SBO_Input

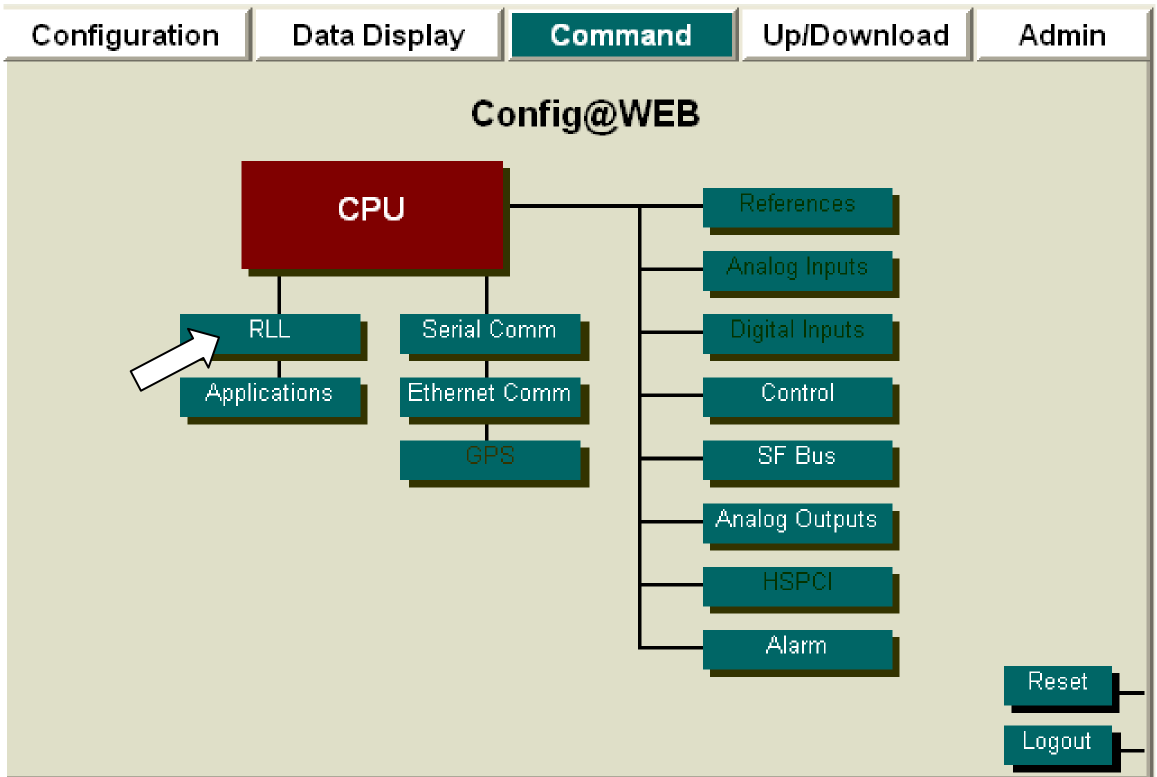
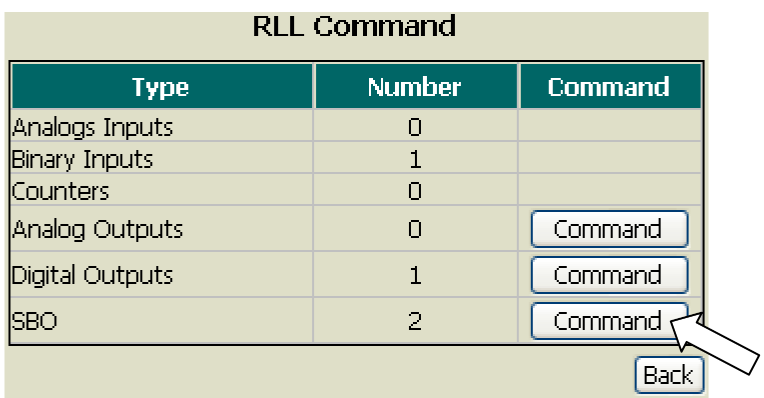


Figure 4-24 Commanding SBOR_SBO_Input


RLL Command

Type	Number	Command
Analogs Inputs	0	
Binary Inputs	1	
Counters	0	
Analog Outputs	0	Command
Digital Outputs	1	Command
SBO	2	Command

Back

To execute the SBOR_SBO_Input command properly, you must match the Trip/Close that was set up in Figure 4-21. In this case, it was Close, as shown in Figure 4-25.

Figure 4-25 Commanding SBOR_SBO_Input

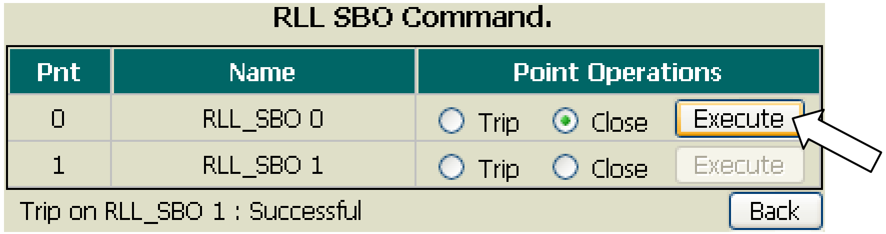
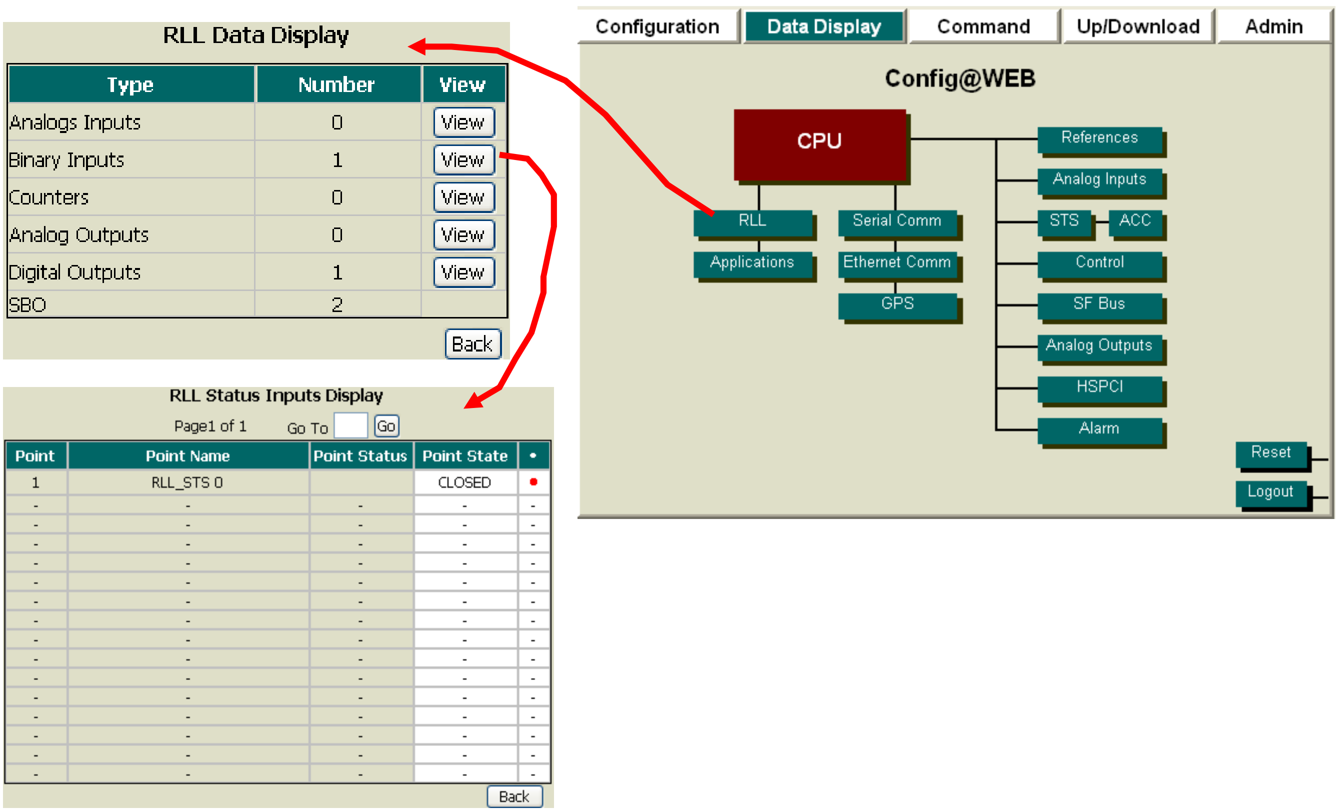


Figure 4-26 Resulting RLL Status Inputs Display



Before the cycle can be executed again, the logic must be reset. SBOR_SBO_Reset corresponds to the second RLL SBO command as shown below. As with the Input command, the Trip/Close must match the setting in Figure 4-21. In this case, we must use Trip, as shown Figure 4-27.

Figure 4-27 Commanding SBOR_SBO_Reset

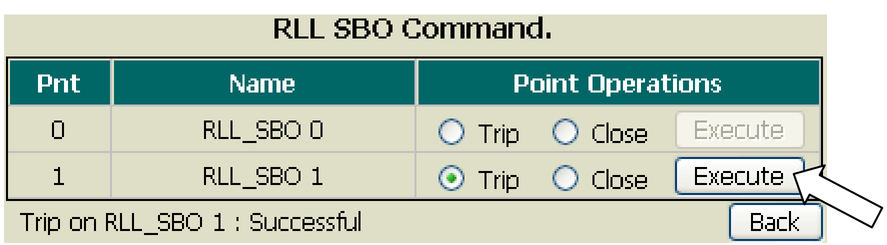
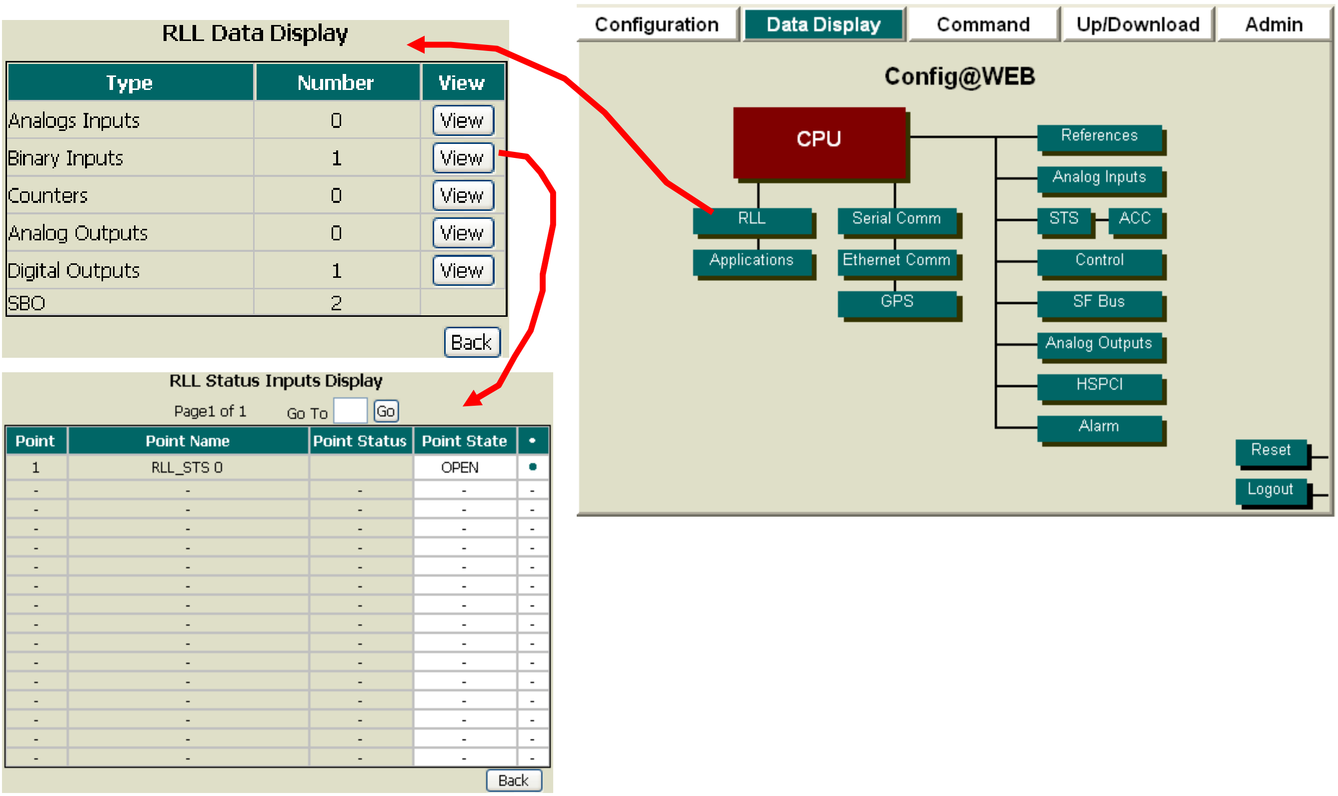


Figure 4-28 Resulting RLL Status Inputs Display



The last part of the SBO_Function_FBD program is the BOR_DO_Control. This RLL DO input provides a way to bypass the Set/Reset block. The RTU command is shown below.

Figure 4-29 Commanding the BOR_DO_Control

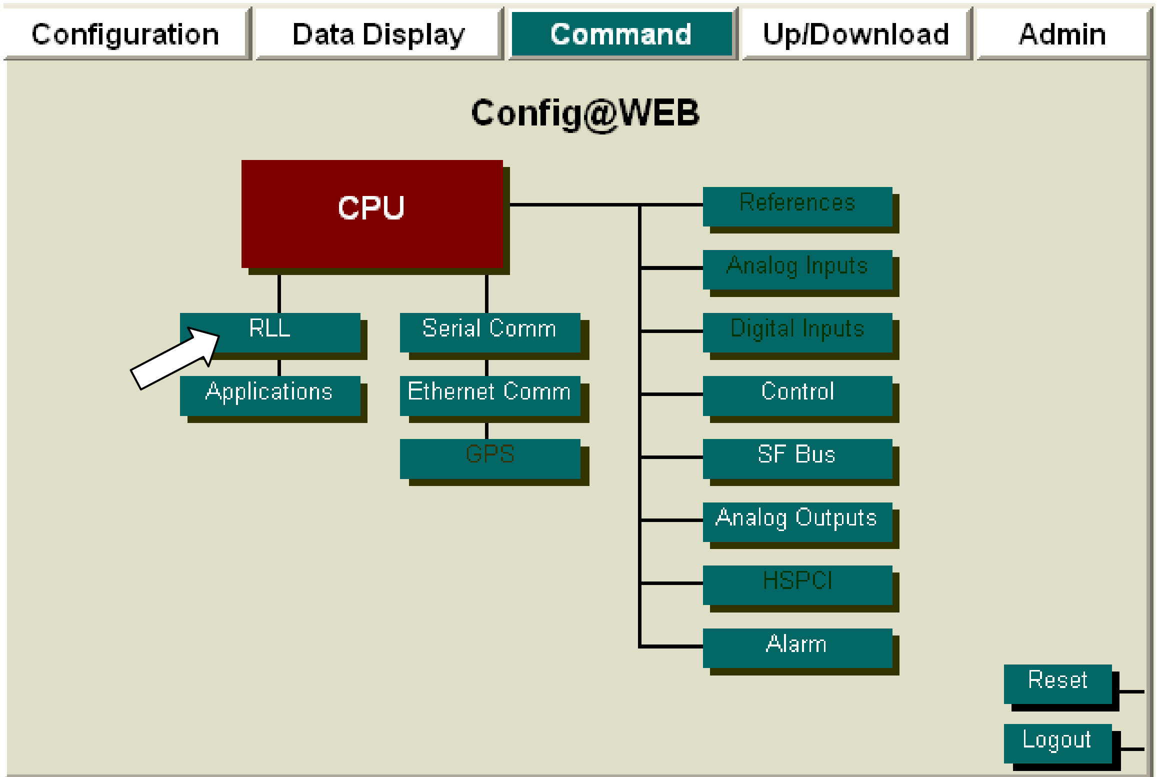
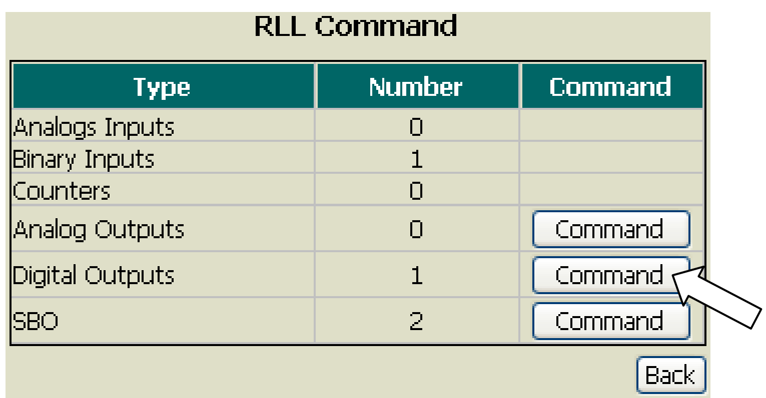
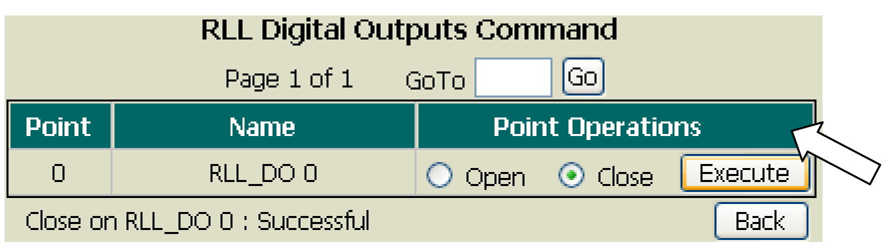


Figure 4-30 Commanding the BOR_DO_Control



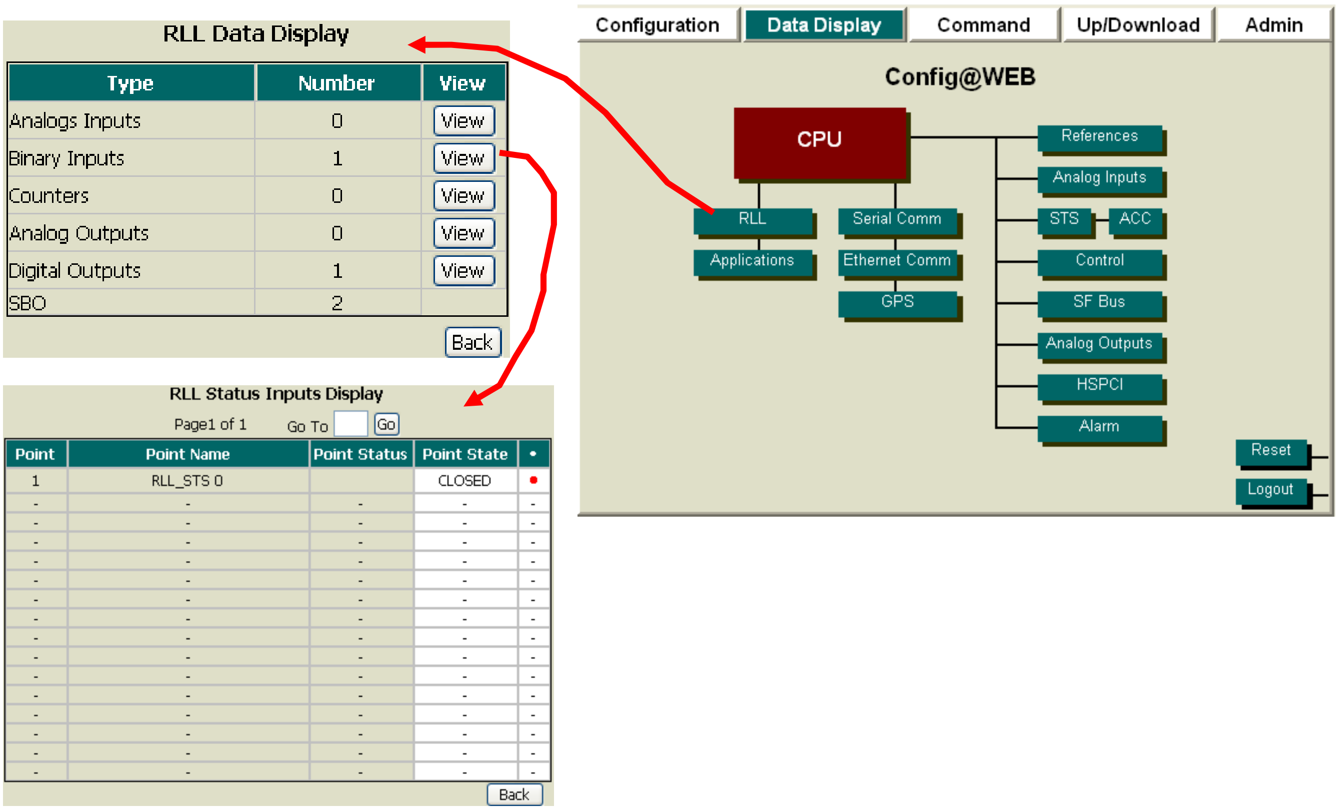
To execute the BOR_DO_Control command properly, you must Close the point, as shown in Figure 4-31.

