

### config@WEB Relay Ladder Logic Manual

S2200-AAA-00003 V6.0

	thport Dr. S.W. B, 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** 

Telvent

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 glad 16-007-2008

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

# **Table of Contents**

CHAPTER	1 Introduction	1-1
1.1	Overview	1-1
	1.1.1 Features	1-1
	1.1.2 Application	1-1
1.2	Points Supported	1-2
	1.2.1 Multiple Programs in the RTU	1-2
1.3	Reference Documents	1-2
1.4	How to Determine Your Number-Of-Points Supported	1-3
CHAPTER	2 Installation	2-1
2.1	RTU Requirements	2-1
2.2	PC Requirements	2-1
2.3	Installation Package Contents	
2.4	Installation Procedure	
	2.4.1 Step 1: Install Sentinel Driver	
	2.4.2 Step 2: Install ISaGRAF PRO	
	2.4.3 Step 3: Install Telvent-Provided Components	
	2.4.3.1 ISaGRAF V4.12:	
	2.4.3.2 ISaGRAF V4.20 (and later):	
	2.4.4 Step 4: Updating The Database	2-5
СНАРТЕВ	3 Operation	2-1
3.1	Introduction	
_		
3.2	Creating Simple Programs	
	3.2.2 Opening an Existing Project	
	3.2.3 RTU Communications Settings	
	3.2.4 Starting a New Ladder Diagram (LD) Program	
	3.2.5 Starting a New Function Block Diagram (FBD) Program	
	3.2.6 Changing Variable Attributes	
	3.2.7 "Wiring" Input/Output Points	
	3.2.8 Compiling the Project and Downloading to the RTU	3-38
	3.2.9 Configuring the RTU for RLL	
3.3	Testing Your Programs	-44
	3.3.1 Multiple Programs in the RTU	
3.4	Drivers	-45
_	3.4.1 Types and Number of Drivers	
	3.4.1.1 Drivers for Inputs to Logic	
	3.4.1.2 Drivers for Outputs from Logic	
	3.4.2 Removing/Adding/Modifying Drivers	3-47
	3.4.2.1 Removing a Driver	
	3.4.2.2 Adding a Driver3	
	3.4.2.3 Modifying a Driver3	-49

	3.5	Program Debug	. 3-50
	3.6	Program Simulation	. 3-52
	3.7	Changing the IP Address	. 3-54
	3.8	Changing the Program Cycle Time	
	3.9	Managing Multiple Programs for Different RTUs	
	3.10	Downloading & Recovering Code to/from Target	
		3.10.1 Downloading Code to Target	
		3.10.2 Recovering Code from Target	
	3.11	ISaGRAF Program Maintenance	. 3-62
		3.11.1 Clean Stored Code	. 3-62
		3.11.2 Cleaning Projects	. 3-62
	3.12	Reference Material - RLL Configuration	
		3.12.1 Create RLL Points	
		3.12.1.1 Analog Inputs	
		3.12.1.2 Binary Inputs	
		3.12.1.3 Counters	
		3.12.1.4 Analog Outputs	
		3.12.1.5 Digital Outputs 3.12.1.6 SBO	
		3.12.1.0 SBC	
		3.12.2.1 Analog Inputs	
		3.12.2.2 Binary Inputs	
		3.12.2.3 Counters	
		3.12.2.4 Analog Outputs	
		3.12.2.5 Digital Outputs	
		3.12.2.6 SBO	
		3.12.3 Map Logical Outputs	
		3.12.3.1 Analog Inputs	
		3.12.3.2 Binary Inputs	
		3.12.3.3 Counters 3.12.3.4 Analog Outputs	
		3.12.3.4 Analog Outputs 3.12.3.5 Digital Outputs	
		3.12.3.6 SBO	
		3.12.4 Import/Export Templates	
		3.12.4.1 Import Template	
		3.12.4.2 Export Template	
	3.13	Reference Material - RLL Data Display	. 3-84
		3.13.1 Analog Inputs	. 3-84
		3.13.2 Binary Inputs	. 3-85
		3.13.3 Counters	
		3.13.4 Analog Outputs	
		3.13.5 Digital Outputs	
		3.13.6 SBO	
	3.14	Reference Material - RLL Command Output	
		3.14.1 Analog Outputs	
		<ul><li>3.14.2 Digital Outputs</li><li>3.14.3 SBO</li></ul>	
CH	APTER	4 Programming Principles & Examples	4-1
	4.1	Introduction	4-1
	4.2	MTU/RTU Programming Model	4-2

	4.2.1	BIRM, BIR Notes	4-4
4.3	Hardwar	re AI to RLL (Pseudo) Points	4-4
	4.3.1	Program	4-4
	4.3.2	Variables	
	4.3.3	Wiring	4-5
	4.3.4	RTU Mapping	
	4.3.5	RTU Display	4-8
4.4	Usina In	put SBO RLL (Pseudo) Points	4-11
	4.4.1	Program	
	4.4.2	Variables	
	4.4.3	Wiring	
	4.4.4	RTU Mapping	
	4.4.5	RTU Display	
4.5	Summin	g Accumulator Points	
	4.5.1	Program	
	4.5.2	Variables	
	4.5.3	Wiring	
	4.5.4	RTU Mapping	
	4.5.5	RTU Display	
4.6		g Status Points	
4.0	4.6.1	Program	
	4.6.2	Variables	
	4.6.3	Wiring	
	4.6.4	RTU Mapping	
	4.6.5	RTU Display	
4.7		AO to AI & AI to AO	
4.7	4.7.1	Program	
	4.7.2	Variables	
	4.7.3	Wiring	
	4.7.4	RTU Mapping	
	4.7.5	RTU Display	
4.8	-	g Analog Points	
4.0	4.8.1	Program	
	4.8.2	Variables	
	4.8.3	Wiring	
	4.8.4	RTU Mapping	
	4.8.5	RTU Display	
4.9		AOR to AOW	
4.9	4.9.1	Program	
	4.9.1	Variables	
	4.9.3	Wiring	
	4.9.4	RTU Mapping	
	4.9.4	RTU Display	
4 10			
4.10	4.10.1	BOR to BOW Program	
	4.10.1 4.10.2	Program Variables	
	4.10.2 4.10.3	Variables	
	4.10.3	RTU Mapping	
	4.10.4	RTU Display	
1 1 1			
4.11		ng 2 DOs from Master to 1 SBO Trip/Close to IED	
	4.11.1	Program	4-45

	Variables	
4.11.3	Wiring	4-46
4.11.4	RTU Mapping	4-47
4.11.5	Testing With Debugging	4-49
4.11.6	Program Variation	4-50
4.11.7	RTU Display	4-50
APPENDIX A Glos	ssary	A-1
APPENDIX B Inde	ex	B-1

# **List of Figures**

Figure 1-1 Help Screen for Version 4.12 Using Unlimited Dongle	1-3
Figure 1-2 Help Screen for Version 4.12 Using 128 Dongle	1-4
Figure 2-1 Directory Structure for V4.12 After Installation of the Telvent-Provided Zip File	2-3
Figure 2-2 Directory Structure for V4.20 After Installation of the Telvent-Provided Zip File	2-4
Figure 2-3 Launching ISaGRAF PRO	2-5
Figure 2-4 Selecting config_web_rll	2-5
Figure 2-5 Selecting PRJlibrary	
Figure 2-6 Database Patcher Notice	2-6
Figure 3-1 Launching a New Project	3-2
Figure 3-2 Selecting the Template	
Figure 3-3 Naming a New Proiject	3-3
Figure 3-4 "My_Project" Project Screen	3-3
Figure 3-5 Launching ISaGRAF PRO	3-4
Figure 3-6 Launching a New Project	3-5
Figure 3-7 Finding an Existing Project (V4.12)	
Figure 3-8 Finding an Existing Project (V4.20)	3-7
Figure 3-9 Opening the Prj folder	
Figure 3-10 Opening My_Project	3-8
Figure 3-11 Opening PRJLIBRARY.MDB Under My_Project	3-8
Figure 3-12 "My_Project" Project Screen	
Figure 3-13 Hardware Architecture Screen	.3-10
Figure 3-14 Connection – Properties	.3-11
Figure 3-15 Example of IP Address for Connection	.3-11
Figure 3-16 Completing the Connection Parameters	.3-12
Figure 3-17 Selecting the Type of Program to Create	.3-13
Figure 3-18 Generic Program Name Selected	.3-13
Figure 3-19 Naming the Program	.3-14
Figure 3-20 Save Warning	
Figure 3-21 Ladder Diagram (LD) Work Space	
Figure 3-22 Selecting a Variable Name	.3-15
Figure 3-23 New Variable	
Figure 3-24 Ladder Rung	
Figure 3-25 Naming the New Variable	
Figure 3-26 The New Variable Name	
Figure 3-27 New Variable	
Figure 3-28 Elements With Variable Names	
Figure 3-29 Adding a Rung Comment	
Figure 3-30 Rung Comments	
Figure 3-31 Selecting the Type of Program to Create	
Figure 3-32 Generic Program Name Selected	
Figure 3-33 Naming the Program	
Figure 3-34 Save Warning	
Figure 3-35 Function Block Diagram (FBD) Work Space	
Figure 3-36 Selecting a Variable Name	
Figure 3-37 New Variable	.3-23

	FBD Workspace	
Figure 3-39	Creating a Function Block	.3-24
	Selecting a Function Block	
Figure 3-41	The OR Function Block	.3-26
Figure 3-42	All Variables and Function Block in Place	.3-26
Figure 3-43	Connecting the Blocks	.3-27
Figure 3-44	The Blocks Connected	.3-28
	Selecting the Dictionary	
Figure 3-46	Expanding the Variables Tree	.3-30
	Opening a Variable Attribute Box	
Figure 3-48	Variable Attribute Box - Default	.3-31
Figure 3-49	Variable Attribute Box – Input Variable	.3-32
Figure 3-50	Changing the Output Variables Attributes	.3-33
	Saving the New Variable Attributes	
	Beginning the Wiring Process	
	The SAGE: BIR Wiring List	
	Connecting Input Variable to Wiring Driver	
•	The Wired Input Variables	
	Connecting Input Variable to Wiring Driver	
	Building the Project/Library	
	Download Dialog Box	
	Resource Already Running	
	Download Completed Successfully	
	Configuration – RLL	
Figure 3-62	Mapping the Binary Input Point	3-42
Figure 3-63	Mapping the Binary Input Points	3-42
	Map Logical Outputs	
	Map One SBO Point	
	Mapping One SBO Point	
	Removing a Driver	
	Adding a Driver	
•	Adding a Driver	
	Device Index and Number of Channels	
•	Modifying a Driver	
•	Starting Debugging	
•	Debugging Mode	
•	Debugging Mode	
	Opening the Diagnosis Dialog Box	
-	Diagnosis Dialog Box	
	Stopping the Debug Mode	
	Starting Simulation	
•	Simulation Mode	
	Stop Simulation	
	Changing the IP Address	
	Finding the Cycle Time Changing the Cycle Time	
	Deleting a Project Embedding Code to Target (RTU)	
-		
-	Deleting the Resource	
	Importing the Resource	
	Importing the Resource	
rigure 3-89	Warning Box	.3-60

Figure 3-90 Importing the Resource	
Figure 3-91 Importing the Resource	
Figure 3-92 Importing the Resource	
Figure 3-93 The Resource has been Recovered from the Target RTU	3-62
Figure 3-94 RLL Configuration	3-63
Figure 3-95 RLL Configuration	3-64
Figure 3-96 RLL Analog Input Configuration	3-64
Figure 3-97 RLL Status Configuration	
Figure 3-98 RLL Accumulators Configuration	
Figure 3-99 RLL Analog Output Configuration	3-66
Figure 3-100 RLL Digital Output Configuration	
Figure 3-101 RLL SBO Configuration	
Figure 3-102 RLL Logical Inputs Mapping	
Figure 3-103 RLL Analog Input Configuration	
Figure 3-104 RLL Status Input Point Mapping	
Figure 3-105 RLL Accumulator Point Mapping	
Figure 3-106 RLL Analog Output Point Mapping	
Figure 3-107 RLL Digital Output Point Mapping	
Figure 3-108 RLL SBO Point Mapping	
Figure 3-109 RLL Logical Outputs Mapping	
Figure 3-110 RLL Analog Input Configuration	
Figure 3-111 RLL Status Input Point Mapping	
Figure 3-112 RLL Accumulator Point Mapping	
Figure 3-113 RLL Analog Output Point Mapping	
Figure 3-114 RLL Digital Output Point Mapping	
Figure 3-115 RLL SBO Point Mapping	
Figure 3-116 RLL Data Display	
Figure 3-117 RLL Analog Inputs Display	
Figure 3-118 RLL Status Inputs Display	
Figure 3-119 RLL Accumulator Inputs Display	
Figure 3-120 RLL Analog Outputs Display	
Figure 3-121 RLL Digital Outputs Display	
Figure 3-122 RLL Command	
Figure 3-122 RLL Analog Outputs Command	
Figure 3-124 RLL Digital Outputs Command	
Figure 3-125 RLL SBO Command	
Figure 4-1 MTU/RTU Programming Model	
Figure 4-2 Mapping RLL Points	
Figure 4-3 Mapping All I/O Points	
Figure 4-5 Mapping All 70 Points Figure 4-4 Link Architecture View	
Figure 4-4 Elink Architecture view	
Figure 4-5 BIW_RLL_Status	
5 = 5	
Figure 4-7 BIW_RLL_Status Program Wiring for Analog Inputs	
Figure 4-8 BIW_RLL_Status Program Wiring for RLL Status Output	
Figure 4-9 RTU Configuration – Create RLL Point	
Figure 4-10 RTU Configuration – Map Logical Outputs	
Figure 4-11 RTU Configuration – Map Logical Inputs	
Figure 4-12 Monitoring the Analogs	
Figure 4-13 RLL Data Display	
Figure 4-14 Displaying the RLL Status with Analog 2 Less Than Analog 1	
Figure 4-15 Displaying the RLL Status with Analog 2 Greater Than Analog 1	
Figure 4-16 Link Architecture View	4-11

Figure 4-17 SBO_Functions_FBD Program	
Figure 4-18 SBO_Functions_FBD Program Variables	
Figure 4-19 SBO_Functions_FBD Program Wiring	
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	
Figure 4-23 Commanding SBOR_SBO_Input	
Figure 4-24 Commanding SBOR_SBO_Input	
Figure 4-25 Commanding SBOR_SBO_Input	
Figure 4-26 Resulting RLL Status Inputs Display	
Figure 4-27 Commanding SBOR_SBO_Reset	
Figure 4-28 Resulting RLL Status Inputs Display	
Figure 4-29 Commanding the BOR_DO_Control	
Figure 4-30 Commanding the BOR_DO_Control	
Figure 4-31 Commanding the BOR_DO_Control	
Figure 4-32 Resulting RLL Status Inputs Display	
Figure 4-33 Link Architecture View	
Figure 4-34 Sum_accumulators Program	
Figure 4-35 Sum_accumulators Program Variables	
Figure 4-36 Sum_accumulators Program Wiring	
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	
Figure 4-40 Resulting RLL Accumulator Display	
Figure 4-41 Link Architecture View	
Figure 4-42 Reduce_Status Program	
Figure 4-43 Reduce_Status Program Variables	
Figure 4-44 Reduce_Status Program Wiring	
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	
Figure 4-48 Resulting RLL Status Display	
Figure 4-49 Link Architecture View	
Figure 4-50 Copying Als to AOs & AOs to Als Program	
Figure 4-51 Copying Als to AOs & AOs to Als Program Variables	
Figure 4-52 Aout2Ana Program Wiring	
Figure 4-53 Aout2Ana Program RTU RLL Point Mapping	
Figure 4-54 Aout2Ana Program RTU Input Logic Point Mapping	
Figure 4-55 Exercising Hardware AI with RLL Analog Display	
Figure 4-56 Exercising AOR to AIW	
Figure 4-57 Link Architecture View	
Figure 4-58 Summing Analog Points Program	
Figure 4-59 Summing Analog Points Program Variables	
Figure 4-60 Summing Analog Points Program Wiring	
Figure 4-61 Summing Analog Points Program RTU RLL Point Mapping	
Figure 4-62 Summing Analog Points Program Hardware Al Point Mapping	
Figure 4-63 Resulting RLL Analog Display	
Figure 4-64 Link Architecture View	
Figure 4-65 AOUT Program	
Figure 4-66 AOUT Program Variables	
Figure 4-67 AOUT Program Wiring	
Figure 4-68 AOUT Program RLL RTU Mapping	4-37

Figure 4-69 AOUT Program Logical Input RTU Mapping Figure 4-70 AOUT Program Logical Output RTU Mapping	
Figure 4-71 AOUT Program RTU Display	.4-39
Figure 4-72 Link Architecture View	.4-39
Figure 4-73 DOUT Program	.4-40
Figure 4-74 DOUT Program Variables	.4-40
Figure 4-75 DOUT Program Wiring	.4-40
Figure 4-76 DOUT Program RLL RTU Mapping	.4-41
Figure 4-77 DOUT Program Logical Input RTU Mapping	.4-41
Figure 4-78 DOUT Program Logical Output RTU Mapping	.4-42
Figure 4-79 DOUT Program RTU Display	.4-43
Figure 4-80 Link Architecture View	.4-44
Figure 4-81 DOs to SBO Program	
Figure 4-82 DOs to SBO Variables	.4-46
Figure 4-83 DOs to SBO Wiring	.4-46
Figure 4-84 DOs to SBO Create RLL Point	.4-47
Figure 4-85 DOs to SBO Logical Input RTU Mapping	.4-47
Figure 4-86 DOs to SBO Logical Output RTU Mapping	.4-48
Figure 4-87 Testing With Debugging	.4-49
Figure 4-88 DOs to SBO Program Variation	

## **List of Tables**

Table 1-1 Reference Documents	1-	-2	,
-------------------------------	----	----	---

### 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.

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	R.W. Lewis	The Institution of
Using IEC 1131-3		Electrical Engineers
IEC 1131-3 Programming Methodology	Flavio Bonfatti, Paola Daniela	ISaGRAF
	Monari, Umberto Sampieri	
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

About			×
Support Info			
Internal Version 4.12 Application Version (D	PM) 2002,2	2,4,12	
CTLD.dll DBL.dll DBR.dll DBR.exe Dbt.dll dci.dll Dcp_gctr0Tr.dll Ddb.dll Ddd.dll DDP.dll Dgb.dll DGE.exe DGP.dll DHK.dll DHK.dll Dhp.dll	2001, 2002, 2002,	2, 1, 0 2, 2, 0 2, 3, 1 2, 2, 1, 0 2, 7, 10 2, 7, 10 2, 3, 10 2, 4, 1 2, 2, 2, 2 2, 1, 3 2, 1, 2 2, 1, 2 2, 1, 2 2, 2, 3 2, 2, 3 2, 2, 3 2, 2, 3 2, 2, 3 2, 1, 2 2, 2, 3 3, 1 2, 1, 1 2, 2, 1 2, 2, 3 3, 1 2, 2, 1 2, 1	
ISaGRAF PRO 128 Copyright 1999-2002 Al		ersion 4.1220	020524
OK		Info	

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 <u>is</u> 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

a config_web_rll		
<u>F</u> ile <u>E</u> dit <u>V</u> iew F <u>a</u> vorites <u>T</u> ools <u>H</u> elp		
🚱 Back 🔹 🕥 🕆 🏂 🔎 Search 🔂 Folders	· · · ·	
Address 🛅 C:\ISaGRAF PRO\Tpl\config_web_rll		🖌 🄁 Go
Folders	× Name A	Size
□       ISaGRAF PRO         □       Bin         □       Grp         □       Help         □       Prj         □       Simul         □       Tmp         □       Tpl         □       Config_web_rli         □       Topl         □       Config1         □       EmptyLibmonoresource         □       EmptyLibmultiresource         □       EmptyPrjmonoresource         □       EmptyPrjmultiresource         □       Libmonoresource         □       Libmonoresource         □       Ibmultiresource         □       Prjmonoresource	Config 1 Compile.txt Conf.mtc Conf.mws DDCconf.ini dgb.ini Dic.Ini Dic.Ini Dic.Ini NetworkConf.rtc NetworkConf.rtc NetworkConf.xtc NetworkConf.xts NetworkConf.xws PRJibrary.mdb	1 KB 1 KB 1 KB 1 KB 1 KB 1 KB 2 KB 2 KB 1 KB 1 KB 3,124 KB 1 KB
🗉 🦲 Prjmultiresource	< <	>

#### 2.4.3.2 ISaGRAF V4.20 (and later):

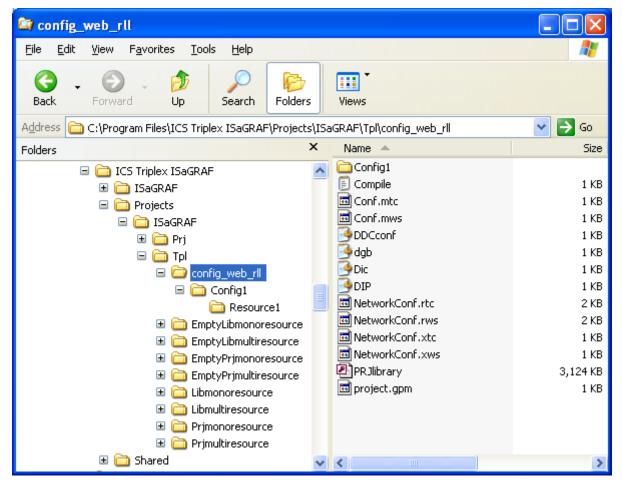
Find the directory "C:\Program Files\ICS Triplex ISaGRAF\Projects\ISaGRAF\Tpl". From the Telvent Installation CD, extract the file "config\_web\_rll.zip" into the "C:\Program Files\ICS Triplex ISaGRAF\Projects\ISaGRAF\Tpl" directory.