Figure 4-31 Commanding the BOR_DO_Control



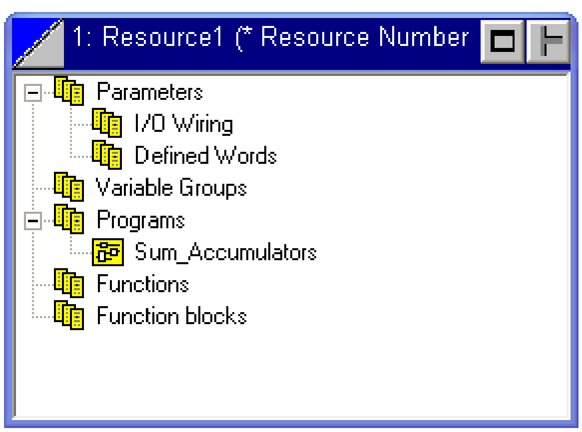
Unlike an SBO momentary input that has to be latched, the DO input that closes the status point will stay closed. If you need the status point to open again, you must open the RLL_DO_0 point.

Figure 4-32 Resulting RLL Status Inputs Display



4.5 Summing Accumulator Points

Figure 4-33 Link Architecture View

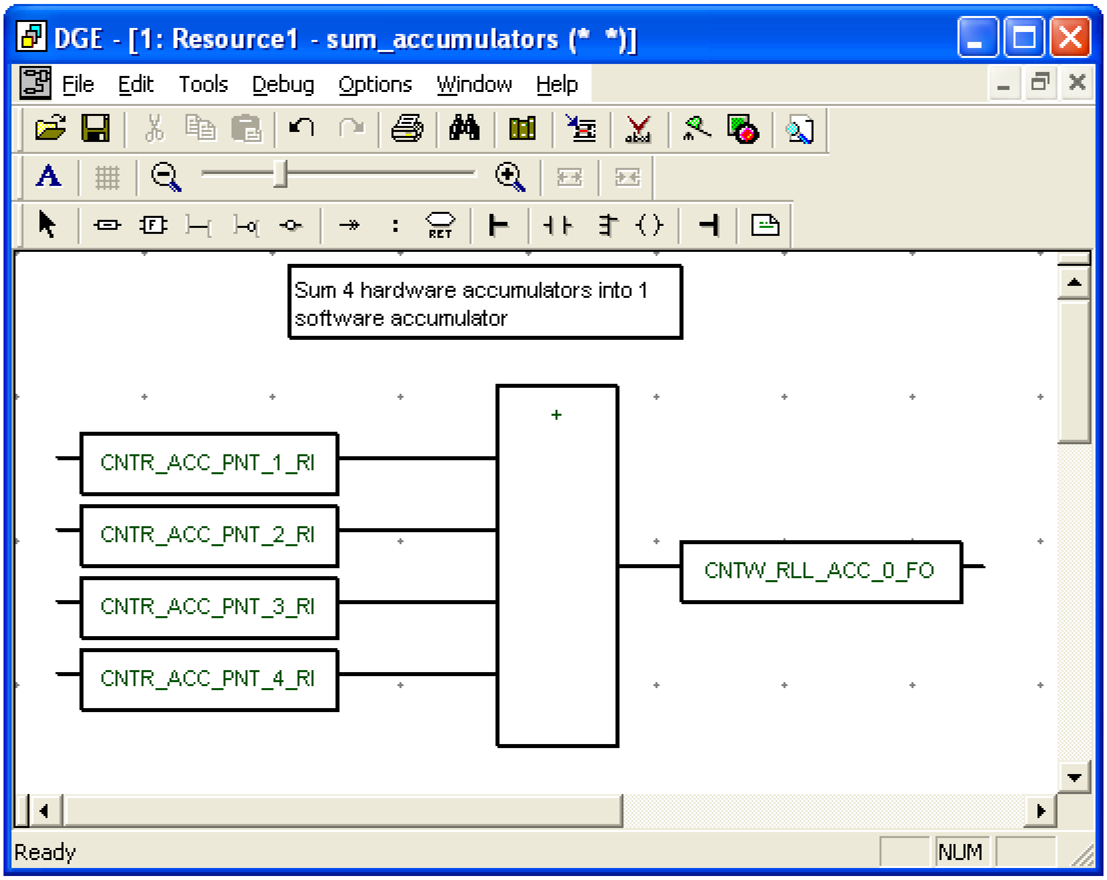


4.5.1 Program

This example shows how to sum four hardware accumulators to one pseudo accumulator point. The pseudo accumulator point may be read by the MTU.

As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-34 Sum_accumulators Program



4.5.2 Variables

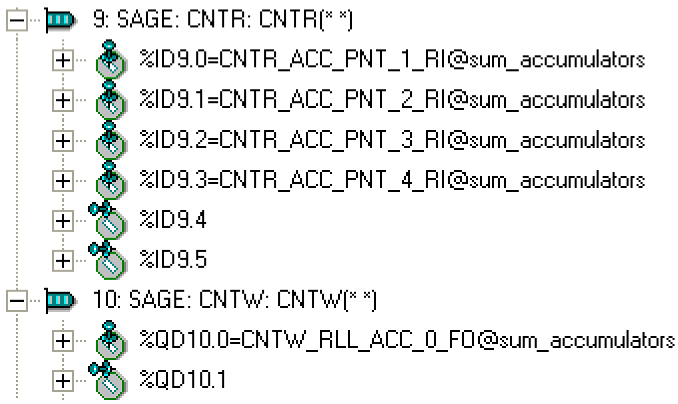
Figure 4-35 Sum_accumulators Program Variables

sum_accumulators

Name	Alias	Type	()	Ini...	Di...	Group	Attribute	Scope	Direction	Retain	Wiring	Address
CNTR_ACC_PNT_1_RI		DINT				None	Read	sum_accumulators	Input	No	%ID9.0	
CNTR_ACC_PNT_2_RI		DINT				None	Read	sum_accumulators	Input	No	%ID9.1	
CNTR_ACC_PNT_3_RI		DINT				None	Read	sum_accumulators	Input	No	%ID9.2	
CNTR_ACC_PNT_4_RI		DINT				None	Read	sum_accumulators	Input	No	%ID9.3	
CNTW_RLL_ACC_0_FO		DINT				None	Free	sum_accumulators	Output	No	%QD10.0	

4.5.3 Wiring

Figure 4-36 Sum_accumulators Program Wiring



4.5.4 RTU Mapping

Figure 4-37 Sum_accumulators Program RTU RLL Point Mapping

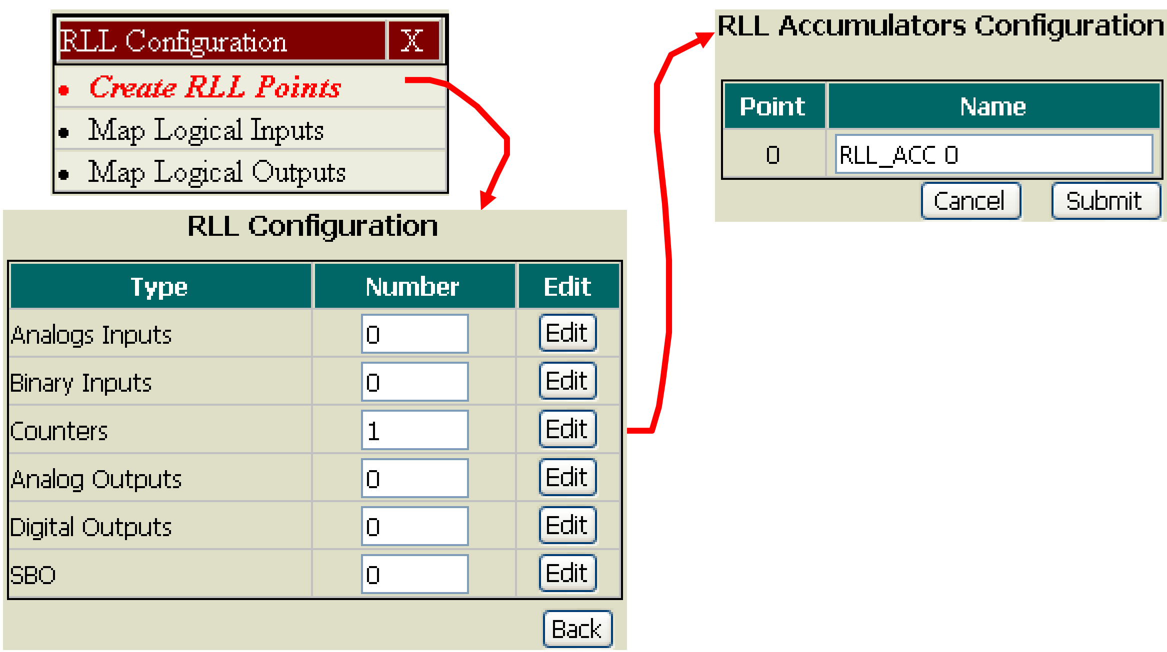


Figure 4-38 Sum_accumulators Program RTU Input Logic Point Mapping

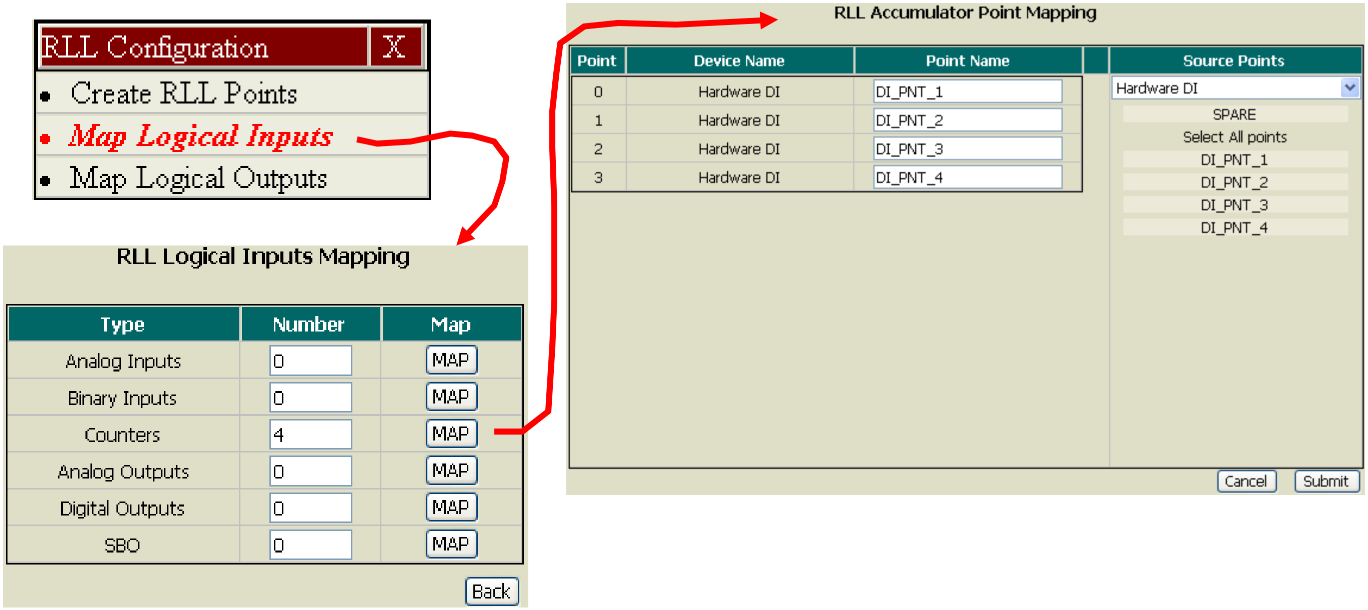
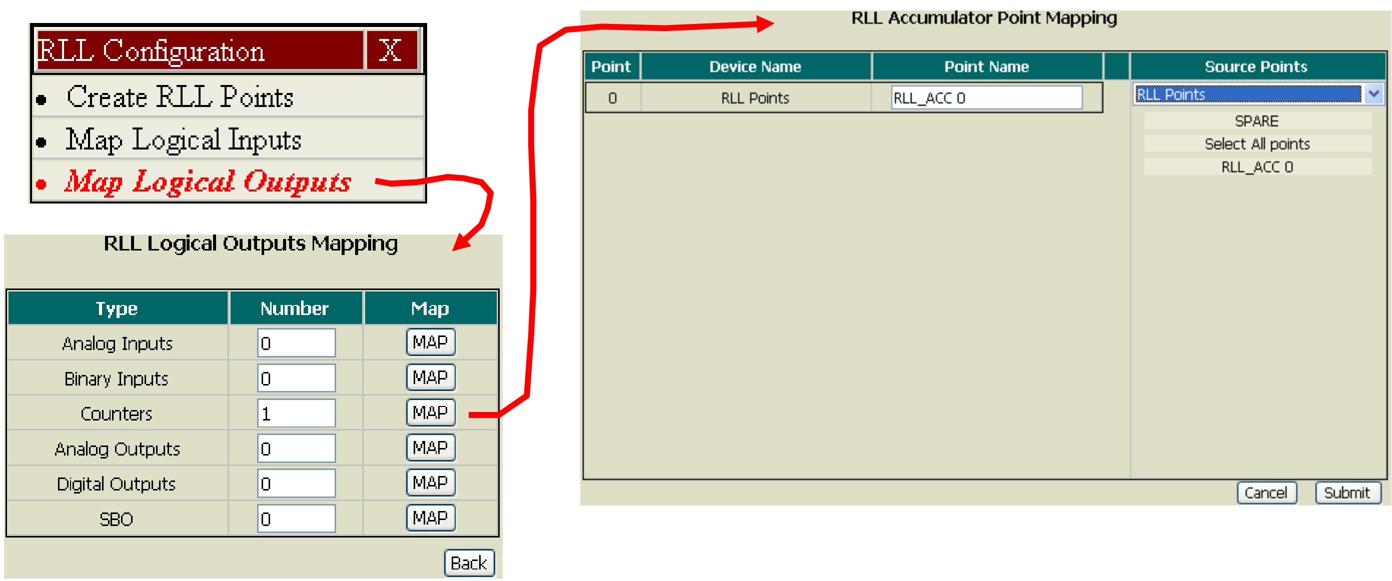


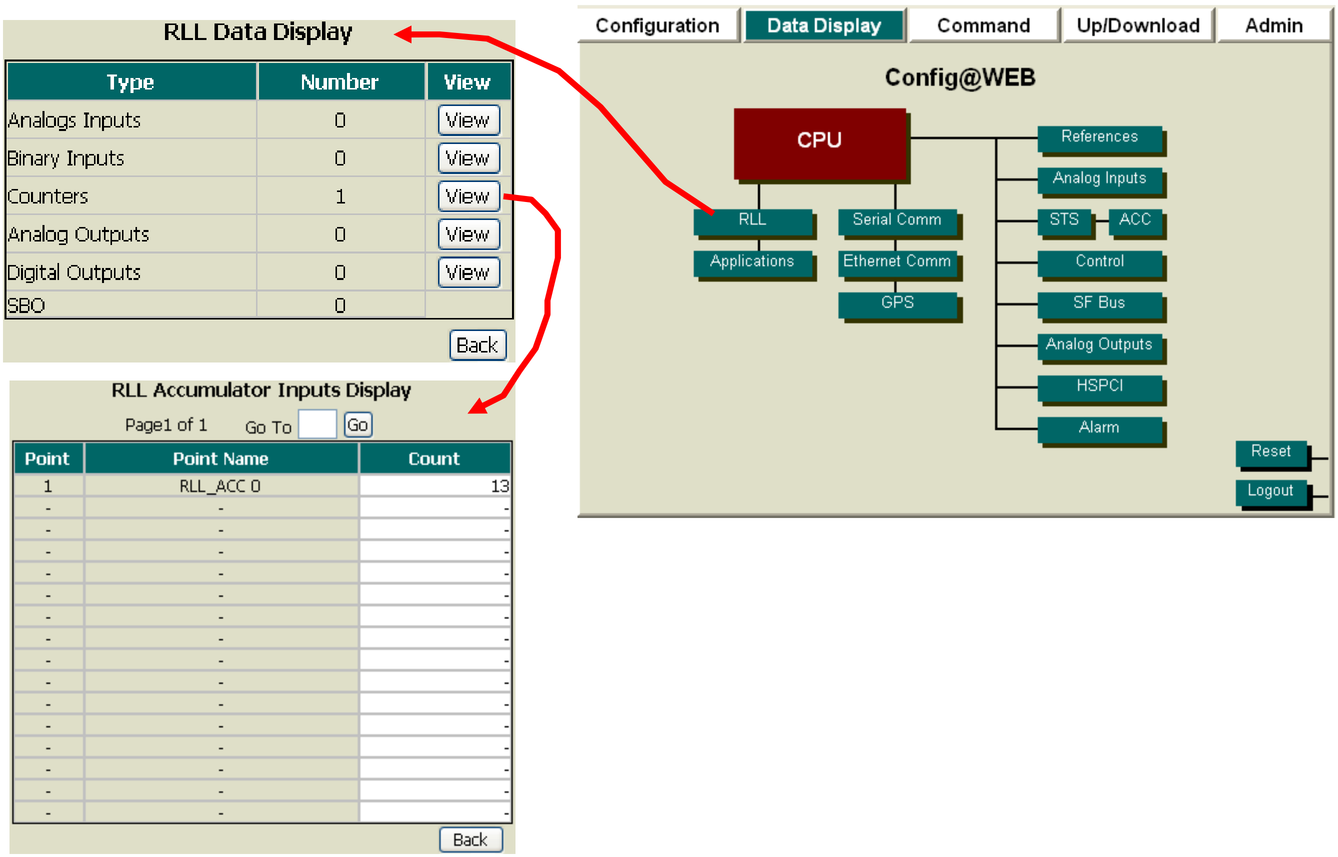
Figure 4-39 Sum_accumulators Program RTU Output Logic Point Mapping



This concludes the configuration of the Sum_accumulators Program. To see the program in operation you must exercise the hardware accumulators, then observe the RLL accumulator output.

4.5.5 RTU Display

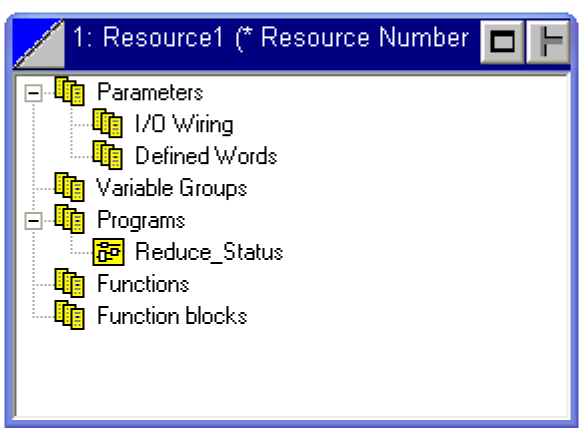
Figure 4-40 Resulting RLL Accumulator Display



The RLL_ACC 0 point will sum all four hardware accumulators.

4.6 Reducing Status Points

Figure 4-41 Link Architecture View

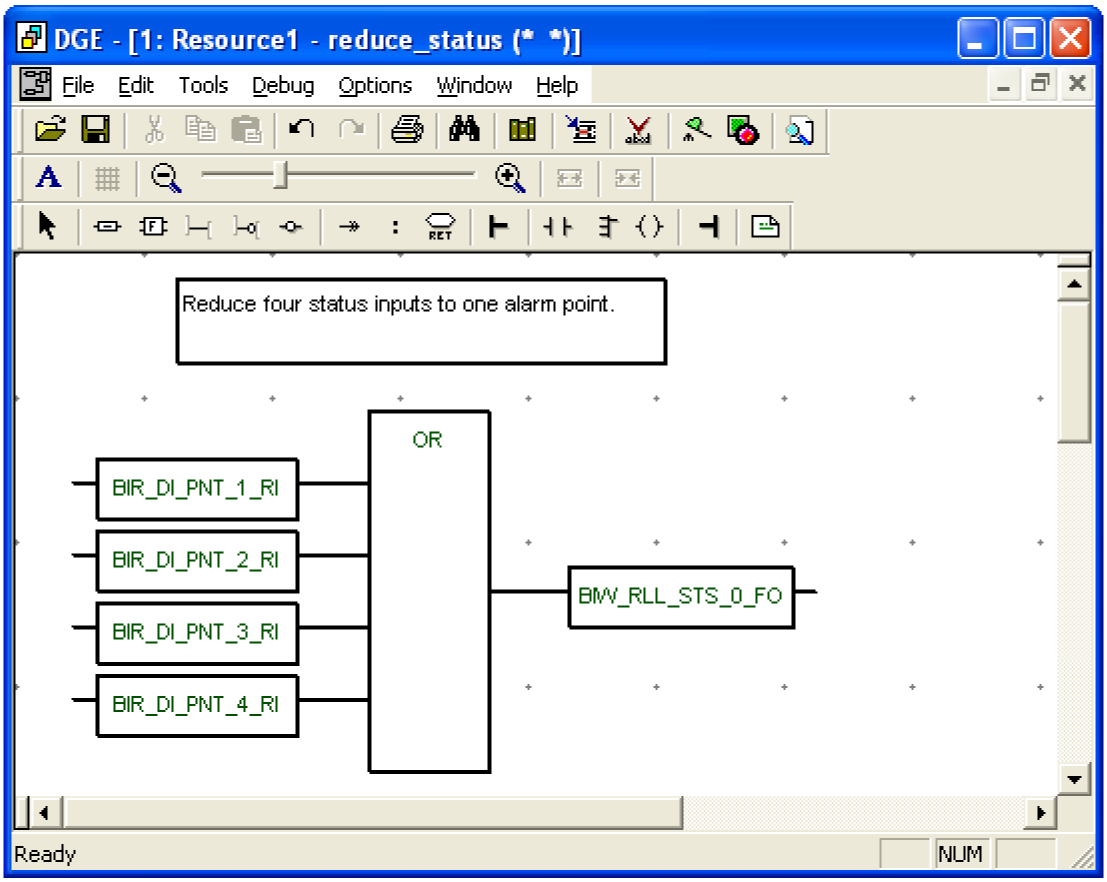


4.6.1 Program

This example shows how to OR four hardware status points to one pseudo point. The pseudo point may be read by the MTU.

As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-42 Reduce_Status Program



4.6.2 Variables

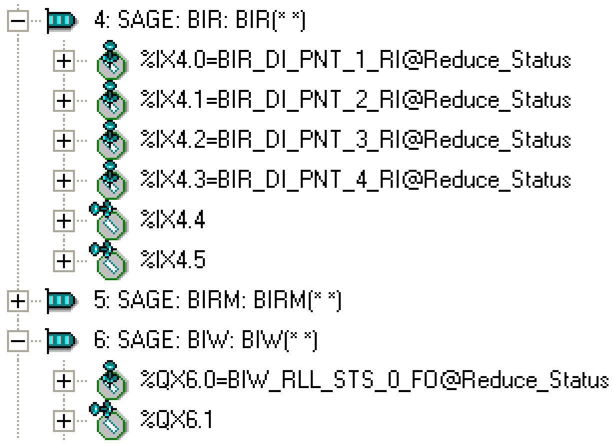
Figure 4-43 Reduce_Status Program Variables

reduce_status

Name	Alias	Type	()	Ini...	Di...	Group	Attribute	Scope	Direction	Retain	Wiring	Address
BIR_DI_PNT_1_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.0	
BIR_DI_PNT_2_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.1	
BIR_DI_PNT_3_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.2	
BIR_DI_PNT_4_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.3	
BIW_RLL_STS_0_FO		BOOL				None	Free	reduce_status	Output	No	%QX6.2	

4.6.3 Wiring

Figure 4-44 Reduce_Status Program Wiring



4.6.4 RTU Mapping

Figure 4-45 Reduce_Status Program RTU RLL Point Mapping

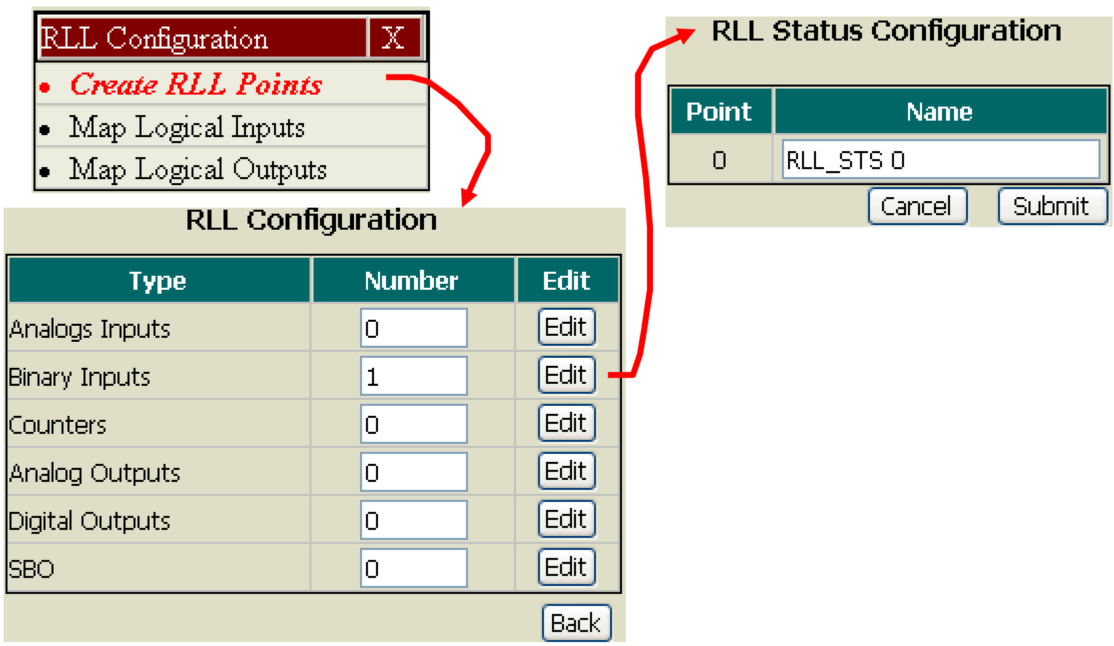


Figure 4-46 Reduce_Status Program RTU Input Logic Point Mapping

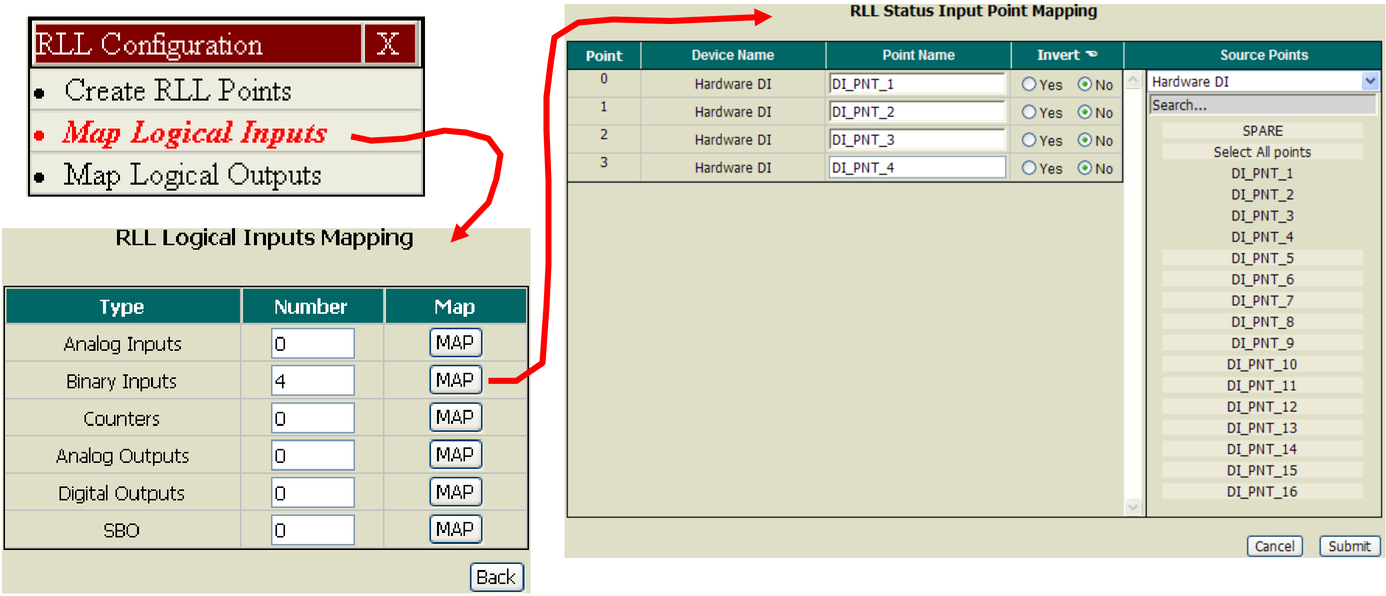
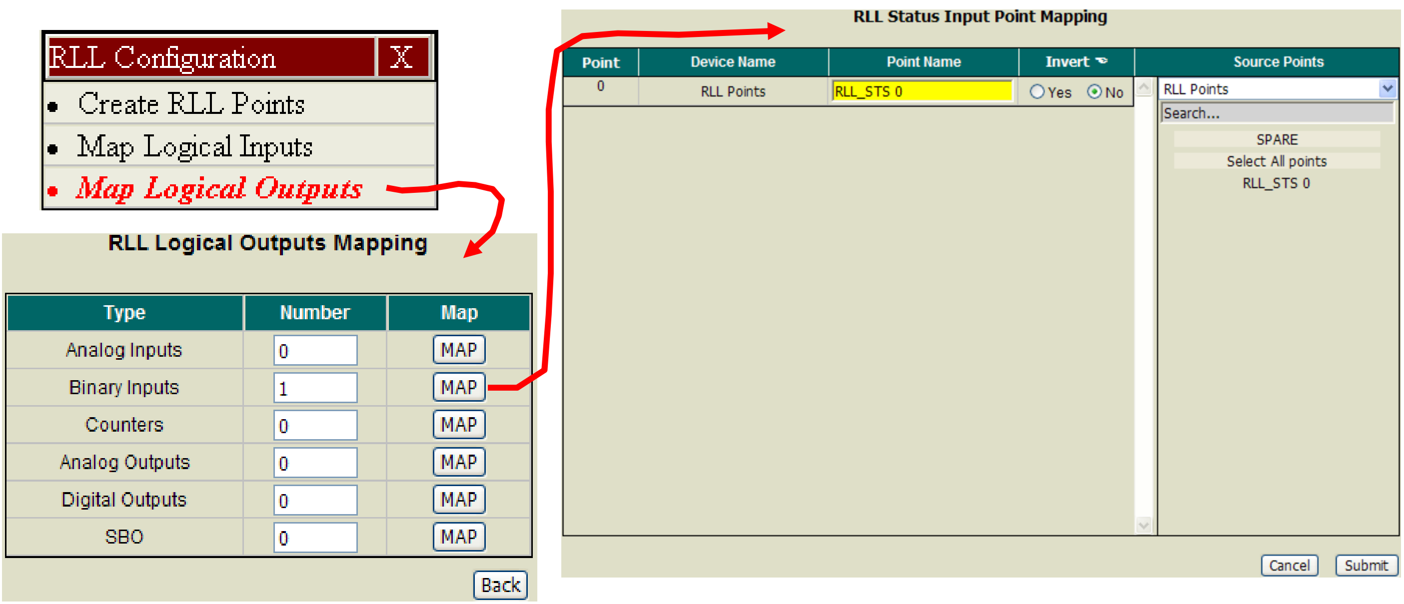


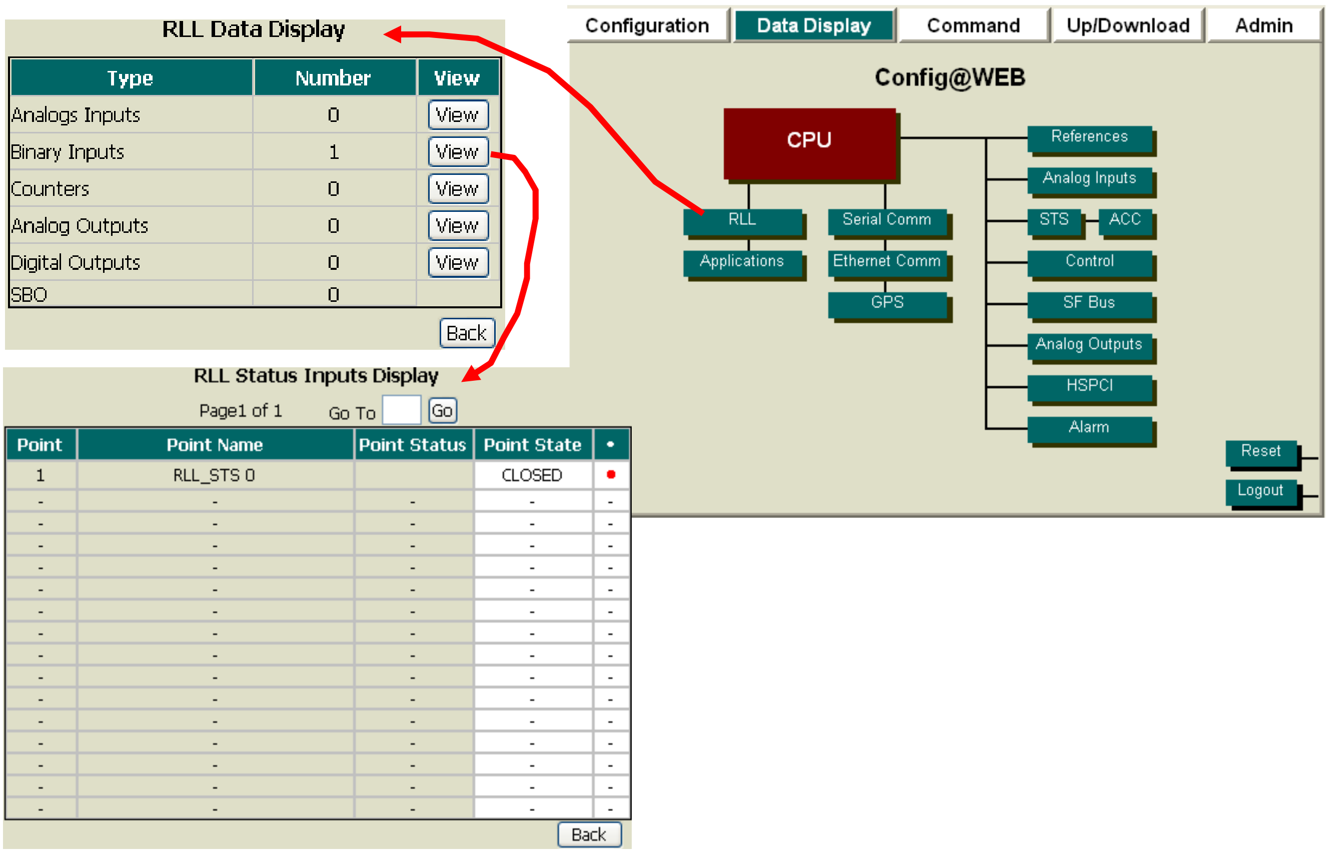
Figure 4-47 Reduce_Status Program RTU Output Logic Point Mapping



This concludes the configuration of the Reduce_Status Program. To see the program in operation you must exercise any of the first four hardware status, then observe the RLL status display.