**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

🔯 ISaGRAF PRO				
<u>File</u> <u>H</u> elp		_		
New Project/Library	Ctrl+N	A 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4	X 🖂 💓 🕯	८ 🖪 🖂
<mark>ট</mark> Open Project/Library	Ctrl+O			
E <u>x</u> it	Ctrl+Q			
		_		
	Open a projec	t or a library		NUM //

Navigate to Tpl directory and select config\_web\_rll as shown:

#### Figure 2-4 Selecting config\_web\_rll

Open	? 🗙
Look jn: 🔎	Tpl 💽 🖛 🗈 📸 📰 -
🚞 EmptyLibm	ionoresource 📄 Prjmonoresource iultiresource 📄 Prjmultiresource ionoresource iultiresource
File <u>n</u> ame:	PrjLibrary Open
Files of <u>type</u> :	Project/Library files (prjlibrary.mdb)
	Open as read-only     Open in single-resource mode

#### In the next pop-up select PRJlibrary:

Figure 2-5 Selecting PRJlibrary

Open	? 🔀
Look in: 🔎	config_web_rll 💽 🗢 🗈 📸 📰 -
Config1	
File <u>n</u> ame:	PRJlibrary <u>O</u> pen
Files of <u>typ</u> e:	Project/Library files (prjlibrary.mdb)
	Open as read-only     Open in single-resource mode

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

Figure 2-6 Database Patcher Notice

Databa	se patcher	$\mathbf{X}$	
	Project database version :	04.12.02	
	Product database manager :	04.20.05	
	The project database is out of date. Updating the project database is irreversible and can take a few minutes. Do you want to update the project database ?		
	Update	Cancel	

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

# CHAPTER 3

#### 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.

🔯 ISaGRAF PRO						×
<u>File</u> <u>H</u> elp						
New Project/Library	Ctrl+N	d 🗐 🕸	¥ 1	D &	5	s
🚰 Open Project/Library	Ctrl+O					
E <u>x</u> it	Ctrl+Q					
		*				
Create a project or a library				NUM		
create a project or a library				NOM		11.

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.

New		
- Destinatio C:\ISaGR	n folder AF PRO∖Prj	Browse
Name:	<u></u>	
Comment:		
Template:	config_web_rll	•
	config_web_rll EmptyLibmonoresource EmptyLibmultiresource EmptyPrjmonoresource	
	EmptyPrimultiresource	~

Figure 3-2 Selecting the Template

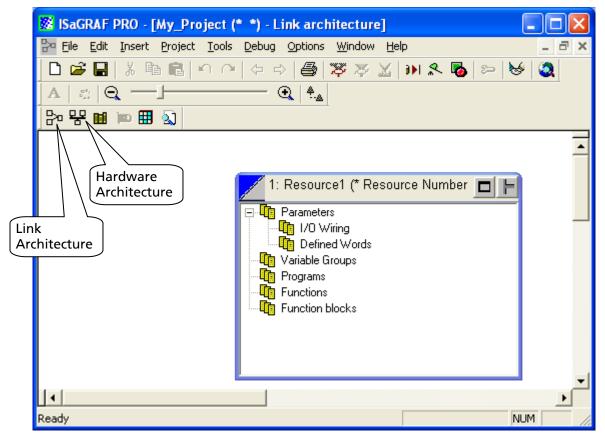
In the "Name" field, type an appropriate name. In the example, "My\_Project" is used. Note that blank spaces are not allowed in the name. You may enter a Comment about the project or you may leave this field blank.

Figure 3-3 Naming a New Proiject

New			
– Destinati C:\\SaGf	on folder 3AF PRO\Prj		Browse
Name:	My_Project		
Comment:			
Template:	config_web_rll		•
	ОК	Cano	cel

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.

😂 Grp		
<u>File E</u> dit <u>Vi</u> ew F <u>a</u> vorites <u>T</u> ools <u>H</u> elp		
🕒 Back 👻 🕑 👻 🍠 Search	Folders	
Address 🗁 C:\ISaGRAF PRO\Grp		🔽 🄁 Go
Folders	× Name A	Size Type
🕝 Desktop		1 KB Shortcut
🗉 🚞 My Documents	ISaGRAF PRO	1 KB Shortcut
🖃 😼 My Computer	licensing	1 KB Shortcut
🗉 🍶 3½ Floppy (A:)	📄 Read Me	1 KB Shortcut
DSK1_VOL1 (C:)		1 KB Shortcut
🛅 16819 Methil	Tutorial	1 KB Shortcut
표 🚞 A_Writer		
🖽 🚞 CANVAS		
표 🚞 Companion Software		
🗉 🧰 Documents and Settings		
🗉 🚞 FLOADER 998		
HP_USB_CD-RW_drivers		
🛅 IE Bookmarks		
🗉 🚞 ImageMate CompactFlash USB		
🖃 🧰 ISaGRAF PRO		
🕀 🧰 Bin		
🗁 Grp		
	× <	
🕀 🦳 Pri		

Figure 3-5 Launching ISaGRAF PRO

Under File, click Open Project/Library.

#### Figure 3-6 Launching a New Project

🔀 ISaGRAF PRO		
<u>File</u> <u>H</u> elp		_
🗋 New Project/Library	Ctrl+N	🕞 🚭 🛱 🏹 🔊 🎗 🐻 🎥 💆 🧕
<mark> O</mark> pen Project/Library	Ctrl+O	
E <u>x</u> it	Ctrl+Q	
		-
Open a project or a library		
open a project or a library		

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).



🔯 ISaGRAF PRO	)	×
<u>File</u> <u>H</u> elp		
	- 副 記 ち つ ( 々 や ) 雪 ( 波 ☆ ズ ) 洲 왕 ( ぬ ) や ) 砂 ( 数 )	
	Open ?X	
	Look in: 🔁 config_web_rll 🗾 🖛 🗈 📸 📰 🗸	
	Config Desktop My Documents PRJlib My Computer	
	3½ Floppy (A:)	
	Image: CD Drive (D:)         Image: CD Drive (E:)           File name         Image: CD Drive (E:)         Image: Dpen	
	Files of tyl       Set Rtueng on 'Rd-source' (H:)       Cancel         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle         Set Revealed a contract of tyle       Set Revealed a contract of tyle       Set Revealed a contract of tyle	
	<ul> <li>SanDisk ImageMate (Z:)</li> <li>My Network Places</li> <li>Nime Fin 2002</li> </ul>	
Ready	Cirus_Fix_0803	N //

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

SagRAF PRO	
File Help	
□ 📚 🗄 🖇 凾 🐨 ㅇ ㅇ ㅇ ◇ 🗐 滋 ☆ 🏹 🖂 ) > 🔗 🚳 🧭	
Open       Image: Config web_fl         Look in:       Config web_fl         Open       Image: Config web_fl         Image: Config web_fl       Image: Config web_fl         Imag	
Ready	//

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

Figure 3-9 Opening the Prj folder

Open			? 🔀
Look in: 🔎	ISaGRAF PRO	- 🗢 🗈	-111 *
i Bin Grp Help Prj Simul Tmp	🛅 Tpl		
File <u>n</u> ame:	PRJLIBRARY.MDB		<u>O</u> pen
Files of <u>t</u> ype:	Project/Library files (prilibrary.mdb)	•	Cancel

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
-------------	---------	-----	---------

Open	?	×
Look jn: 🛅	Prj 💽 🔁 📸 📰 🗸	
Demo ISaGRAF_D My_Project Project1 Project2 Project3	Como Conject4 Demo Conject5	
File <u>n</u> ame:	PRJLIBRARY.MDB	]
Files of <u>t</u> ype:	Project/Library files (prjlibrary.mdb)	

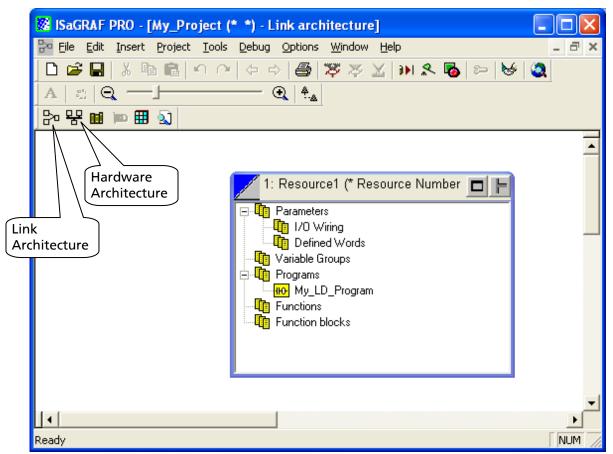
The final step is opening the PRJLIBRARY.MDB file as shown below.

Figure 3-11 Opening PRJLIBRARY.MDB Under My\_Project

Open			? 🔀
Look in: 🔎	My_Project	- 🕈 🖻	) 💣 🎟 •
Config1 환 PRJlibrary.	mdb		
File <u>n</u> ame:	PRJLIBRARY.MDB		pen
Files of type:	Project/Library files (prilibrary.mdb)	•	Cancel

Click Open. My\_Project project screen will appear. This view is called Link Architecture.





#### 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.

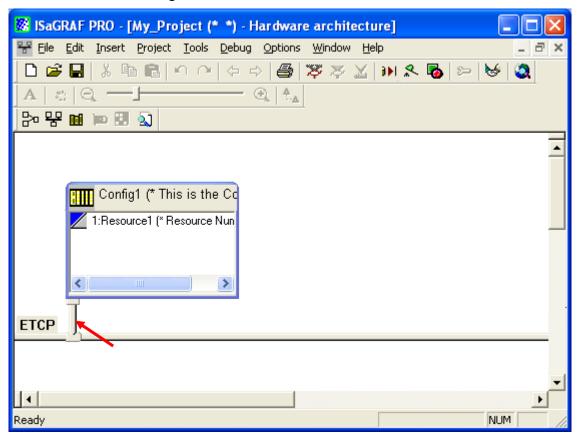


Figure 3-13 Hardware Architecture Screen

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

Connecti	on - Properties		
Network	Parameters		1
	Name	Value	
1	IPAddress	0.0.0.0	IP (
<			>
	ОК	Cancel	) Apply

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

Figure 3-15 Example of IP Address for Connection

Co	onnect	ion - Properties		
ſ	Network	Parameters		1
		Name	Value	
	1	IPAddress	172.18.150.51	IP (
		••••		
	<	Ш		>
		OK	Cancel	Apply

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

🔀 ISaGRAF PRO - [My_Project (* *) - Hardware architecture]	
🚼 File Edit Insert Project Tools Debug Options Window Help	- 8 ×
📴 🚅 🔜 🕹 🛍 🛍 🗠 🏹 🗢 😂 🎒 💥 🌫 🔛 🔃 🗠 🕅	ৰ্জ 🧕
문 명 📾 📼 🕄 🔬	
Link architecture	<u> </u>
Config1 (* This is the Co	•
ETCP	
	▼ ►
Link architecture	

# 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, <u>right</u>-click on Programs and select LD: Ladder Diagram as shown below.