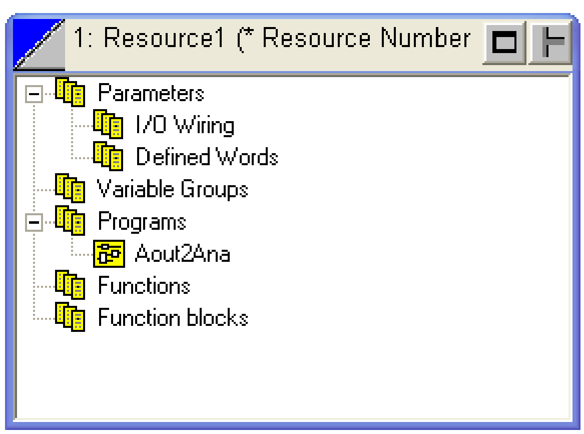
4.6.5 RTU Display

Figure 4-48 Resulting RLL Status Display



4.7 Copying AO to AI & AI to AO

Figure 4-49 Link Architecture View



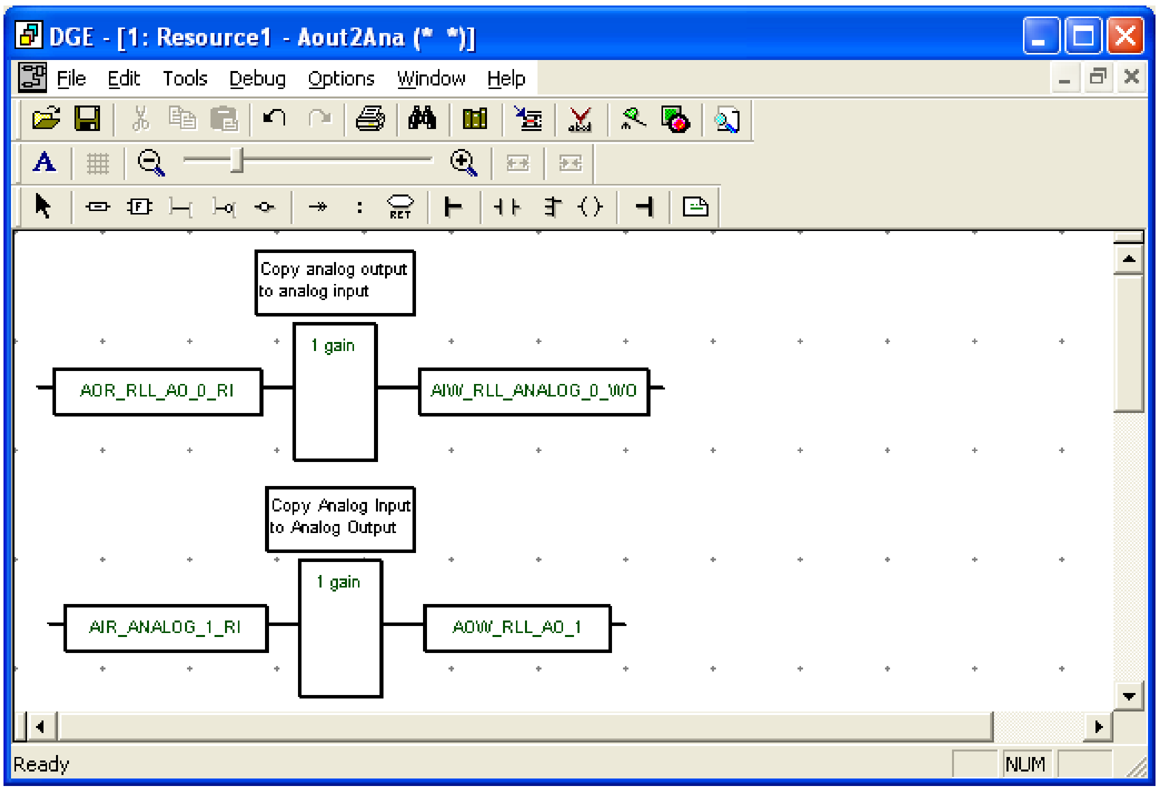
4.7.1 Program

This example shows how to copy AOs to AIs and AIs to AOs. The pseudo point AIW_RLL_ANALOG_0_WO may be read by the MTU.

As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

This example shows how various types of analogs may be converted into other types. Notice that the variable RLL_AO_1_Finternal is internal only. That is, it is not wired to a driver.

Figure 4-50 Copying AIs to AOs & AOs to AIs Program



4.7.2 Variables

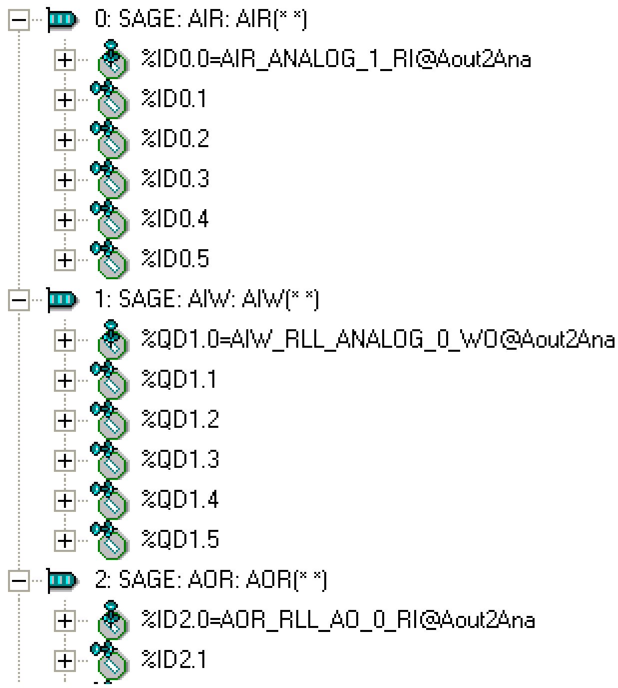
Figure 4-51 Copying AIs to AOs & AOs to AIs Program Variables

Aout2Ana

Name	Alias	Type	[]	Init. v...	Dimen...	Group	Attribute	Scope	Direction	Retain	Wiring	Address
AOR_RLL_AO_0_RI		DINT				None	Read	Aout2Ana	Input	No	%ID2.0	
AIW_RLL_ANALOG_0_WO		DINT				None	Write	Aout2Ana	Output	No	%QD1.0	
AIR_ANALOG_1_RI		DINT				None	Read	Aout2Ana	Input	No	%ID0.0	
AOW_RLL_AO_1		DINT				None	Write	Aout2Ana	Output	No	%QD3.0	

4.7.3 Wiring

Figure 4-52 Aout2Ana Program Wiring



4.7.4 RTU Mapping

Figure 4-53 Aout2Ana Program RTU RLL Point Mapping

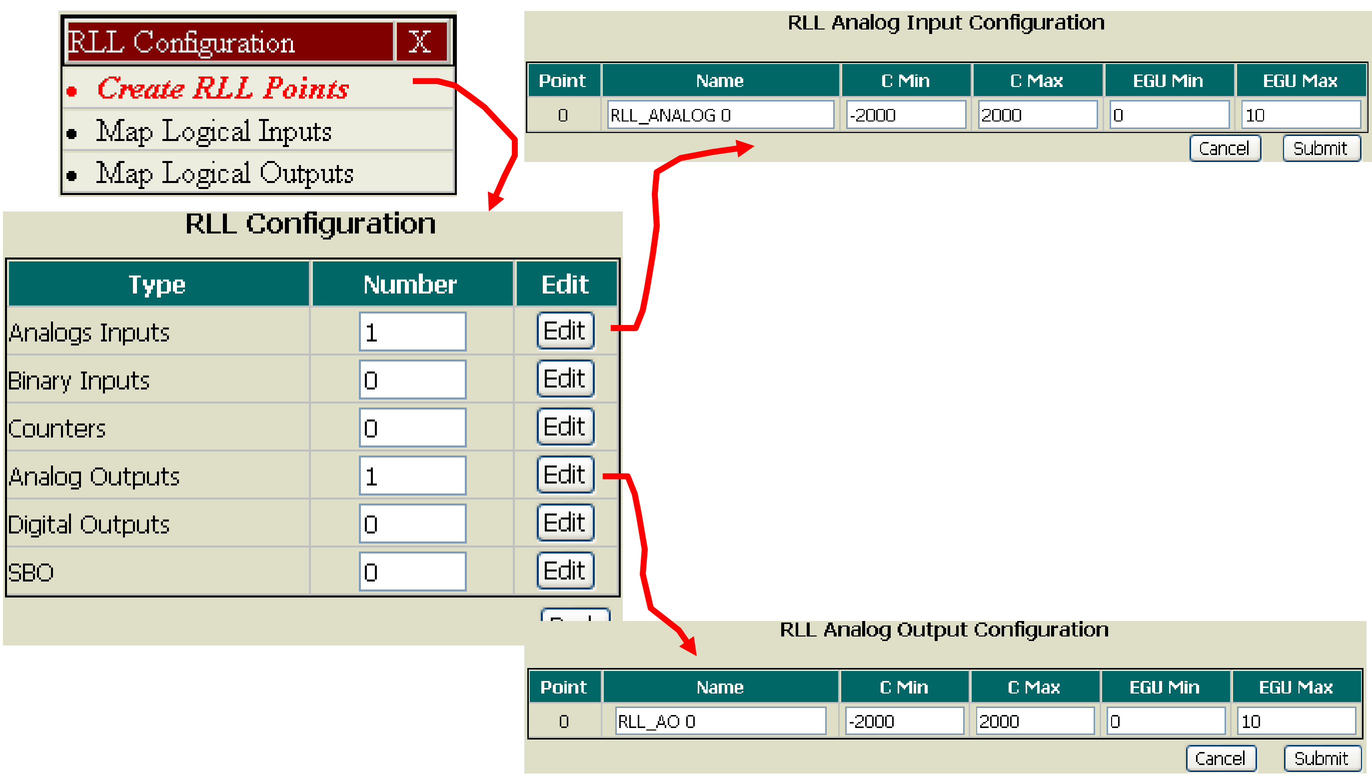
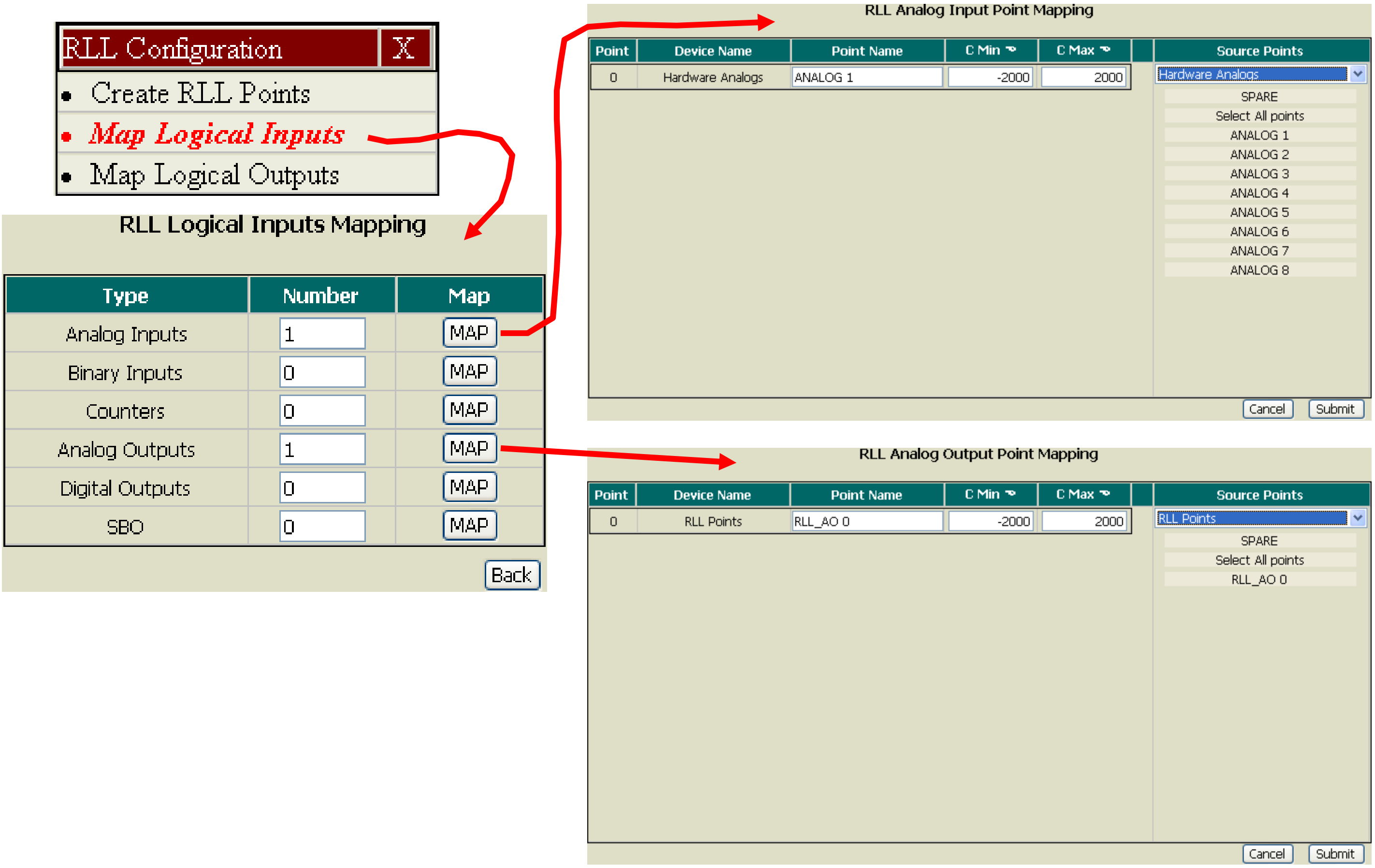


Figure 4-54 Aout2Ana Program RTU Input Logic Point Mapping

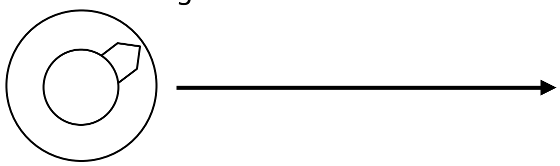


This concludes the configuration of the Aout2Ana Program. To see the program in operation you must exercise the single hardware analog, observe the results, then exercise the RLL analog and observe the result.

4.7.5 RTU Display

Figure 4-55 Exercising Hardware AI with RLL Analog Display

Hardware Analog



Caution: Because we have turned an "Other I/O" type of point (AOW) into an RLL point, the results may be unpredictable. That is, while we are trying to drive AOW with AIR, the RTU may be trying to control it also because AOW is a hardware output point.

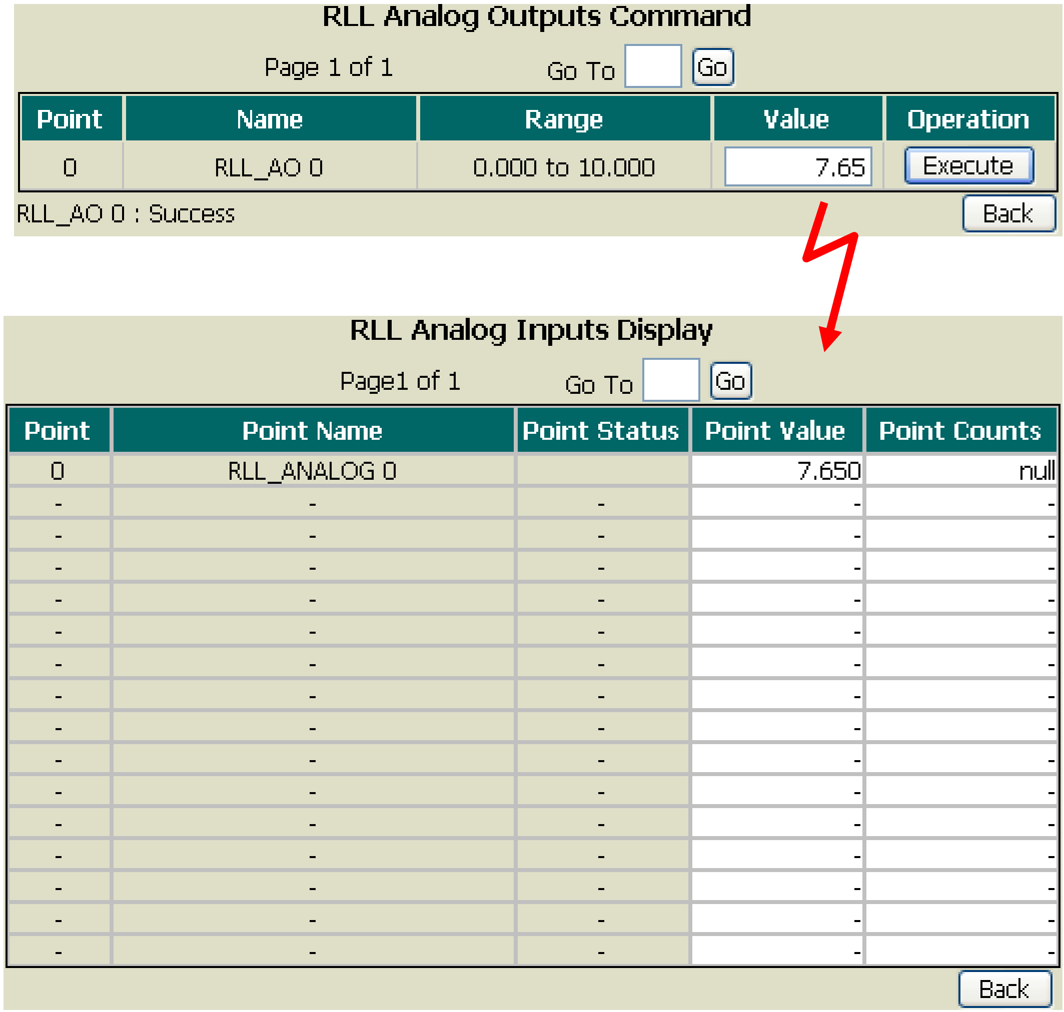
RLL Analog Outputs Display

Page1 of 1 Go To [ ] Go

Point	Point Name	Point Status	Point Value
0	RLL_AO 0		4.342
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-
-	-	-	-

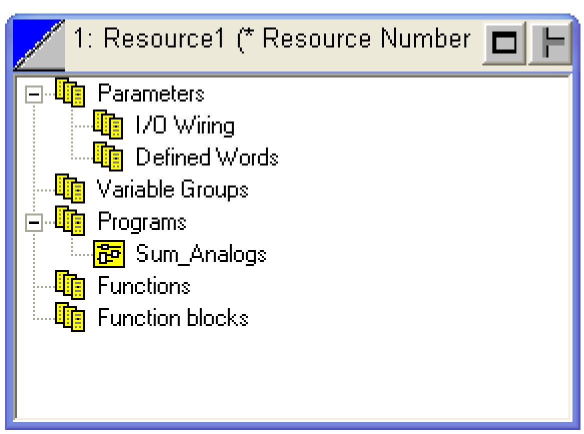
Back

Figure 4-56 Exercising AOR to AIW



4.8 Summing Analog Points

Figure 4-57 Link Architecture View



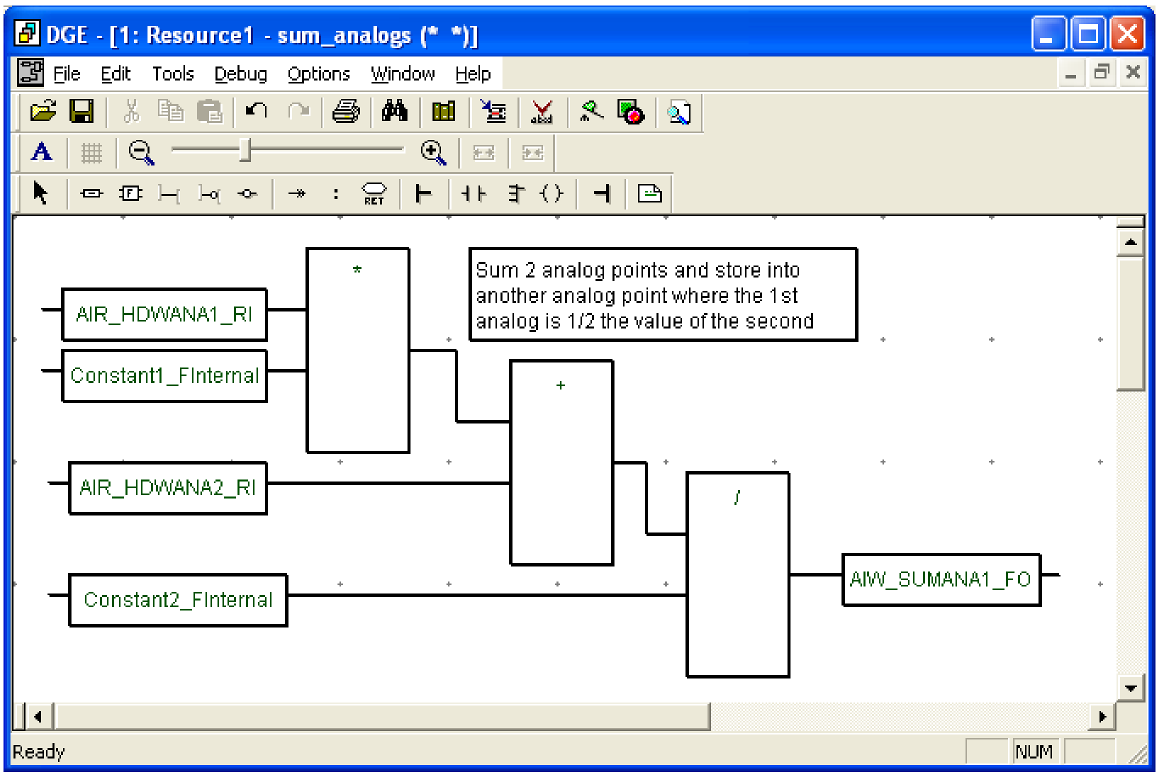
4.8.1 Program

This example shows how to sum two analog points and store into another analog point where the 1st analog is 1/2 the value of the second analog.