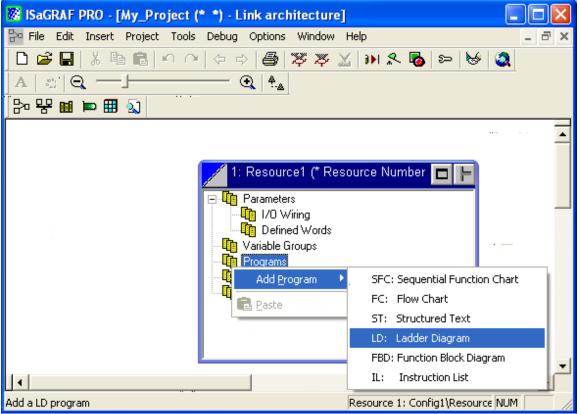


Figure 3-17 Selecting the Type of Program to Create

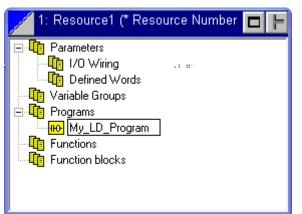
After the type of program has been selected, the generic title will remain highlighted so that you can change the name to something more meaningful.

🖊 1: Resource1 (* Resource Number 🗖 📙
🖃 🗓 Parameters
- 🗓 1/0 Wiring
Defined Words
Variable Groups
Programs
HO UntitledLD
-

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

ISaGRA	F PRO
⚠	The project architecture has been modified, and must be saved before opening document. Continue?
	<u>Y</u> es <u>N</u> o

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
-------------	--------	---------	------	------------

🗗 DGE -	[1: Resour	ce1 - My_LD	Program (* 1	")]				(	
🐮 Eile 🤅	<u>E</u> dit Tools <u>[</u>	<u>ebug</u> Options	<u>W</u> indow <u>H</u> elp						_ 8 ×
F2: HEIE F	3: ⊞HE F4: 👫	F5:-OH F6:⊕⊞	F7: 🗄 🗗 F8: 🔂 F	'9: <b>≫ +F9:@&gt;</b> -∃	ezhe =	: 🔶			
🖻 🖼 🛛	🕺 🖻 🛢	n n   <b>E</b>	) 🚧   🖬   🎽	z   🔏   🗶 🛙	6				
A   #		J	🔍   🖽	€€					
		Ŧ	Ŧ	<b>.</b>	Ŧ	Ŧ	<del>.</del>	Ŧ	^
,	Ļ	+	+	+	+	+	+	+	
·	+	+	+	+	+	+	+	+	4
ŀ	+	+	+	+	+	+	+	+	-
r	+	+	+	+	+	+	+	+	4
	+	+	+	+	+	+	+	+	-
									>
Ready								NU	м //

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.

Select variable							
Input_01		_				BOOL	-
Global + local	C Local						
Name		Alias	Туре	Direct	Dime	Comment	
, Г	OK	7			(	Cancel	
L	5.0						

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.

New Vari	able 🛛 🔀
Name:	locut 01
Name.	Input_01
Туре:	BOOL
Scope:	Global
Alias:	
Comment:	Input element comment
	OK Cancel

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

Figure	3-24	Ladder	Runa
inguic		Lagadi	nung

🗗 DGE - [1: Resource	1 - My_LD_Program	ı (* *)]					
😰 Eile Edit Tools Del	bug <u>O</u> ptions <u>W</u> indow	<u>H</u> elp					- 8 ×
F2: HHE F3: HHE F4: HH	-OH F6: (DHE F7: HE) F8:	τ <mark>0</mark> Τ F9:→» <b>+</b> F9:→@> ·	+ <b>⊧</b> ‡++ =:\$	11			
🖉 🖬   👗 🖻 💼	≏ ∩   🎒   🗛   🛙	1 🛓 🖌 🙎	😼 🔬				
<b>∧</b> ∣⊯∣Q —⊢	•	E E					
		<b>T</b>	Ŧ	Ŧ	Ŧ	Ŧ	
(* *)	<b></b> + +	+	+	+	+	+	+
[1] Input_C	)1						
, ⊢ <del>,</del> ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,		H +	+	+	+	+	+
• •	+ +	+	+	+	+	+	+
		+	+	+	+		+
í í		Ŧ	r	·	r	ŕ	Ť
· ·	+ +	+	+	+	+	+	+
Ready						NUM	

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
-------------	----------------	----------

DGE - [1: Resource1 - My_LD_Pr	ogram (* *)]	
📅 Eile Edit Tools Debug Options W	indow <u>H</u> elp	- 8 ×
	Select variable	
≝ ■   ≵ № ®   ≏ ∩   ⊕   (  ▲   Ⅲ   Q, — ] — — — — — — — — — — — — — — — — —	BOOL Global + local C Local	•
(* *) + / [1] Input_01 + + + +	Name Alias Type Direct Dime Comment Input_01 BOOL Internal	
· · ·	OK Cancel	
Ready		JUM

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

Figure 3-26 The New Variable Name

Select variable						
Output_01	_				BOOL	4
Global + local C Local						
Name	Alias	Туре	Direct	Dime	Comment	
Input_01		BOOL	Internal			
ОК				(	Cancel	

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

Figure 3-27 New Variable

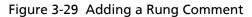
New Varia	able 🔀
Name:	Output_01
Туре:	BOOL
Scope:	Global
Alias:	
Comment:	Output element comment
	OK Cancel

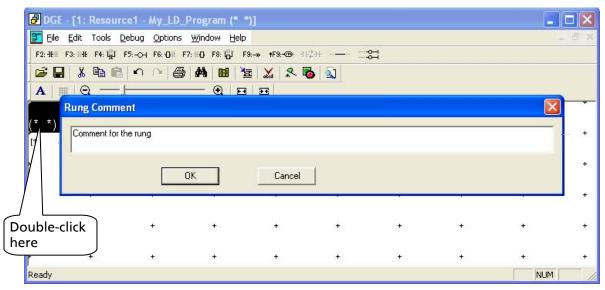
Now both elements of our ladder have variable names as shown below.

Figure 3-28 Elements With Variable Names

DGE - [1: Resource1 - /	My_LD_Program	(* *)]				[	
😰 File Edit Tools Debug	<u>O</u> ptions <u>W</u> indow	Help					- 8 ×
F2: HHE F3: HHE F4: HE F5: -OH	F6: -():II F7: -II:(): F8: Ü	"] □ F9:—» +F9:-@> ÷	IF <b>‡</b> ₩ =:\$	11			
🖉 🖬 👗 🖻 💼 🕥	°   🖨   🛤   🖬	1 🗏 🔛	🗞 🔬				
<b> </b> ▲	•	£. £.					
· · · ·			Ŧ	<del>,</del>	Ŧ	+	
(* *) + +		+	+	+	+	+	+
[1] Input_01	Output_01						
· · ·		+	+	+	+	+	+
			+				
	+ +	+	+	+	+	+	+
	+ +	+	+	+	+	+	+
• •	+ +	+	+	+	+	+	+
, Ready						NUN	1 _//

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





The rung comment results are shown below.

Figure 3-30 Rung Comments

BGEPosource1 - My_LD_Program (* *)]		×
Eile Save Jis Debug Options Window Help	7	×
F2: HEE A: HE F4: HT F5: -OH F6: DE F7: HD F8: HT F9:-∞ HF9:-∞ HF2:HE == 3=		
🖆 🖬 🕼   🍳   🍋   🎒   🏙   🖉   🔬   ዱ 🗞   🔬		
▲   ⅲ   Q,		
		^
(* Comment for the rung *)	+	
[1] Input_01 Output_01		
	+	
• • • • • • •	+	
		<b>~</b>
Ready		

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, <u>right</u>-click on Programs and select FBD: Function Block Diagram as shown below.

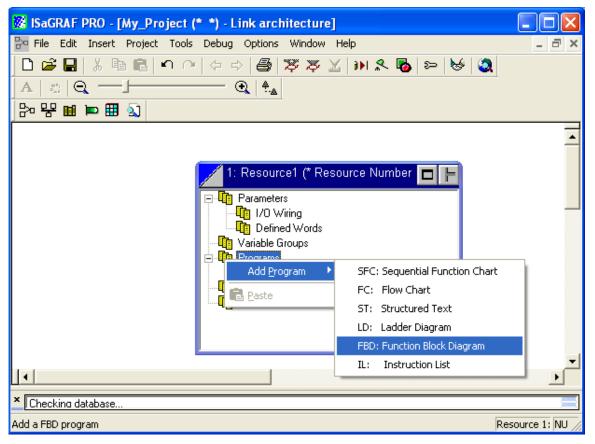
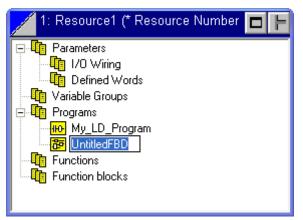


Figure 3-31 Selecting the Type of Program to Create

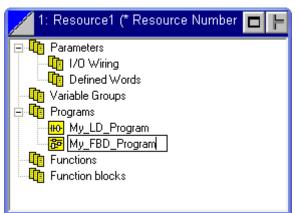
After the type of program has been selected, the generic title will remain highlighted so that you can change the name to something more meaningful.





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

ISaGRA	F PRO
⚠	The project architecture has been modified, and must be saved before opening document. Continue?
	<u>Y</u> es <u>N</u> o

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.

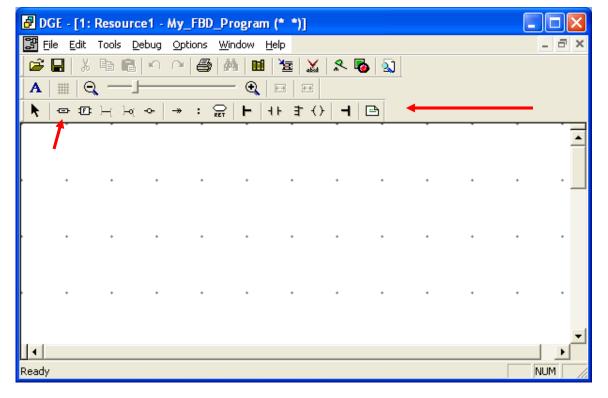


Figure 3-35 Function Block Diagram (FBD) Work Space

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.

Select variable							×
Input_02 Global + local	C Local	_				BOOL	•
Name Input_01 Output_01		Alias	Type BOOL BOOL	Direct Input Output	Dime	Comment	
[	OK				0	Cancel	

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

New Varia	ıble 🔀
N	Laut 02
Name:	Input_02
Туре:	BOOL
Scope:	Global
Alias:	
Comment:	
	OK Cancel

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

Figure 3-38 FBD Workspace

🗗 DGE	- [1:	Resourc	e1 - My	_FBD_I	Program	(* *)]					_	
		Tools <u>D</u>	ebug <u>O</u>	ptions <u>V</u>	<u>V</u> indow <u>H</u>	<u>H</u> elp						- 8 ×
j 🗳 🔒	1   X	<b>F</b>	<b>n</b> 0		#4   <b>11</b>	2	( I 🔍 🖣	3 🔊				
<b>A</b>   #	#   G	<u>ر</u>			- 🔍							
▶   •	∍ (C	$[-1] \to [$	•   →	: 🔤	<b>}   ⊢</b>   ·	11 1 4	)   <b>-</b>	æ				
	·	Ť	Ť	Ť	*	Ť	Ť	Ť	Ť	*	Ť	· ·
ŀ	· 」	Input_0:	<u> </u>	*	+	+	+	*	*	+	+	•
	l	mpor_b.	2									
ŀ	*	*	+	*	+	*	+	*	*	+	+	•
												_
Deadu												
Ready											N	

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

🗗 DGE - [1: Resource1 -	My_FBD_Program	n (* *)]			
🔀 File Edit Tools Debug	<u>O</u> ptions <u>W</u> indow !	Help			- 8 ×
🎽 🖬   👗 🖻 💼 🗠	🗠 🕼 🖊 🖬	🛓 👗 🕷	3 🔊		
<b>∧</b>	•	F-8 - F-6			
🕨 📼 🖬 H H 🗠		न⊩ <b>∄ () न</b>	<b></b>		
Function Block			* *	Ť	· · ·
, , <u>Input_02</u>	L * *	• •	* *	+	
horTot					
	* *	+ +		+	
	* *	+ +	* *	+	• •
					<b>▼</b>
Add a Function Block				Γ	

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.

Fi	gure 3-40 S	selecting a Function Block
Select Blocks		X
OR Blocks Parameters		Help
Name NOT_MASK ODD OR OR_MASK POW R_TRIG RAND REPLACE RIGHT	Type SFU SFU SFU SFU SFU SFB SFU SFU SFU	Comment         bit to bit negation         Odd parity         Boolean OR of two or more terms         Analog bit to bit OR mask         Power calculation         Rising edge detection         Random value         Replace sub-string         Extract right of a string
Inputs: Select number of inputs	OK	Instance:

### Below is the result of adding an OR Function Block.

Figure 3-41 The OR Function Block

	🗗 DGE - [1	: Resourc	e1 - My_	_FBD_P	rogra	m (* *)]					_	
	🔓 Eile Edit	Tools <u>D</u> e	ebug <u>O</u> pl	tions <u>W</u> i	ndow	<u>H</u> elp						- 8 ×
	) 🗃 🖬 🛛 🐰	B 🔒	$\mathbf{v}_{-} \cap$		<b>4</b>   🖬	≛ .	🖌   🔍 🛚	o 🔊				
	<b>A</b>   ⊞   €	⊇ — ŀ			- 🔍		£					
	k 📼 E	⊧ <u>⊢</u> [ ∔0[	•   →	: OR RET	H	4F 🗄	$\left  \cdot \right  $	æ				
	ľ i	Ť	Ť	Ť	Ť	Ť	Ť	Ť	Ť	*	Ť	Ť.
	• •	+	*	+	+	+	+	+	+	+	+	•
		Input_02	$\neg$	ſ	OR							
		•		. –			*	*	*			
						F						
· · · · · <b>L</b> · · · · · · ·												
	• •	+	*	. l		•	+	+	+	+	+	•
												_
Ready NUM	Beadu											

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.

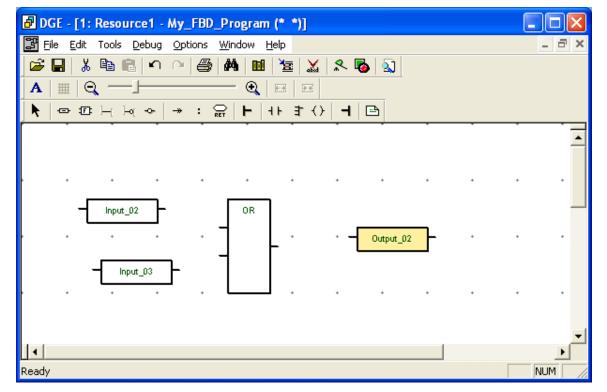
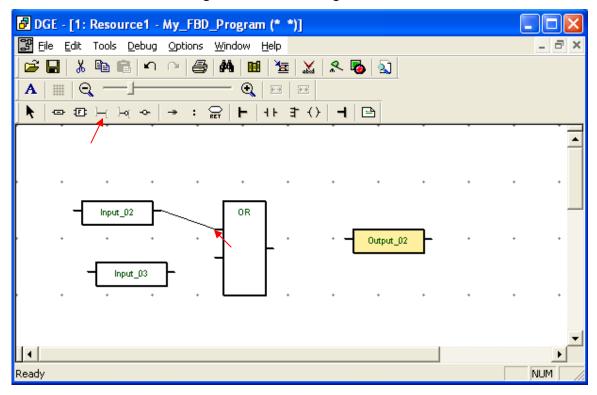


Figure 3-42 All Variables and Function Block in Place

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.



DGE - [1: Resource1 - My_FBD_Program (* *)]	
File Edit Tools Debug Options Window Help	- 8 ×
🚔 📮 👗 🛍 🖻 🔍 🔿 🎒 🎒 🛍 🖉 🔬 🗶 🔕	
▶   ━ ☜ ⊣ ⊣ • = + - = - + = - + =	
* * * * * * * * *	
· · · · · · · · Output_02	
Input_03	
	-
Ready	NUM /

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.

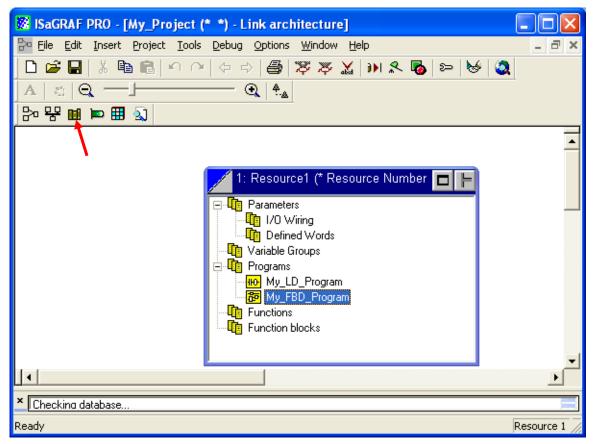


Figure 3-45 Selecting the Dictionary

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	e Variables Tree
-------------	---------------	------------------

🔀 ISaGRAF PRO - [My_Project (*	*) - Dictional	y - Varial	oles]			. 🗆 🛛
Eile Edit Debug Tools Options	<u>W</u> indow <u>H</u> elp					- 8 ×
A   😄   🎟						
🖬 🕹 🖻 🖻 🖉 🏉 🖗		-	6			
응 명 🖬 🖿 🗐 🕥						Ļ
	All variable					
BM. Variables ↑ BA. Resource1 (Config1)	Name	Alias	Туре	0	Init. value	Dimensi.
Any Group	Input_01		BOOL			
All variables	Output_01 Input_02		BOOL BOOL			
Global variables	Input_02		BOOL			
My_CD_Program	Output_02		BOOL			
	<	1111				>
× Checking database						
Ready					Re	esource 1

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

🔀 ISaGRAF PRO - [My_Project (*	*) - Dictionary - Variables]						
Eile Edit Debug Tools Options	<u>W</u> indow <u>H</u> elp	- B ×					
A 🚓 🔳							
🔲 🕹 🖬 🖻 🗠 🔿 🎒 🖗							
음 명 🖬 📼 🕄 🕥							
🔼 😽 🌳 🕩	All variables						
<ul> <li>B ▲ Resource1 (Config1)</li> <li>B ▲ Resource1 (Config1)</li> <li>B ▲ Any Group</li> <li>All variables</li> <li>Global variables</li> <li>My_LD_Program</li> <li>My_FBD_Program</li> </ul>	NameAliasTypeInput_01BOOLOutput_01BOOLInput_02BOOLInput_03BOOLOutput_02BOOL	() Init. value Dimensi.					
	<	>					
× Checking database							
Ready		Resource 1					

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

Input_01				
Name Input_01	Alias		Туре	BOOL
0	Init, value		Dimension	
Group None 💌	Attribute	Free	Scope	Global 💌
Direction Internal 💌	Retain	No	Wiring	
Address	Comment	Input element com		
	ОК		·Cancel	

### Figure 3-49 Variable Attribute Box – Input Variable

Input_01					
Name	Input_01	Alias		Туре	BOOL
0		Init. value		Dimension	
Group	None	Attribute	Read	Scope	Global 💌
Direction	Input 💌	Retain	No	Wiring	
Address		Comment	Input element com		
		ОК		Cancel	

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.

🔀 ISaGRAF PRO - [My_Project (* *) - Dictionary - Variables]							
Eile       Edit       Debug       Tools       Options       Window       Help							
A 🛛 🙁 🗍 🎟							
🔚 👗 🖻 💼 🗠 🗠 🚭 🐺 📑 🛃 🤰 👬 📑 🛠 🌄							
· 양 맨 💌 🕄 🔊							
🔼 🚭 🏘 💶 All variables							
Name     Alias     Type     Init. value     Dimensi.       Image: Market Strength     Input 01     BOOL							
Any Group Input_01 BOOL							
Global variables Input_02 BOOL							
My_LD_Program Input_03 BOOL							
My_FBD_Program Output_02 BOOL							
× Checking database							
Ready Resource 1							

Figure 3-50 Changing the Output Variables Attributes

Output_01					
Name	Output_01	- Alias		Туре	BOOL
0	<u> </u>	Init. value		Dimension	
Group	None	Attribute	Write	Scope	Global 💽
Direction	Output	Retain	No	Wiring	
Address	[	- Comment	Output element co		
		ОК		Cancel	

Repeat this process for Output\_02.