As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

The constants, Constant1_Finternal and Constant2_Finternal, have been assigned a value of 2, as you can see in the variables.

Figure 4-58 Summing Analog Points Program



4.8.2 Variables

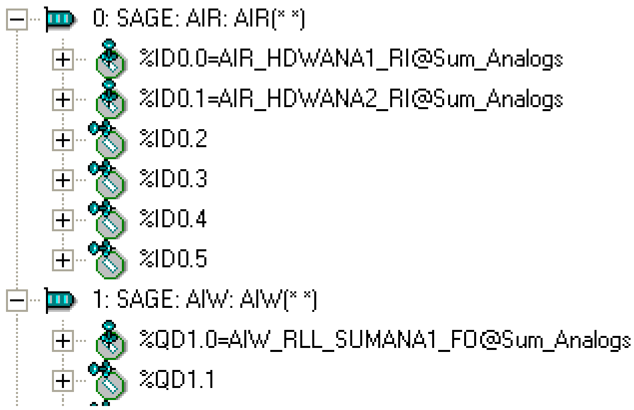
Figure 4-59 Summing Analog Points Program Variables

Sum_Analogs

Name	Alias	Type	()	Init. value	Dimen...	Group	Attribute	Scope	Direction	Retain	Wiring	Address	C
AIR_HDWANA1_RI		DINT				None	Read	Sum_Analogs	Input	No	%ID0.0		
AIR_HDWANA2_RI		DINT				None	Read	Sum_Analogs	Input	No	%ID0.1		
AIW_RLL_SUMANA1_FO		DINT				None	Free	Sum_Analogs	Output	No	%QD1.0		
Constant1_FInternal		DINT		2		None	Free	Sum_Analogs	Internal	No			
Constant2_FInternal		DINT		2		None	Free	Sum_Analogs	Internal	No			

4.8.3 Wiring

Figure 4-60 Summing Analog Points Program Wiring



4.8.4 RTU Mapping

Figure 4-61 Summing Analog Points Program RTU RLL Point Mapping

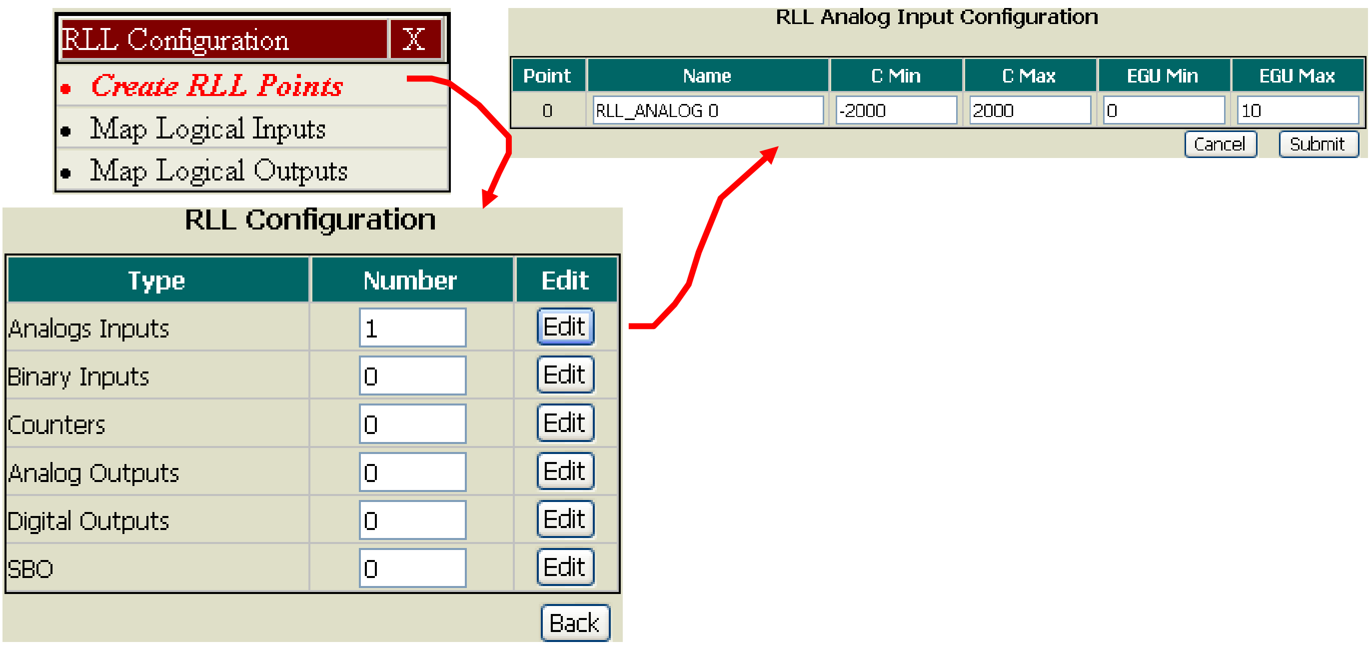
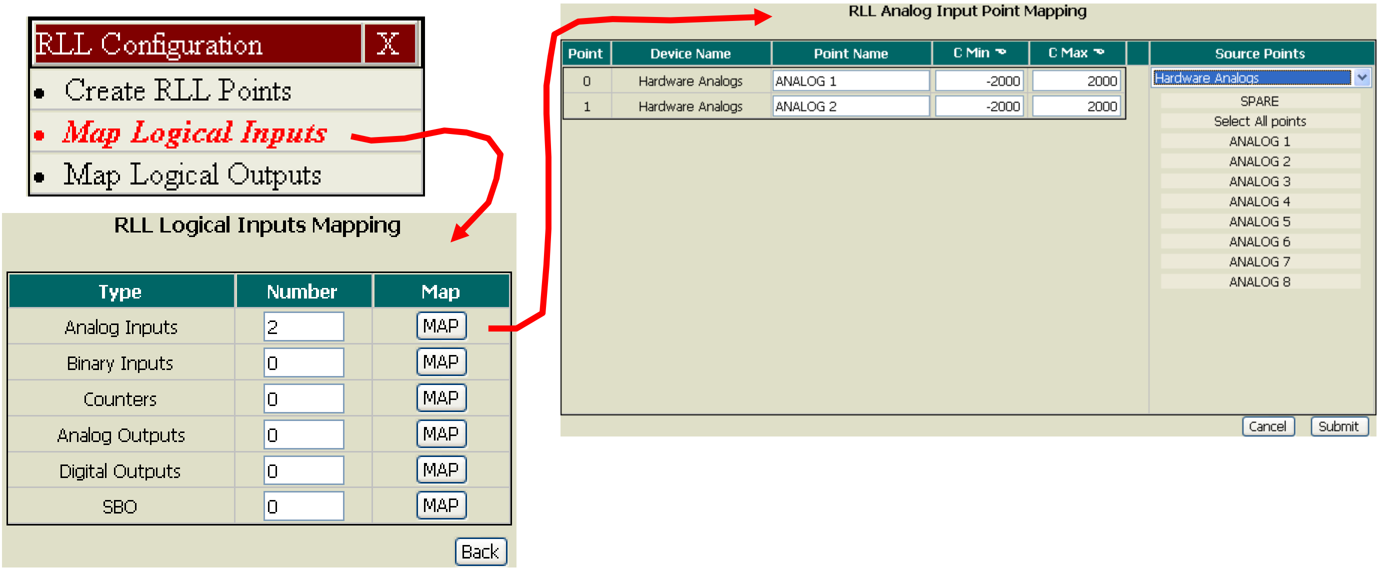


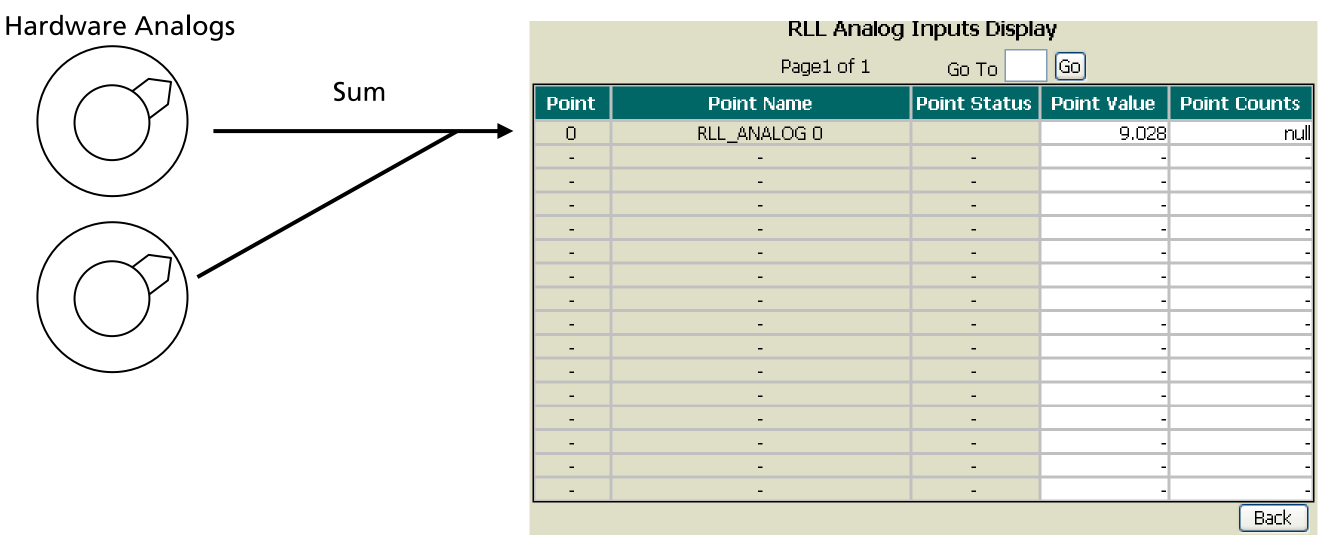
Figure 4-62 Summing Analog Points Program Hardware AI Point Mapping



4.8.5 RTU Display

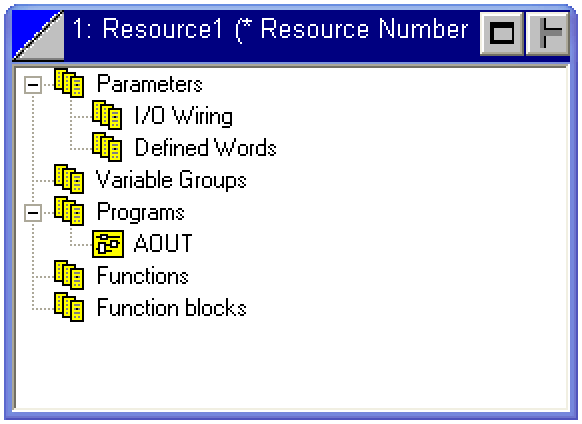
To see the program in operation you must exercise the hardware analogs, then observe the RLL analog output.

Figure 4-63 Resulting RLL Analog Display



4.9 Copying AOR to AOW

Figure 4-64 Link Architecture View

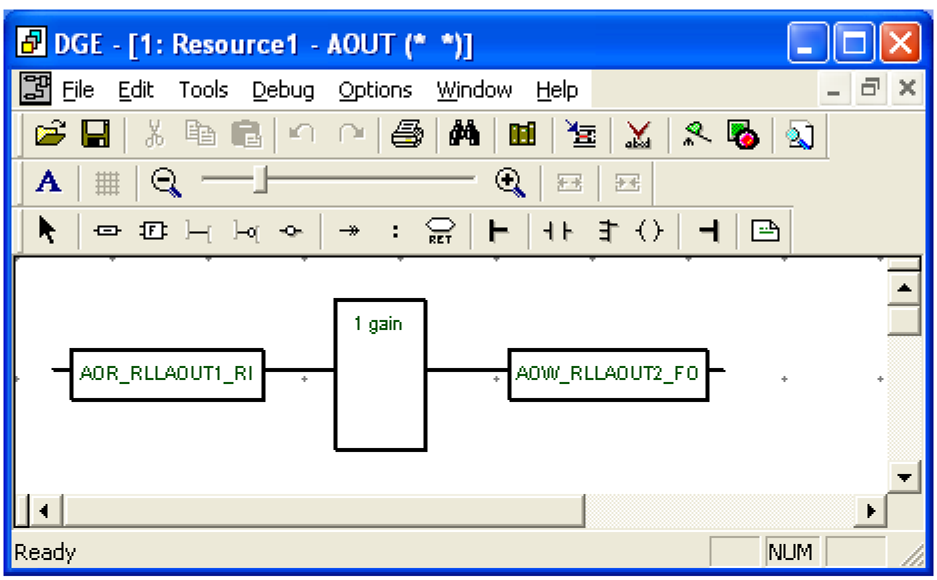


4.9.1 Program

This example shows how to copy an Analog Output Read (AOR) point to an Analog Output Write (AOW) point, without any changes.

As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-65 AOUT Program



4.9.2 Variables

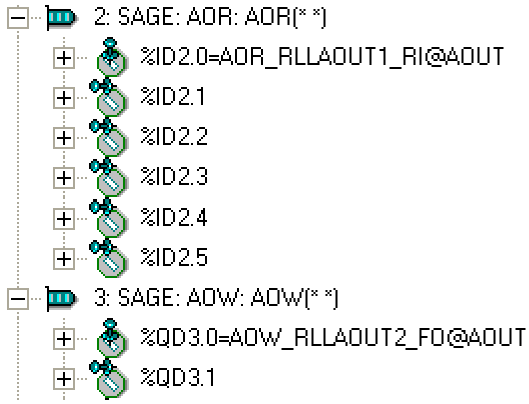
Figure 4-66 AOUT Program Variables

AOUT

Name	Alias	Type	()	Init. value	Dimensi...	Group	Attribute	Scope	Direction	Retain	Wiring	Address	C
AOR_RLLAOUT1_RI		DINT				None	Read	AOUT	Input	No	%ID2.0		
AOW_RLLAOUT2_FO		DINT				None	Write	AOUT	Output	No	%QD3.0		

4.9.3 Wiring

Figure 4-67 AOUT Program Wiring



4.9.4 RTU Mapping

Figure 4-68 AOUT Program RLL RTU Mapping

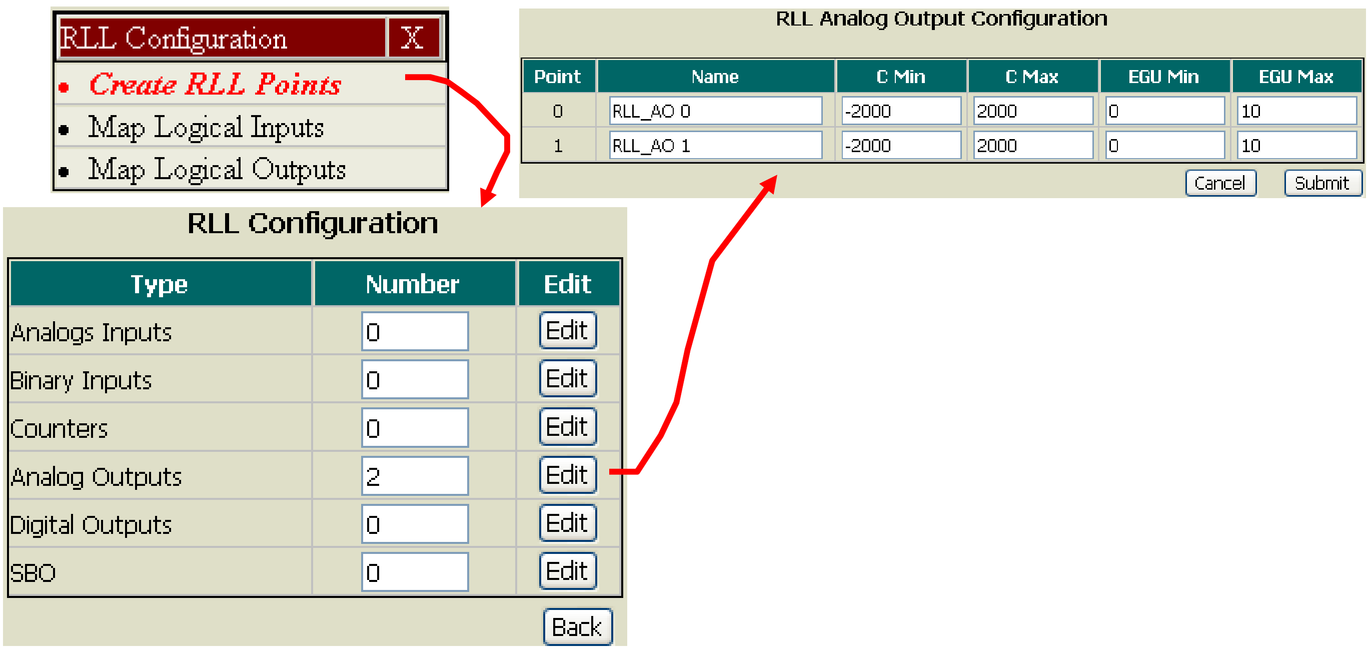


Figure 4-69 AOUT Program Logical Input RTU Mapping

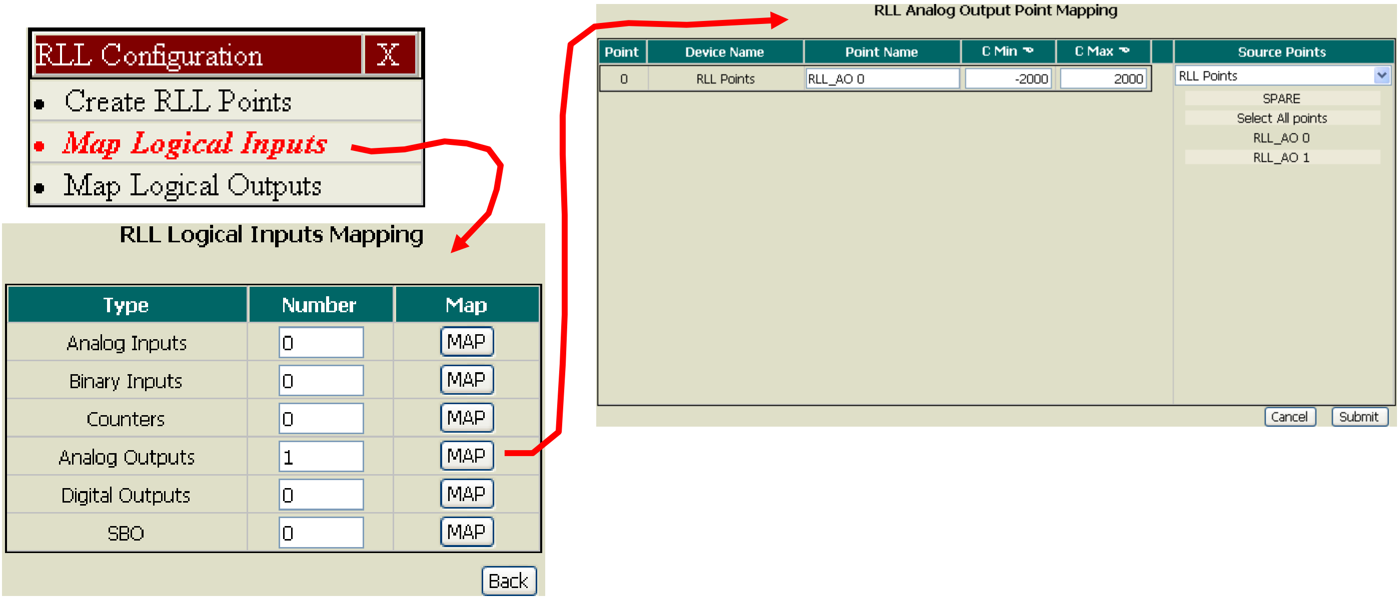
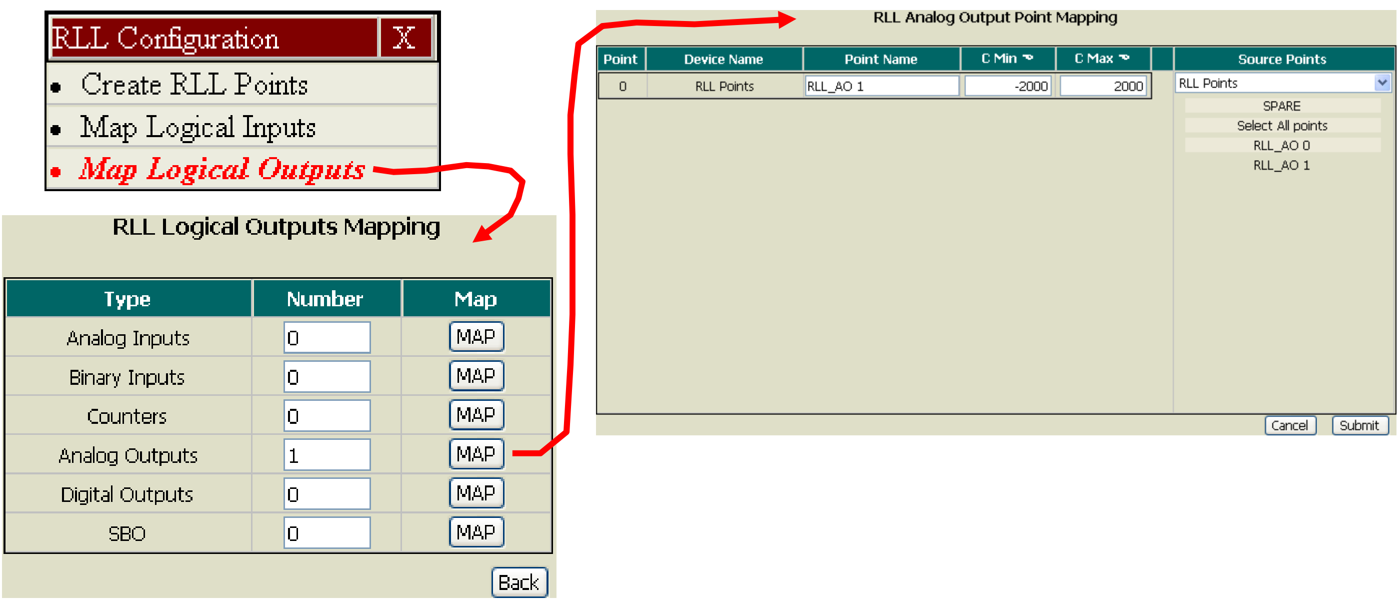


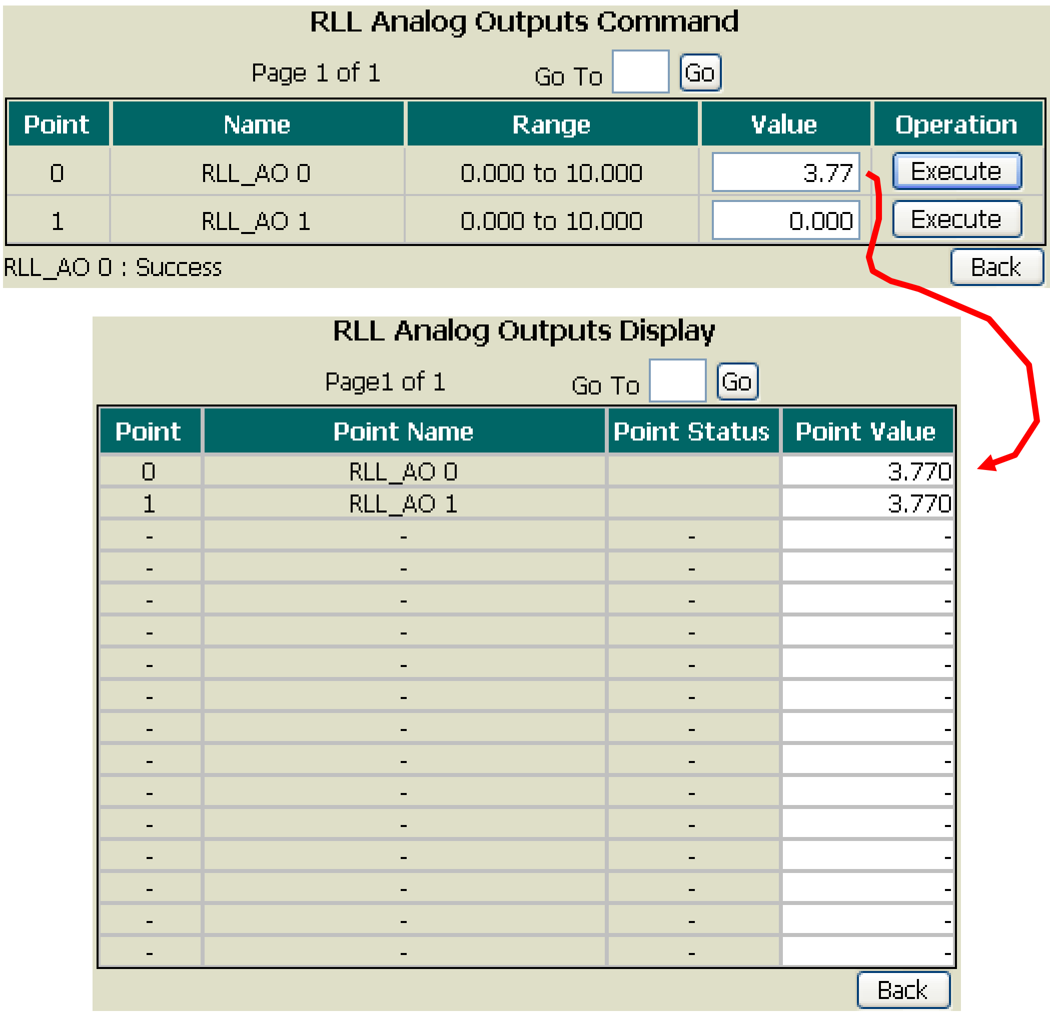
Figure 4-70 AOUT Program Logical Output RTU Mapping



4.9.5 RTU Display

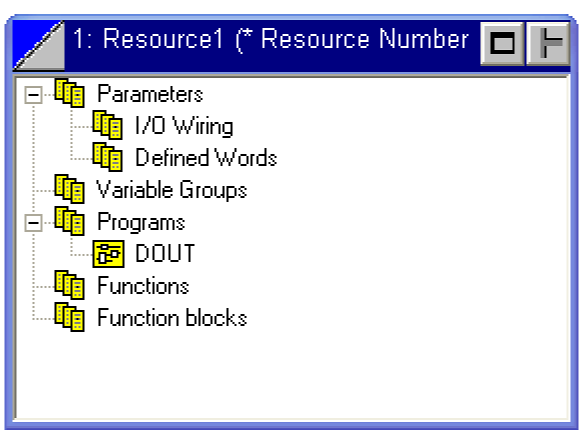
When the first RLL point is commanded (AOR), the Display shows both RLL points (AOR & AOW) changing to the commanded value.

Figure 4-71 AOUT Program RTU Display



4.10 Copying BOR to BOW

Figure 4-72 Link Architecture View

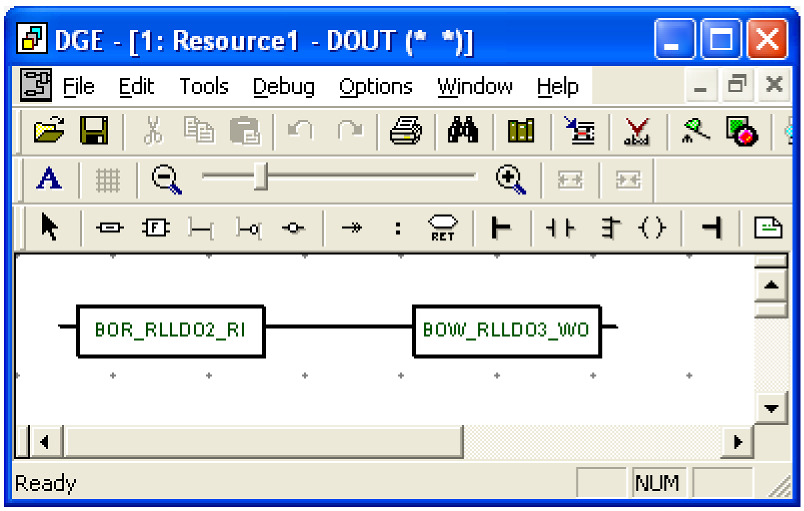


4.10.1 Program

This example shows how to copy a Binary Output Read (BOR) point to a Binary Output Write (BOW) point, without any changes.

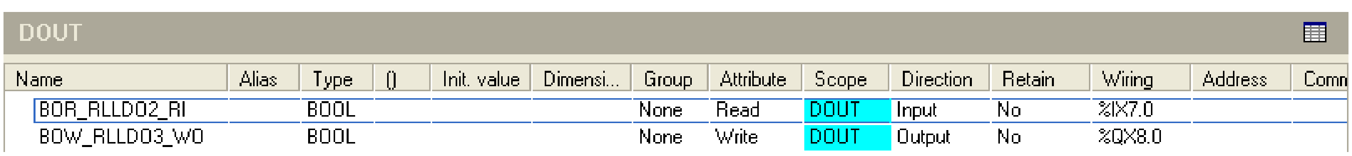
As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-73 DOUT Program



4.10.2 Variables

Figure 4-74 DOUT Program Variables

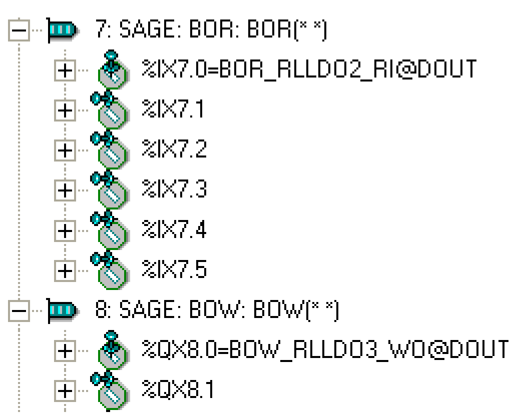


DOUT

Name	Alias	Type	()	Init. value	Dimensi...	Group	Attribute	Scope	Direction	Retain	Wiring	Address	Comn
BOR_RLLD02_RI		BOOL				None	Read	DOUT	Input	No	%IX7.0		
BOW_RLLD03_WO		BOOL				None	Write	DOUT	Output	No	%QX8.0		

4.10.3 Wiring

Figure 4-75 DOUT Program Wiring



4.10.4 RTU Mapping

Figure 4-76 DOUT Program RLL RTU Mapping

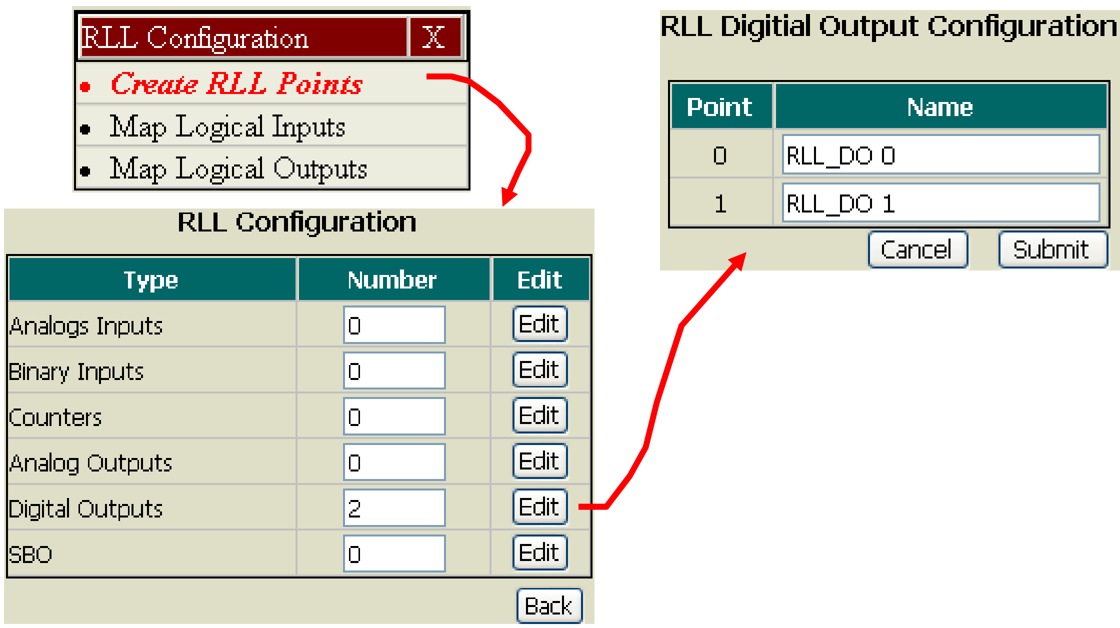


Figure 4-77 DOUT Program Logical Input RTU Mapping

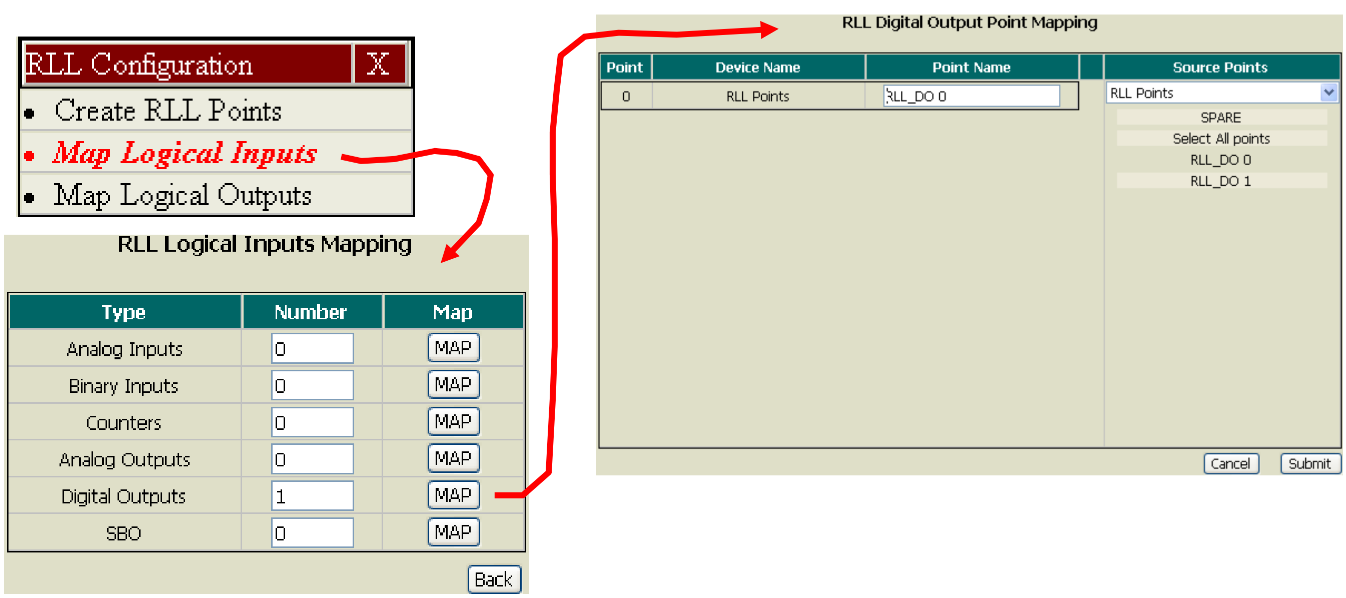
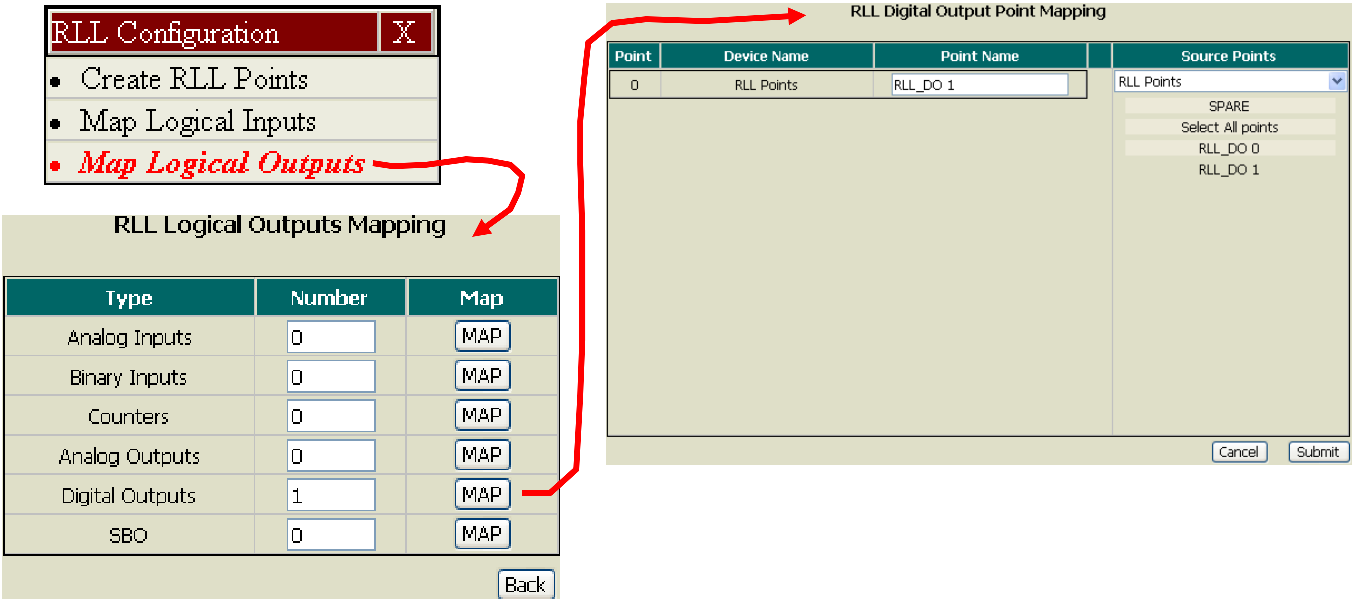