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

Figure 3-51 Saving the New Variable Attributes

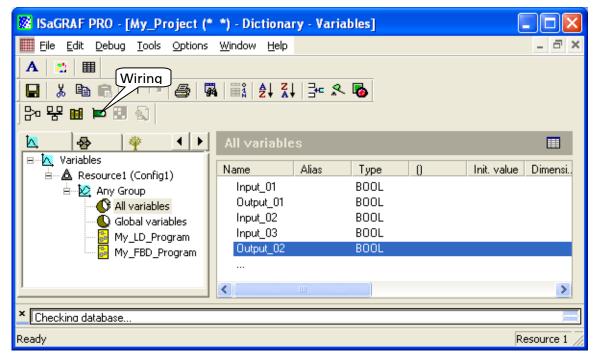
🔯 ISaGRAF PRO - [My_Project (*	*) - Dictionar	y - Variab	les]			
Eile Edit Debug Tools Options	<u>W</u> indow <u>H</u> elp					- 8 ×
A 🗠 🖿						
┣┫│Ӽ╚┓╔╢╹∩()/@/ Ø	ă I ≣i I ≜↓ Z↓	⊒•∈ 🔍 [	0			
윤 명 🖬 🕨 🕄 🕥						
	All variable:	S				
B <mark>A</mark> Variables È <b>A</b> Resource1 (Config1)	Name	Alias	Туре	0	Init. value	Dimensi.
ia → 22 Any Group	Input_01		BOOL			
C All variables	Output_01		BOOL			
🖳 💽 Global variables	Input_02		BOOL			
My_LD_Program	Input_03		BOOL			
👸 My_FBD_Program	Output_02		BUUL			
·	<					>
× Checking database						5
Ready					R	esource 1

# 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".

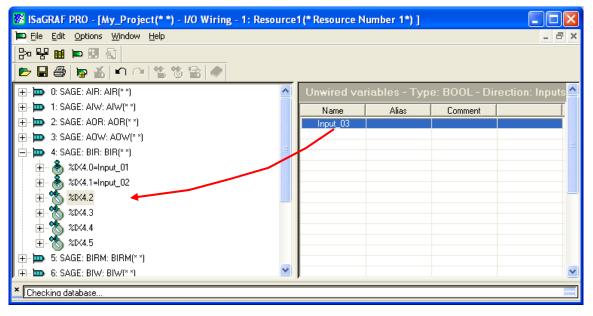
Expand and select the SAGE: BIR wiring list as shown below. Notice that the Input variable appears in this list.

Figure 3-53 The SAGE: BIR Wiring List

🔀 ISaGRAF PRO - [config_web_	rll(* *) - I/O Wiring	- 1: Resource1(* R	esource Number 1*)	1 - 🗆 🗙		
P File Edit Options Window He	p			- @ ×		
응 명 🖬 🖿 🕄 🕥						
🖻 🖩 🚭 🐚 🖄 🗠 🕅	š 🐮 😭 🥏					
🕀 👜 0: SAGE: AIR: AIR(* *)	Unwired v	ariables - Type:	BOOL - Direction:	Inputs 🔷		
1: SAGE: AlW: AlW(* *)	Name	Alias	Comment			
庄 🖿 2: SAGE: AOR: AOR(* *)	Input_01					
🗄 🖿 🖿 3: SAGE: AOW: AOW(* *)						
4: SAGE: BIR: BIR(* *)	Device Selection					
⊞ 📸 %IX4.0	Target: VXW-TARGET	T		-		
⊞ 📸 %IX4.1		-5				
⊞- <b>*</b> %IX4.2	4: SAGE: BIR: BIR(* *)					
⊞ 🐔 %IX4.3	Device index:	4				
± 🐔 %IX4.4		<u></u>				
🛨 🐔 %IX4.5	Number of channels:	6 🚊		=		
🕀 👜 5: SAGE: BIRM: BIRM(* *)	<u>N</u> 2					
庄 🛅 6: SAGE: BIW: BIW(* *)	Binary Input for read, BC	)OL 🚤				
庄 🛅 7: SAGE: BOR: BOR(* *)						
庄 🛅 8: SAGE: BOW: BOW(* *)						
庄 🛅 9: SAGE: CNTR: CNTR(* *)				_		
🕂 🚥 10: SAGE: CNTW: CNTW(*						
+ 11: SAGE: SBOR: SBOR(* *)						
🛨 🏧 12: SAGE: SBOW: SBOW(*	1					
	Help		ОК	Close		
-	J.					

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
-------------	------------	-------	-------------	---------------

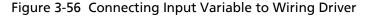


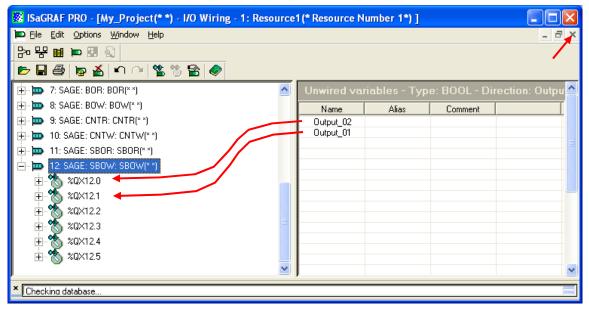
The result is as shown below.



ISaGRAF PRO - [My_Project(* *) - I/O Wiring - 1: Resource	1 (* Resource N	umber 1*)]			X
📼 <u>F</u> ile <u>E</u> dit <u>O</u> ptions <u>W</u> indow <u>H</u> elp				_ 6	F ×
유명 🖩 📼 🕄 🖏					
📂 🖬 🎒 🐚 🔏 🔍 🔍 🐮 🖏 🗟 🧼					
⊕ 🛅 0: SAGE: AIR: AIR(* *)	Unwired val	riables - Typ	e: BOOL - Di	rection: Input	s ^
	Name	Alias	Comment		Ī
🚊 🔤 4: SAGE: BIR: BIR(* *)					
	L				
표···· 💑 %IX4.2=Input_03					
±					
±					
🛨 🐔 %IX4.5					
					~
	r				_
Checking database					

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.





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).

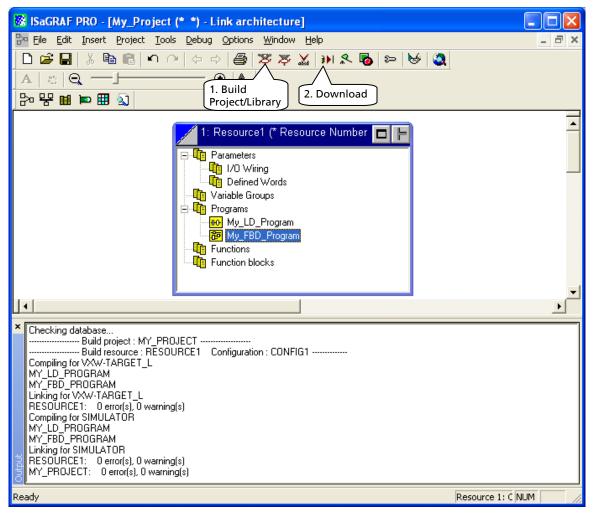
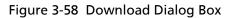


Figure 3-57 Building the Project/Library



Download	×
Check resources to download their code:	
Config1: 1:Resource1 (* Resource Number 1 *)	
Toggle Select All Unselect All	
Save on target after download	
Start after download	
Download Cancel	

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

Resource already running 🛛 🔀			
RESOURCE1 in CONFIG1			
Resource is already running. Do you want to stop it ?			
Stop and download	No Stop, No Download		

**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.

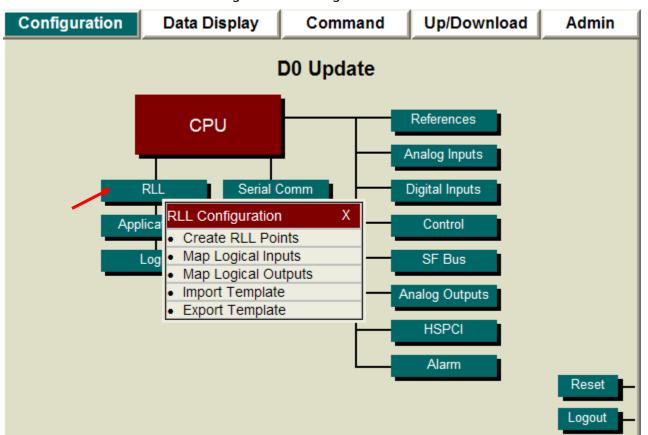
Figure 3-60 Download Completed Successfully

ISaGRAF PRO - [My_Project (* *) - Link architecture]	
Project Insert Project Iools Debug Options Window Help	- 8 ×
🗋 🖆 🔚 🕺 🛍 💼 📭 🖓 🗇 🎒 🎒 🦉 🎘 🎒 🛠 🐻 🏷 😽 🧕	
응 뿡 🖩 📾 🖽 🔊	
1: Resource1 (* Resource Number         Parameters         1/0 Wiring         Defined Words         Variable Groups         Program         10 My_LD_Program         10 My_FBD_Program         10 My_FBD_Program         10 Functions         11 Function blocks	
	•
16-September-03 15:51 Download is completed successfully	
Ready Resource 1: C	

**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

RLL Logical Inputs Mapping				
Туре	Number	Map		
Analog Inputs	0	MAP		
Binary Inputs	3	MAP		
Counters	0	MAP		
Analog Outputs	0	MAP		
Digital Outputs	0	MAP		
SBO	0	MAP		
Back				

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

Figure 3-63 Mapping the Binary Input Points

# RLL Status Input Point Mapping

Detet	Device Neme	Deint Name	Invest 5		Course Deinte
Point	Device Name	Point Name	Invert 🄊		Source Points
0	Hardware DI	DI_PNT_1	🔾 Yes 💿 No	_	Hardware DI 🛛
1	Hardware DI	DI_PNT_2 DI_PNT_1	⊖Yes ⊙No		Search
2	Hardware DI	DI_PNT_3 DI_PNT_3	OYes ⊙No		SPARE
	Hardward D1		0163 0110		Select All points
					DI_PNT_1
					DI_PNT_2
					DI_PNT_3
					DI_PNT_4
					DI_PNT_5
					DI_PNT_6
					DI_PNT_7
					DI_PNT_8
					DI_PNT_9
					DI_PNT_10
					DI_PNT_11
					DI_PNT_12
					DI_PNT_13
					DI_PNT_14
					DI_PNT_15
					DI_PNT_16
				$\leq$	
					Cancel Submit
					Cancer J Submic

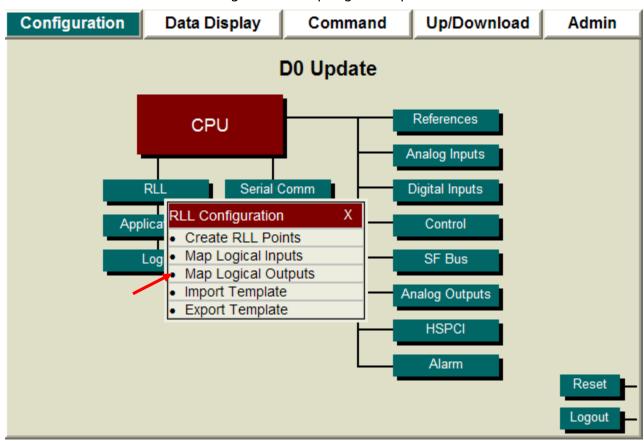


Figure 3-64 Map Logical Outputs

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

RLL Logical Outputs Mapping				
Туре	Number	Мар		
Analog Inputs	0	MAP		
Binary Inputs	0	MAP		
Counters	0	MAP		
Analog Outputs	0	MAP		
Digital Outputs	0	MAP		
SBO	2	MAP		
		Back		

Figure 3-65	Map One SBO Point
rigare 5 05	

Map two SBO hardware points as shown.

RLL SBO Point Mapping							
Point	Device Name	Point Name	State 🄊		Source Points		
0	Hardware Controls	SBO 1	⊙ Trip OClose		Hardware Controls	~	
1	Hardware Controls	SBO 2	⊙Trip OClose		SPARE		
					Select All points		
			SBO 1		SBO 1		
			SBO 2		SBO 2		
					SBO 3		
					SBO 4		
					Cancel	Submit	

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 realworld 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:** <u>Plan ahead</u>. 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

🔀 ISaGRAF PRO - [My_Project(* *) - I/O Wiring - 1: Resource1(* Resource N
Eile Edit Options Window Help
음 명 🗰 📼 🕄 🔬
🖻 🖬 🖨 🕞 🎽 🗠 🍽 📽 🐮 😭 🧇
🗄 🖮 🧰 2: SAGE: AOR: AOR(* *)
i∰…im 3: SAGE: AOW: AOW(* *)
🗄 🖮 🧰 4: SAGE: BIR: BIR(* *)
🗄 🖮 🧰 5: SAGE: BIRM: BIRM(* *)
🗄 🖮 🧰 6: SAGE: BIW: BIW(* *)
🗄 🖮 🧰 7: SAGE: BOR: BOR(* *)
🗄 🖮 🏧 8: SAGE: BOW: BOW(* *)
🗄 🖮 🧰 9: SAGE: CNTR: CNTR(* *)
🗄 🖮 🧰 11: SAGE: SBOR: SBOR(* *)
É I2: SAGE: SBOW: SBOW(* *)

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

Figure 3-69 Adding a Driver

Device Selection	
Target: VXW-TARGET_L	
SAGE: AIR: AIR(* *)	•
SAGE: BOR: BOR(* *) SAGE: BIRM: BIRM(* *) SAGE: SBOR: SBOR(* *) SAGE: AOW: AOW(* *)	
SAGE: AIR: AIR(* *) SAGE: BOW: BOW(* *) SAGE: BIR: BIR(* *) SAGE: SBOW: SBOW(* *) SAGE: CNTR: CNTR(* *) SAGE: AIW: AIW(* *) SAGE: BIW: BIW(* *) SAGE: CNTW: CNTW(* *)	
Help	OK Close

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

Device Selection	
Target: VXW-TARGET_L	
SAGE: AIR: AIR(* *)	•
Device index: 0	
Analog Input for read, 32 bit	
Help	OK Close

### 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

🔀 ISaGRAF PRO - [My_Project(* *	) - I/O Wiring - 1: Resource1(* Resource Number 1*) ]
P File Edit Options Window Help	Device Selection
<ul> <li>0: SAGE: AIR: AIR(* *)</li> <li>1: SAGE: AIW: AIW(* *)</li> <li>2: SAGE: AOR: AOR(* *)</li> <li>3: SAGE: AOW: AOW(* *)</li> <li>4: SAGE: BIR: BIR(* *)</li> <li>5: SAGE: BIRM: BIRM(* *)</li> <li>5: SAGE: BIW: BIW(* *)</li> <li>6: SAGE: BIW: BIW(* *)</li> <li>7: SAGE: BOR: BOR(* *)</li> <li>8: SAGE: BOW: BOW(* *)</li> <li>9: SAGE: CNTR: CNTR(* *)</li> <li>10: SAGE: CNTW: CNTW(* *)</li> <li>11: SAGE: SBOR; SBOR(* *)</li> <li>12: SAGE: SBOW: SBOW(* *)</li> </ul>	Target: VXW-TARGET_L     O: SAGE: AIR: AIR(**)     Device index: 0     O •     Number of channels: 14     Analog Input for read, 32 bit
1,2003,500,000 03 07	Help OK Close

# 3.5 **Program Debug**

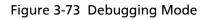
You can debug your program while the program is running in the RTU. If you prefer to debug your program off-line, see section 3.6 Program Simulation.

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.

🗗 DGE - [1: Resou	irce1 - My_LD_F	Program (* *)	1			
📅 Eile Edit Tools	Debug Options	<u>W</u> indow <u>H</u> elp				- 8 ×
F2: HHE F3: HHE F4: GE	条 Debug 🛛 🔶	F8: T <mark>0</mark> T F9	:	1e#1e ==	H	
🖉 🖬 👗 🖻 🛙	o Simulation	11 🖄	🔛 😤	😼 🔬		
A         Q	🚊 Spy Selection	<b>Q</b> =	- <del></del>			
	ଜିଙ୍ଗ Debug FB F	11	<b>.</b>	· ·	Ŧ	· ·
(* *)	+					+
[1] Ing	ut 01 Output	01	т	Ŧ	Ŧ	Ŧ
, <u> </u>	× ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰	+		+	+	+
+ +	+	+	+	+	+	+ 💌
						>
						NUM   //

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.



🗗 DGE - [1: Res	ource1 - My_Ll	)_Program (* *)	1					
📅 Eile Edit Too	ols <u>D</u> ebug <u>O</u> ptior	ıs <u>W</u> indow <u>H</u> elp						- 8 ×
🎯 🖬 主 🕺	🤇 66° 🔬   🌆							
<b>  A</b>		• =	₽€					
	<b>-</b>	Ŧ	Ŧ	Ŧ	Ŧ	Ŧ	Ŧ	^
(* *)	+	+	+	+	+	+	+	
[1]	Input_01 Out	put_01						
· +	+ +	< > <del></del> +		+	+	+	+	
· · · · · · · · ·	+	· <b>4</b> ·····	+	+	+	+	+	>
×								
× Name Re:	source Value	Physical Scop	e Alias	Comment 1	ype Dire	ection		
<u>ज</u> ि								
Ready							NUM	

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.

DGE - [1: Resource1 - My_LD_Prog	m (* *)]		
🐮 Eile Edit Tools Debug Options Wind	w <u>H</u> elp		_ 6 ×
🗃 🖬 🟦 🗶 60 🔬 🌆			
A   Ⅲ   Q	Q 🖽 🖂		6 - E - E
· · ·	Write BOOL varia	able 🛛 🔀	^
(* *) [1] Input 01 Output 01		Variable INPUT_01	
	0 [	FALSE TRUE 1	
· · ·	Lock	Unlock	>
Name Resource Value Physic	Scope Alias Commen	it Type Direction	
Ready			

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

🔏 ISaGRAF PRO - [My\_Project (\* \*) - Link architecture] Pa File Edit Insert Project Tools Debug Options Window Help - 8 A | 🚓 | 🗨 🗕 🚽 Download 옷 Stop Debug 🏱 무 🖬 🗖 🚮 🌄 Simulation 🔊 🐵 🕨 N 🕨 🙆 🌆 🞾 On-line Change: Download SI 🗖 On-line Change: Update Start 혠 Stop Start from code saved on Target Save Code on target Clean stored code Diagnosis 🔹 R<u>e</u>fresh Status Real Time / Cycle to Cycle Execute one cycle 🚷 Change Cycle timing Ionfig1\Resource1 NUM

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.

S2200-AAA-00003

Figure 3-76 Diagnosis Dialog Box

1:Resource1 (Config1) - Diagnosis		×
Timing System Variables Locked Variables		
INPUT_01	Unlock <u>A</u> ll	1
	<u>U</u> nlock	ī I
OK Cancel	Apply	

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

🗗 DGE - [1: Resou	irce1 - My_LD_Program	· (* *)]			
😰 Eile Edit Tools	Debug Options Window	Help			- 8 ×
🎯 🖬 🚊 💉	条 Stop Debug				
A         Q	Simulation	. <del>Et</del> <del>Et</del>			
r -	🚊 Spy Selection				
(* *)	ຜູ້ຜູ້ Debug FB F11	+	+ +	+	+
[1] Ing	out_01 Output_01	·			
, ⊢ <b>,</b> ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	· · · · · ·	+ +	+ +	+	+
· · ·	+ +	+	+ +	+	+ >
×					
× Name Resou	irce Value Physical	Scope Alias	Comment Type	Direction	
Debug					

# 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

B DGE - [1: Resource1My_LD_Prog	ram (* *)]					
😰 File Edit Tools Debug Options Win	dow <u>H</u> elp					- 8 ×
F2: HHE F3: HHE F4: 대한 🗶 Debug	F8: T <mark>0</mark> J F9:→ <b>≫ +</b> F9:-@	) i <b>t‡</b> tt	H			
🖙 🖬 🐰 🖻 👔 🚺 Simulation 🛛 🔪	🔟 🗏 📈 🕺	2 😼 🔬				
🔺 🏢 🔍 — 🚊 Spy Selection 🔪	• 🕰 😖 🗵					
ຜ່ຜ່ Debug FB F11		+	<b>-</b>	<b>.</b>		^
(* *)	+ +	+	+	± •	L	
[1] Input_01 Output_01					·	
╷╶┝╤═┛┇═┱╡╌╴╡╱╴╴	+ +	+	+	+	+	
	+ +	+	+	+	+	
						~
						>
×						
Simulation					NUM	

Double-click on Input\_01 to get the dialog box shown below.

Figure 3-79 Simulation Mode

🗗 DGE - [1: Resource1 - My_LD_Program (* *)]	
🗊 Eile Edit Tools Debug Options Window Help	- 8 ×
🗃 🖬 🧝 🕺 🚳 🔊 🏙	
A   Ⅲ   Q,	
(* *) Variable INPUT_01	
[1]     Input_01     Output_01     0     FALSE     TRUE     1	
Lock Unlock Cancel	~
× Name Resource Value Physical Scope Alias Comment Type Direction	
E Contraction of the second seco	
Ready	

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

🗗 DGE - [1: Resource1 - My_LD_Program	n (* *)]		
File Edit Tools Debug Options Window	Help		_ 8 ×
🖆 🖬 🚊 🍝 🔍 Debug			
A I III   Q - 🐱 Stop Simulation 🧕 🤉	E 22		
🚊 Spy Selection	······		-
(* *) &d d' Debug FB F11	+ +	+ +	+
[1] Input_01 Output_01			
╷╴┍╴ <mark>╤══┇┇╧═┑</mark> ┥┈╡╳┈╌╸╴		+ +	+
+ + +	+ +	+ +	+ 🗸
			>
×			
× Name Decourse Vielas Discipal	Carran Alian Carrant	Tura Disation	
Name Resource Value Physical	Scope Alias Comment	Type Direction	
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



🔀 ISaGRAF PRO - [My_Project (*	*) - Hardware architecture]			
Here Edit Insert Project Tools	ebug Options Window Help	_ 8 ×		
🗋 🖆 🔚 👗 🛍 💼 🍋 여 이 (수 수) 🞒 💥 🌾 🏹 ())) 🛠 🗞 (수) 👹 🥘				
A   2   Q	Connection - Properties			
Bo 명 🛍 🖿 🖫 🔕	Network Parameters			
	Name Value			
Config1 (* This is the	1 IPAddress 172.18.150.51 IP (			
1:Resource1 (* Resource )	(3)			
<				
	<			
•	OK Cancel Apply			
×				
Ready NUM //				

# 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, <u>right</u>-click on the Resource1 header and select Properties as shown below.