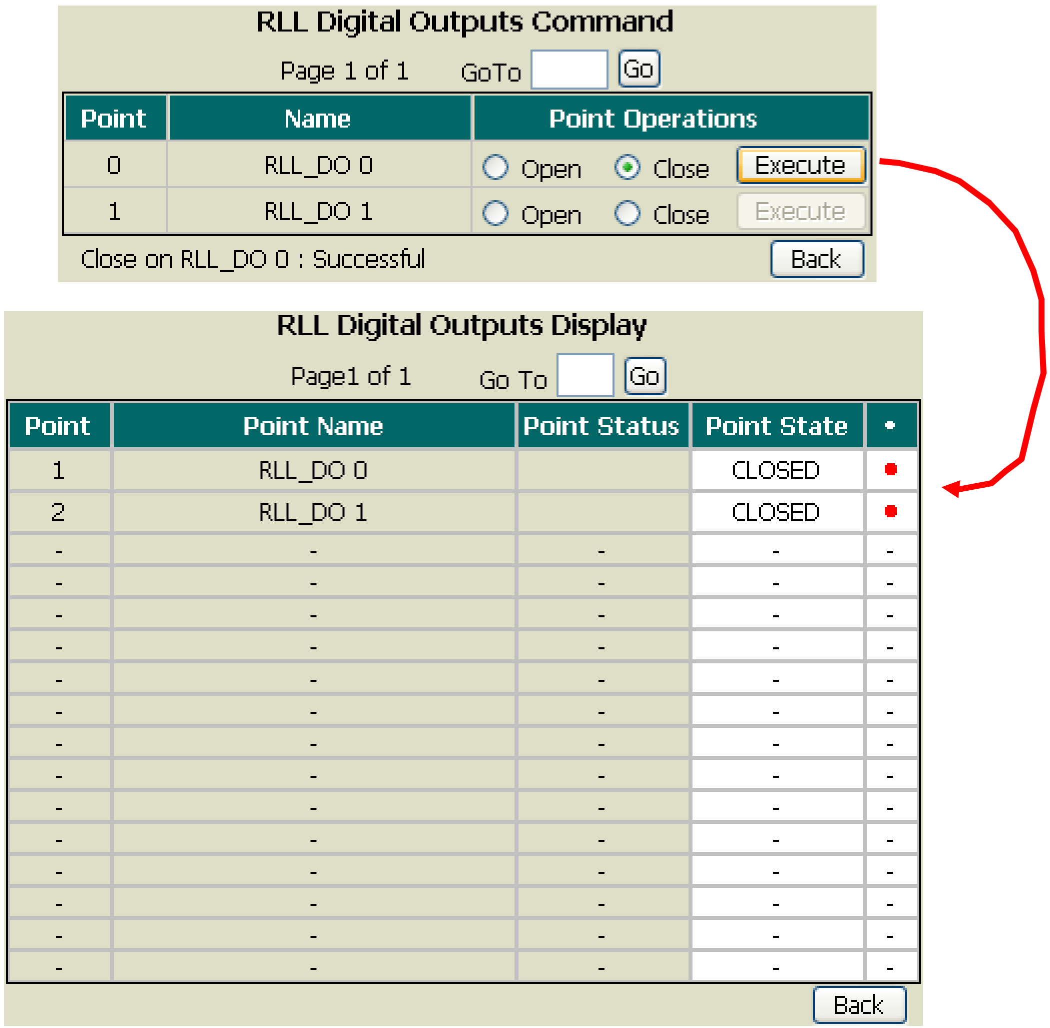
Figure 4-78 DOUT Program Logical Output RTU Mapping



4.10.5 RTU Display

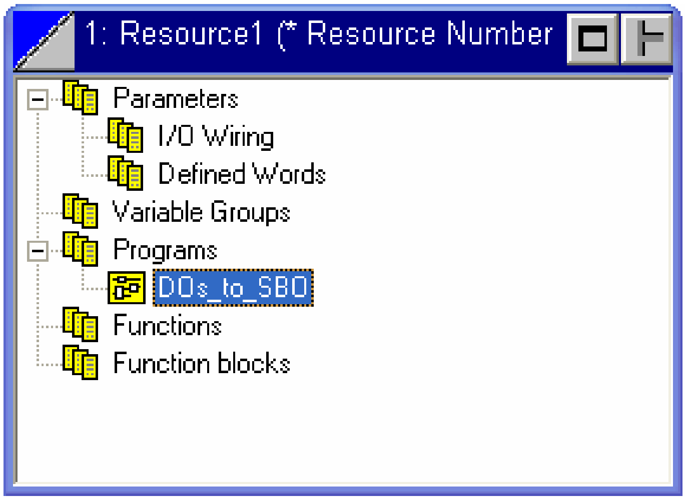
When the first RLL point is commanded (BOR), the Display shows both RLL points (BOR & BOW) changing to the commanded value.

Figure 4-79 DOUT Program RTU Display



4.11 Converting 2 DOs from Master to 1 SBO Trip/Close to IED

Figure 4-80 Link Architecture View

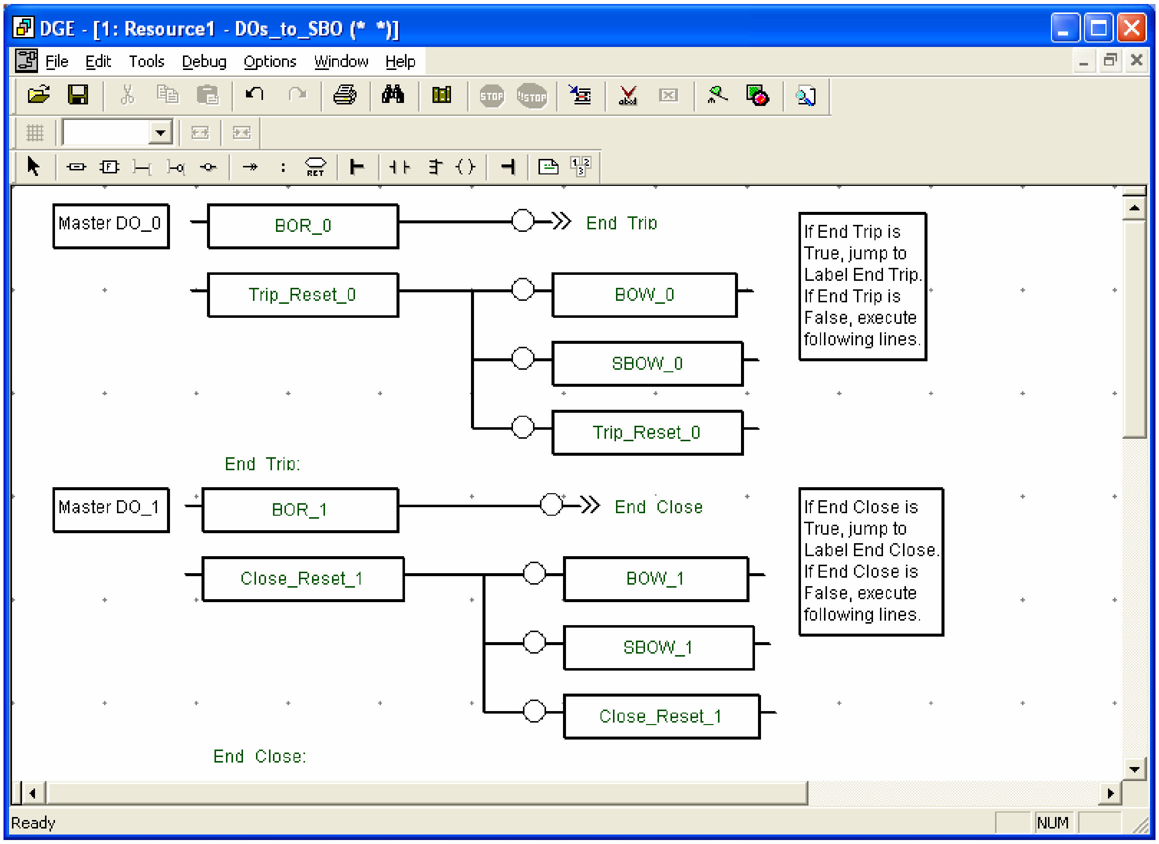


4.11.1 Program

This example shows how to copy a Binary Output Read (BOR) point to a Binary Output Write (BOW) point, without any changes.

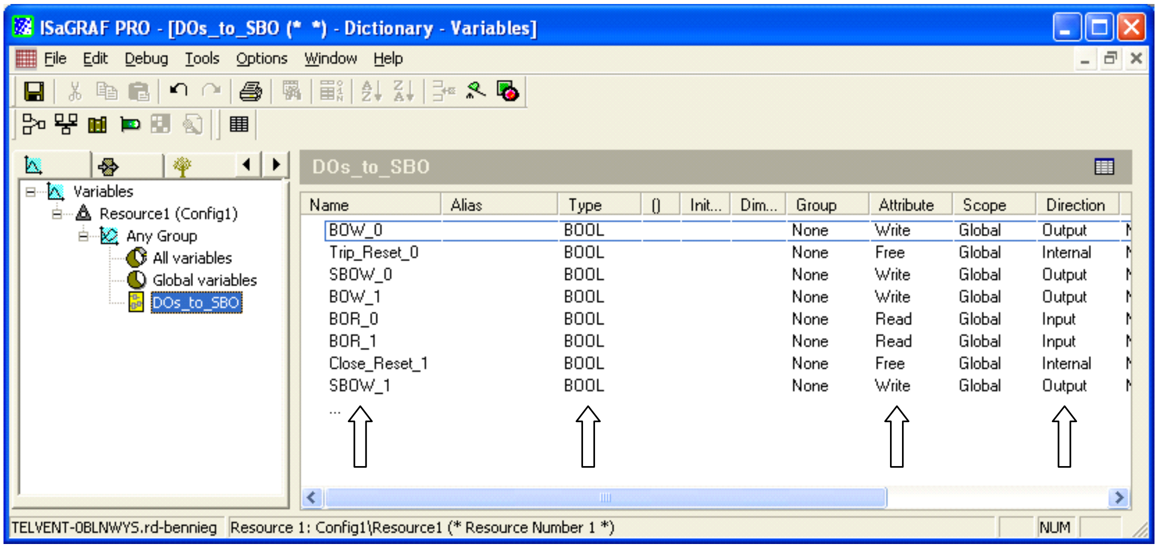
As a programming aide, the names of variables have incorporated the Wiring codes for the particular mapping required. See Figure 4-1 for the correlation. This technique, although by no means mandatory, helps the user follow the mapping thread from program creation to point mapping on the RTU.

Figure 4-81 DOs to SBO Program



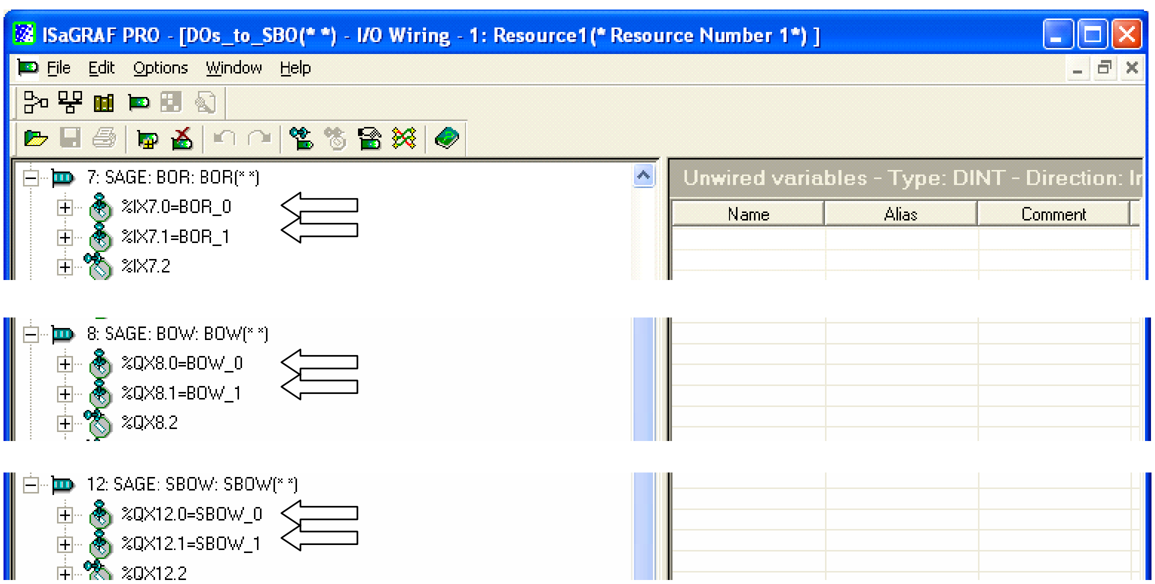
4.11.2 Variables

Figure 4-82 DOs to SBO Variables



4.11.3 Wiring

Figure 4-83 DOs to SBO Wiring



4.11.4 RTU Mapping

Figure 4-84 DOs to SBO Create RLL Point

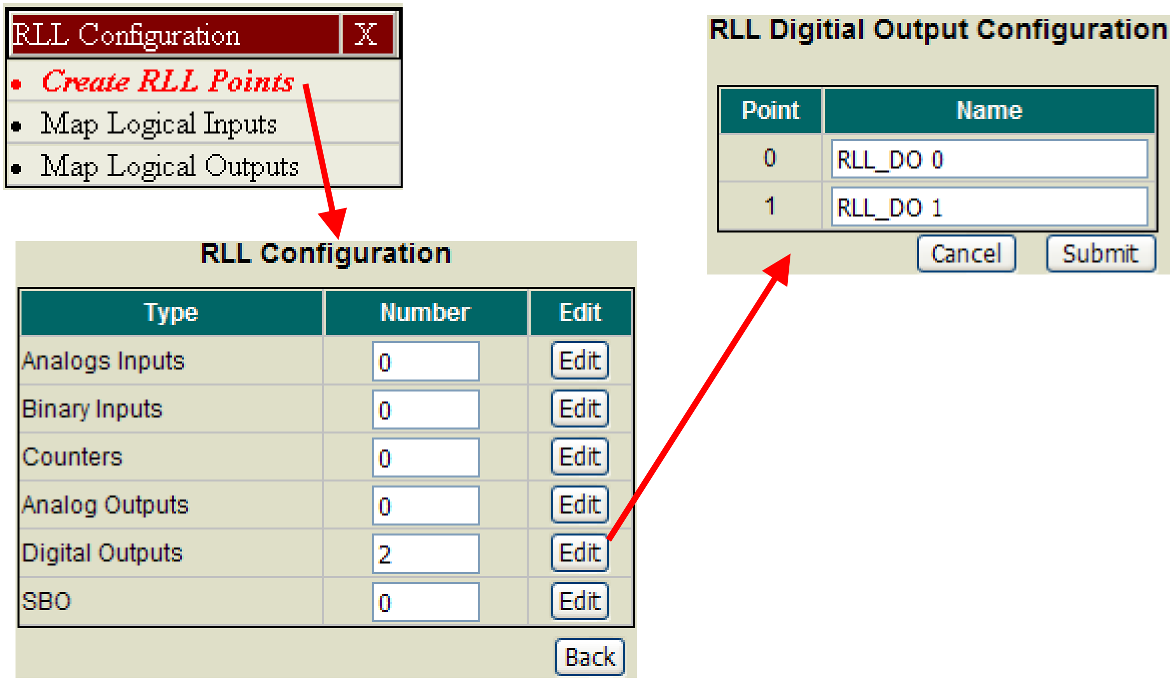


Figure 4-85 DOs to SBO Logical Input RTU Mapping

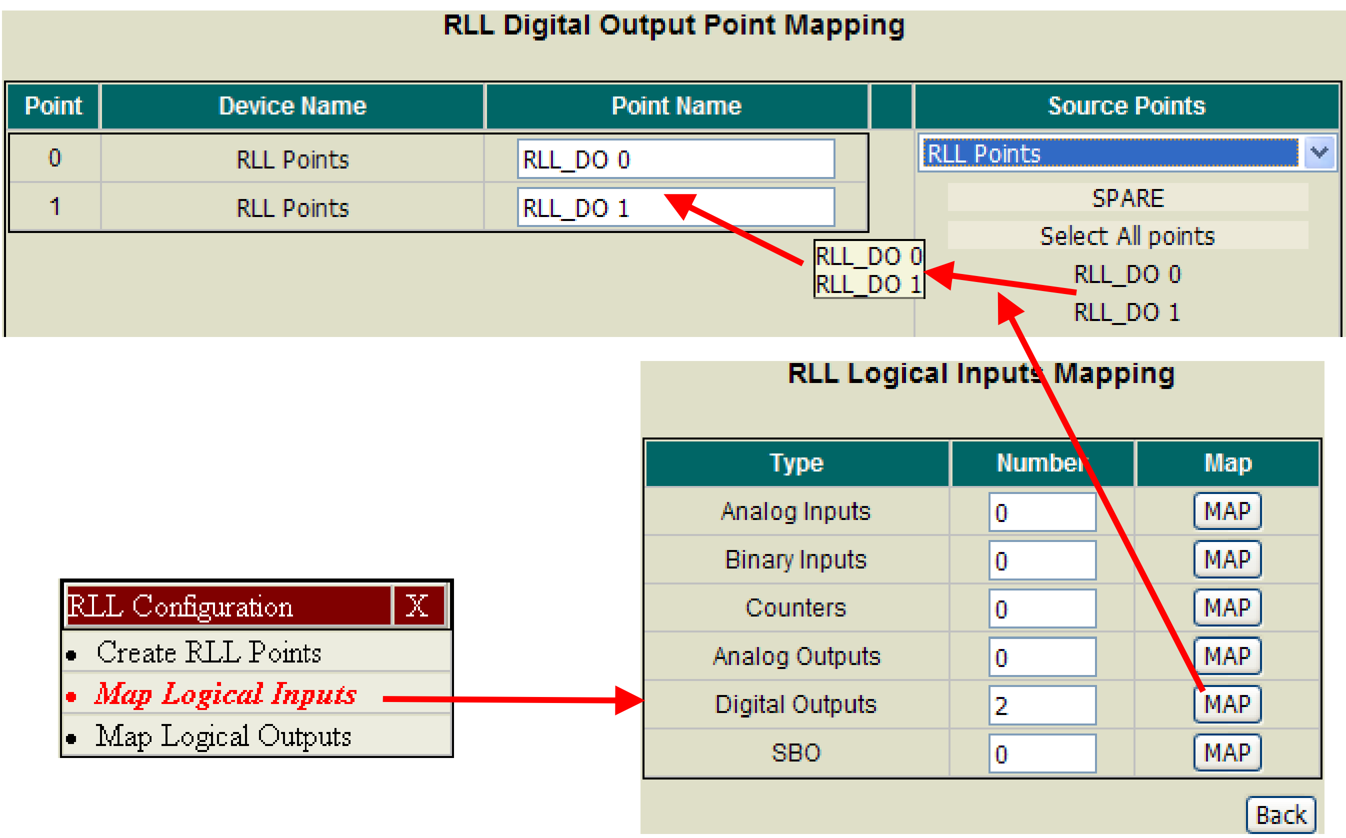
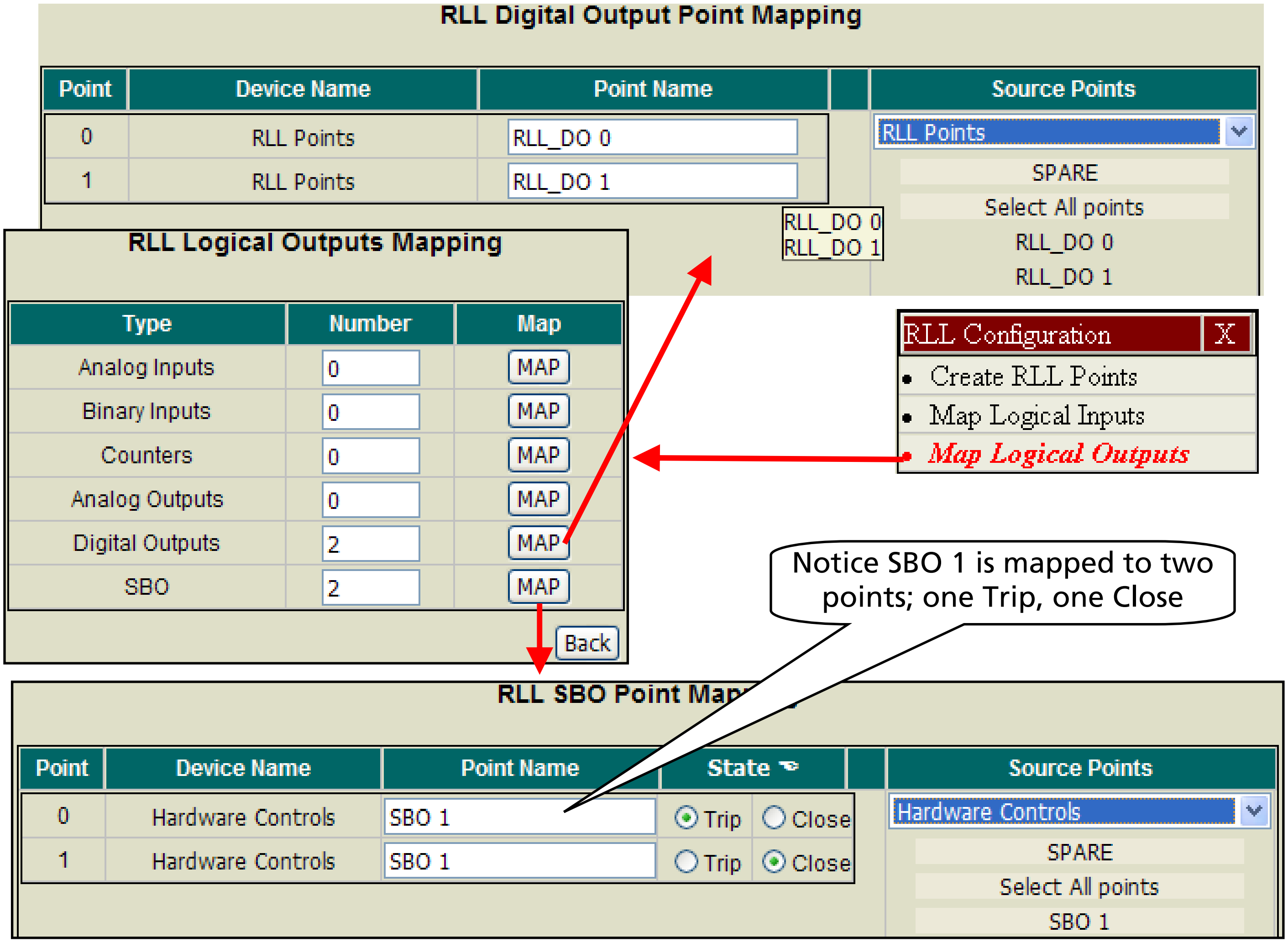
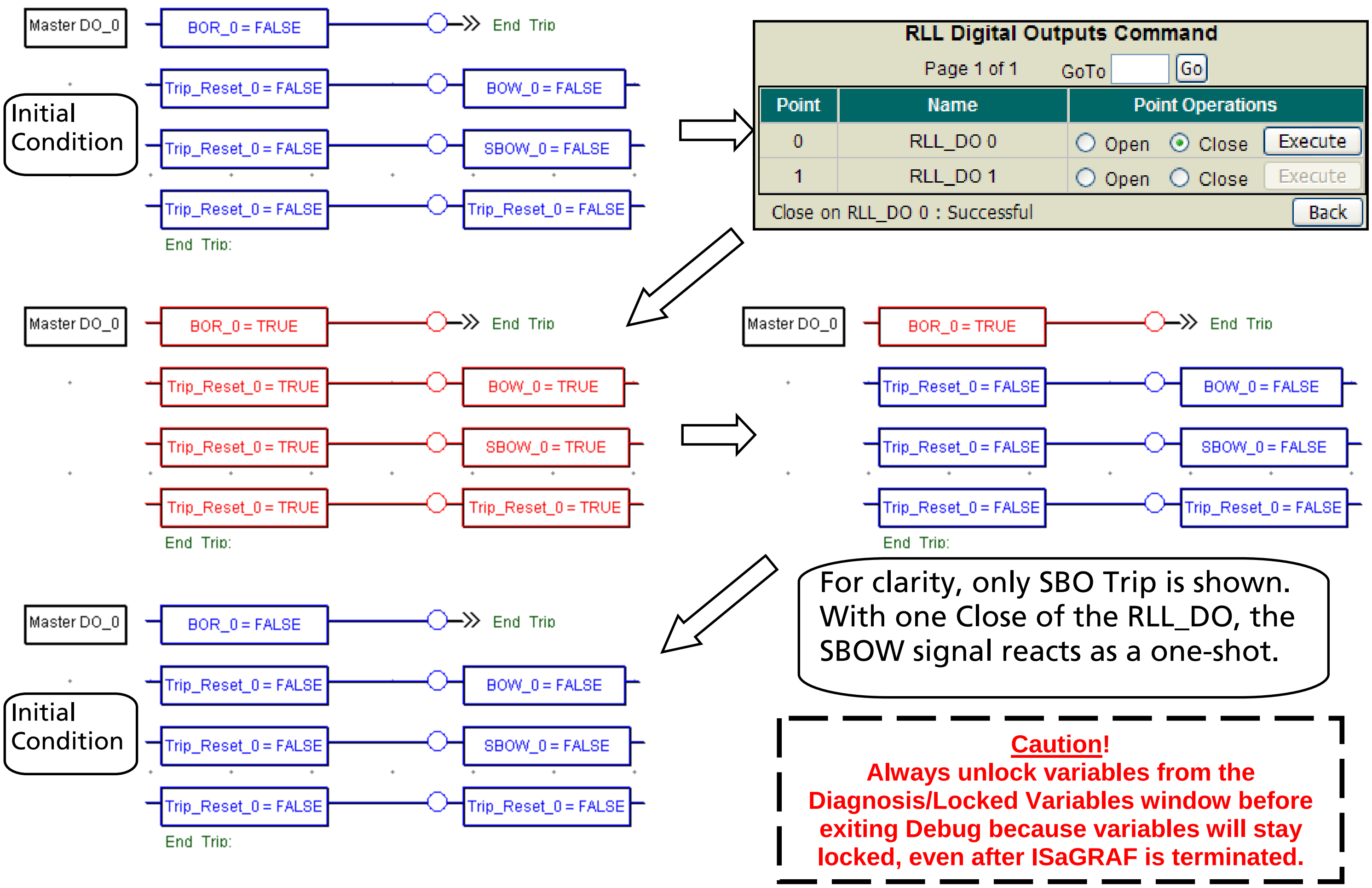


Figure 4-86 DOs to SBO Logical Output RTU Mapping



4.11.5 Testing With Debugging

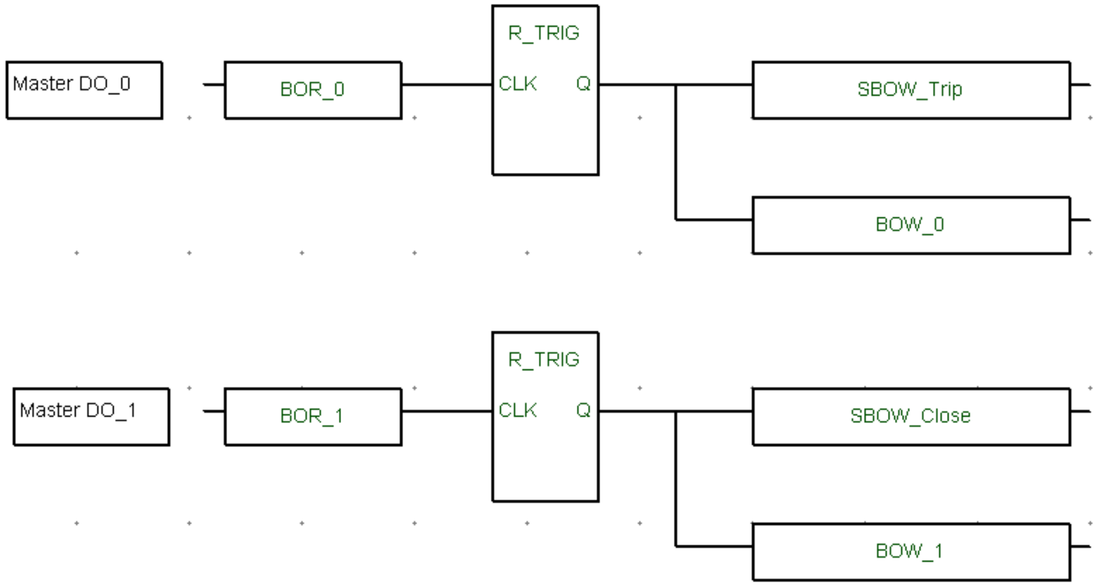
Figure 4-87 Testing With Debugging



4.11.6 Program Variation

This is another way of doing the same thing. Configuration mapping would be exactly the same. The big difference is that the program is simpler.

Figure 4-88 DOs to SBO Program Variation



4.11.7 RTU Display

There is no display for SBOs.

APPENDIX A

Glossary

A/D	Analog to Digital
AC	Alternating Current
ACI	AC Input
ADC	Analog to Digital Converter
AI	Analog Input, also AIN
ANSI	American National Standards Institute
AO	Analog Output, also AOUT
ASCI	Asynchronous Serial Communications Interface
ASCII	American Standard Code for Information Interchange
ASIC	Application Specific Integrated Circuit
AWG	American Wire Gauge
baud	Modem speed in Bits Per Second
bps	Bits Per Second
bridge	A network device capable of connecting networks that use similar protocols
C	Celsius or the programming language C
CEB	Communication Expansion Board
check-back	Hardware/Software method of control output protection
CCITT	Comité Consultatif Internationale de Télégraphique et Téléphonique
CMOS	Complementary Metal Oxide Semiconductor
COMM	Communication, also COM
COS	Change of State
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check; a method for error checking that detects randomly occurring single and multiple bit errors and is widely accepted for the detection of "burst" errors encountered in communication networks.
CTS	Clear To Send
DAC	Digital to Analog Converter
dBm	Decibels relative to 1mW
DC	Direct Current
debounce	Filtering of contact closure noise
DHCP	Dynamic Host Configuration Protocol – often used to refer to the network server that performs this function
DI	Digital Input
DFT	Discreet Fourier Transform
DMA	Direct Memory Access
DMM	Digital Multimeter
DNS	Domain Naming Service – often used to refer to the network server that performs this function
DO	Digital Output

DSP	Digital Signal Processor
DTR	Data Terminal Ready
DVM	Digital Volt Meter
EIA	Electronic Industries Association
EEPROM	Electrically Erasable Programmable Read Only Memory
EPLD	Electrically Programmable Logic Device
EPROM	Erasable Programmable Read Only Memory
Ethernet	A broadcast networking technology that can use several different physical media, including twisted pair cable and coaxial cable. TCP/IP is commonly used with Ethernet networks.
FB	Function Block – an element is the Function Block Diagram graphical language
FBD	Function Block Diagram graphical language – one of the IEC 61131-3 programming languages
FC	Flow Chart graphical language – one of the IEC 61131-3 programming languages
FF	Flip-Flop
FIFO	First In First Out
FIP	Fieldbus implementation based on French standard
firmware	Program held in ROM or Flash memory
Flash Memory	A type of non-volatile storage device similar to EEPROM
FMR	Feeder Management Remote
FMS	Feeder Management System
form A	Relay contact, single throw, normally open
form C	Relay contact, double throw
FRF	Full Range Factor; a method used for analog scaling; $\text{FRF} = \frac{\text{Data Value} - \text{Data Min}}{\text{Data Max} - \text{Data Min}}$
FS	Full Scale
FTP	File Transfer Protocol – A TCP/IP application used for transferring files from one system to another
GPS	Global Positioning System
GUI	Graphical User Interface
H	Hexadecimal (base 16), as in XXXXh
HEX	Hexadecimal (base 16), as in XXXXh
HDLC	High-level Data Link Control
HSPCI	High Speed Pulse Counter Input
Hz	Hertz, frequency in cycles per second
I/O	Input/Output
ID	Identification
IEC	International Electro-technical Commission
IED	Intelligent Electronic Device
IEEE	Institute of Electrical and Electronic Engineers
IL	Instruction List language – one of the IEC 61131-3 programming languages
ISA	Instrument Society of America
ISO	International Standards Organization
ISP	Integrated Software Project – Fieldbus implementation using existing IEC standards
ITU	Intelligent Terminal Unit
JEDEC	Joint Electronic Device Engineering Council

k	Kilo - kB is kilobytes, kV is kilovolts, etc.
KHz	Kilo Hertz
LAN	Local Area Network
LCD	Liquid Crystal Display
LD	Ladder Diagram graphical language – one of the IEC 61131-3 programming languages
LED	Light Emitting Diode
LRC	Longitudinal Redundancy Check; uses both "horizontal" and "vertical" parity bits to detect errors in the messages between the Master and the RTUs. This technique is also known as Geometric Coding.
LSB	Least Significant Bit
mA	Milliampere
MAP	Manufacturing Automation Protocol
MEB	Memory Expansion Bus (also, Memory Expansion Board)
MHz	Megahertz
MMI	Man Machine Interface
MMS	Manufacturing Message Specification
MSB	Most Significant Bit
msec	Millisecond
MTU	Master Terminal Unit, also Master Station
MUX	Multiplexer
NC contact	Normally Closed relay contact
NEMA	National Electrical Manufacturers Association
NO contact	Normally Open relay contact
O/S or OS	Operating System
OSI	Open Systems Interconnection
oz	Ounce
PC	Power Converter, also Personal Computer
PCI	Pulse Counter Input
PF	Power Factor
PID	Three term controller, proportional, integral, derivative closed-loop control algorithm
PLD	Programmable Logic Device
PLC	Programmable Logic Controller
POU	Program Organization Unit
PPP	Point-to-Point Protocol – A TCP/IP protocol that provides host-to-host network and router-to-router connections. Can be used to provide a serial line connection between two machines.
pps	Pulses Per Second
PWR	Power
RAM	Random Access Memory
RLL	Relay Ladder Logic
ROM	Read Only Memory
router	A device that connects LANs into an internetwork and routes traffic between them
RS232C	EIA Serial data communications standard
RST	Reset
RTOS	Real Time Operating System
RTS	Request To Send
RTU	Remote Terminal Unit

Rx	Receive
SAP	Substation Automation Platform
SBO	Select Before Operate
SCC	Serial Communications Controller
SCADA	Supervisory Control And Data Acquisition
SCTO	Soft Carrier Turn Off
SDLC	Synchronous Data Link Control
SEB	Surge Protection Expansion Board
SFB	Sequential Function Block – one of the IEC 61131-3 programming languages
SFB	Special Function Bus
SFC	Sequential Function Chart graphical language
SOE	Sequence of Events
ST	Structured Text language – one of the IEC 61131-3 programming languages
STS	Status
SWC	Surge Withstand Capability, IEEE C37.90a 1978
TCP/IP	Transmission Control Protocol/Internet Protocol
Tx	Transmit
UART	Universal Asynchronous Receiver Transmitter
UIF	User Interface Function
USART	Universal Synchronous Asynchronous Receiver Transmitter
msec	Microsecond
UVPROM	Ultraviolet erasable Programmable Read Only Memory
VAC	Volts Alternating Current
VAR	Volt-Amperes Reactive
VARH	VAR Hours
VDC	Volts Direct Current
VxWorks	Real Time Operating System made by Wind River for embedded computer systems
W	Watt
Watchdog Timer	Circuit that resets CPU if it fails to execute program
WH	Watt Hours
XB	Expansion Board
XML	Extensible Markup Language – The method used be Telvent for the storing and retrieval of config@WEB RTU data. The data is stored in the form of a series of XML files (files with an XML extension).
XT	External Termination (panel, module or assembly)

APPENDIX B

Index