🔀 ISaGRAF PRO - [My_Project (* *) - Link arc	chitecture]
File Edit Insert Project Tools Debug Option	s Window Help
D 📽 🖬   X 🖻 🖻   🍳 🔿   🖨	🎏 🎘 🏹 🕪 🛠 🐻 😕 😽 🧕
] A   ⊴   Q, ── <b>」</b> ──────── Q,   ♠,	<b>a</b>
원 명 🖬 📼 🌐 🔕	
Link Architecture	Rename Resource       Delete     DEL
	Edit Description Properties
×[	
<u> </u>	Resource 1: Config1\Resource1 NUM

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

Resource Properties	×
General   Target / Code   Settings   Network   Extended	1
Trigger cycles Cycle Timing (ms): 1000	
Detect errors Nb of stored errors: 16	
C Cycle to cycle 💿 Real time	
Memory for Retain:	
Advanced	
OK Cancel Apply	

# 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.

My_Project				
<u>File E</u> dit <u>V</u> iew F <u>a</u> vorites <u>T</u> ools <u>H</u> elp				A
🚱 Back 🝷 🕥 🔹 🏂 🔎 Search 🞼 Fo	lders	,		
Address 🛅 C:\ISaGRAF PRO\Prj\My_Project				🖌 🄁 Go
Folders	<	Name 🔺	Size	Туре
🖃 🛅 ISaGRAF PRO	^	Config1		File Folder
🗉 🚞 Bin	_	📃 Compile.txt	1 KB	Text Documer
🛅 Grp	_	🔟 Conf.mtc	1 KB	MTC File
👝 Help		🔟 Conf.mws	1 KB	MWS File
🖃 🧰 Pri	_	DDCconf.ini	1 KB	Configuration
🔳 🧰 Demo		📴 dgb.ini	1 KB	Configuration
🖃 🧰 My_Project 🔪		📴 Dic. Ini	1 KB	Configuration
🗉 🧰 Config1		📴 DIP. INI	1 KB	Configuration
🗄 🦳 Project1		🔤 NetworkConf.rtc	2 KB	RTC File
B Copect2		🔤 NetworkConf.rws	2 KB	RWS File
🗄 🔂 Project3 Select and		🔤 NetworkConf.xtc	1 KB	XTC File
🗄 🔂 Project4 hit Delete		📾 NetworkConf.xws	1 KB	XWS File
E C Project5		📼 PRJlibrary.ldb	1 KB	Access.LockFi
C Simul		PRJlibrary.mdb	3,124 KB	Microsoft Acc
ing Tmp		🖬 project.gpm	1 KB	GPM File
🐨 🛄 Tpl	~			>

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)

ISaGRAF PRO - [Test101 (* *) - Link arcl	nitecture			
📴 Eile Edit Insert Project Tools Debug Opti	ons <u>W</u> indow <u>H</u> elp			
D 🚔 🔚   👗 🛍 💼   🌕 rei ( 🗢 🔿   🖨	B 💥 🎘 🔟 🖂 🕪 🗶 🗁 😸 🧕			
🔁 😤 🛍 🖿 🌐 🔬 🛝 100% 🔍				
1: RESOURCE1 (* Reso 🗖 📔	Resource Properties			
Parameters Variable Grou Programs Columbu: Functions Function blo Right-Click	Network       Extended       Security         General       Target / Code       Settings         Target:       VXW-TARGET_L       Help         VxWorks Target       Generate debug information          Image:       Code for simulation       Compiler Options         Image:       Code       Image: Code			
Check these two				
	OK Cancel Apply			

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

🔀 ISaGRAF PRO - [Test101 (* *) - Link architecture	•]
Pile Edit Insert Project Tools Debug Options Wind	low <u>H</u> el
🗋 🖆 🛃 👗 🛍 💼 🍙 🏊 🗇 🎒 💥 🔅	<b>\$</b> 🔏 1
응 명 🛍 🖿 🌐 🔬 🛝 100% 💽	
Right-Click	
	1
Z Maximize	
🕂 🔏 Cu <u>t</u> Resource Ctrl+X	
📑 📴 Copy Resource Ctrl+C	
<u>R</u> ename Resource	
<u>D</u> elete Resource DEL	
Show current step	
Clean <u>s</u> tored code	
Export resource	
Edit Description	
Properties	
Enter Password of RESOURCE1	

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

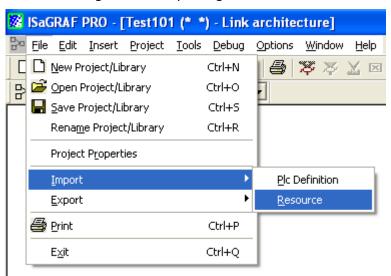


Figure 3-87 Importing the Resource

#### Figure 3-88 Importing the Resource

Import resource		×
2	Import in configuration: Config1 (*This is the Config 1*): ETCP: 172.18.150.51	
		>
1	<ul> <li>○ Import from <u>file</u></li> <li>○ <u>U</u>pload from target</li> </ul>	
	< <u>B</u> ack <u>N</u> ext > Cancel	

Figure 3-89 Warning Box

DUP	
?	Upload will be possible only if the connected target has the same target type as the selected configuration. Continue?
	<u>Yes</u> <u>N</u> o

Figure 3-90 Importing the Resource

Import resource	🔀
	Select the exchange file or resource to import: TEST101\CONFIG1\RESOURCE1 TEST101\CONFIG1\RESOURCE1 TEST101\CONFIG1\RESOURCE1 TEST101\CONFIG1\RESOURCE1
	Refresh
	Resource list upload started Resource list upload processing Resource list upload done
	< <u>B</u> ack <u>N</u> ext > Cancel

#### Figure 3-91 Importing the Resource

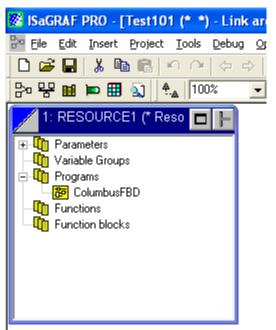
Import resource		×
	Uploading Resource 'RESOURCE1'	
	Resource upload done	
	Resource upload: 32% completed Resource upload: 65% completed Resource upload: 98% completed Resource upload: 100% completed	<
		2
	< <u>B</u> ack <u>N</u> ext > Cano	el

Figure 3-92 Importing the Resource

Import resource		
	Importing Resource 'RESOURCE1'	
	Creating resource properties. Creating code for resource. Disconnection from database. Deleteting old resource files. Copying importation result files to project directory. Deleting importation working files. Import resource OK	
		<u>&gt;</u>
	< <u>B</u> ack. <b>Finish</b>	Cancel

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.

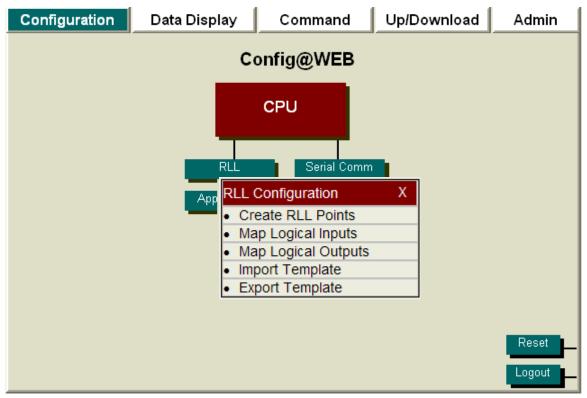


Figure 3-94 RLL Configuration

# 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.

RLL Configuration			
Туре	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	

Figure 3-95	<b>RLL Configuration</b>
-------------	--------------------------

#### Туре

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

	Input Configuration
KEL ANAIOO	Input Configuration
	Inpac ournigated of

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
З	RLL_ANALO Click on Heade	er to Change All	701	100	101
Change All X Value Set and/or change individual values					

#### 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

#### RLL Status Configuration

Point	Name		
0	RLL_STS 0		
1	RLL_STS 1		
2	RLL_STS 2		
3	RLL_STS 3		
	Cancel Submit		

#### 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

#### **RLL Accumulators Configuration**

Point	Name		
0	RLL_ACC 0		
1	RLL_ACC 1		
2	RLL_ACC 2		
3	RLL_ACC 3		
	Cancel Submit		

#### 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

RLL Analog Output Configuration					
Point	Name	C Min	C Max	EGU Min	EGU Max
0	RLL_AO 0	2000	7001	100	101
1	RLL_AO 1	2000	2001	100	101
2	RLL_AO 2	2000	2001	100	101
3	RLL_AO 3 Click on Head	ler to Change Al		100	101
Change All X Cancel Submit					
and/or change individual values					

#### 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

#### RLL Digitial Output Configuration

Point	Name		
O	RLL_DO 0		
1	RLL_DO 1		
2	RLL_DO 2		
3	RLL_DO 3		
	Cancel Submit		

#### 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

#### RLL SBO Configuration

Point	Name		
0	RLL_SBO 0		
1	RLL_SBO 1		
2	RLL_SBO 2		
3	RLL_SBO 3		
	Cancel Submit		

#### 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.

# 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 RLL Logical Inputs Mapping

Туре	Number	Map
Analog Inputs	8	MAP
Binary Inputs	8	MAP
Counters	8	MAP
Analog Outputs	8	MAP
Digital Outputs	8	MAP
SBO	8	MAP
		Back

#### Туре

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.

#### Мар

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	Device Name	Point Name	C Min 🥆	C Max 🤊	Source Points
0	DNPM_IED_1	IED_ANALOG 0	-2000	2000	DNPM IED 1
1	DNPM_IED_1	IED_ANALOG 1	-2000	2000	SPARE 🔷
2	DNPM_IED_1	IED_ANALOG 2	-2000	2000	Select All points IED_ANALOG 0
З	DNPM_IED_1	IED_ANALOG 3	-2000	2000	IED_ANALOG 0
4	DNPM_IED_1	IED_ANALOG 4	-2000	2000	IED_ANALOG 2
5	DNPM_IED_1	IED_ANALOG 5	-2000	2000	IED_ANALOG 3
6	DNPM_IED_1	IED_ANALOG 6	-2000	2000	IED_ANALOG 4 IED_ANALOG 5
7	DNPM_IED_1	IED_ANALOG 7	-2000	2000	IED_ANALOG 5
					ED_ANALOG 7
				Cli	ck on Header to Change All
Change All X Value Set and/or change individual values					
Cancel Submit					

#### 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.

### 3.12.2.2 Binary Inputs

Figure 3-104 RLL Status Input Point Mapping

RLL Status Input Point Mapping

Point	Device Name	Point Name	Invert 🔊	Source Points
0	DNPM_IED_1	COMM_STS	OYes ⊙No	
1	DNPM_IED_1	IED_STS 0	OYes ONO	
2	DNPM_IED_1	IED_STS 1	OYes ONO	Click on any Header that has a
3	DNPM_IED_1	IED_STS 2	OYes ⊙No	hand 💶 to Change All
4	DNPM_IED_1	IED_STS 3		Change All X
5			OYes ⊙No	Value 🖲 Yes 🔿 No Set
6	DNPM_IED_1	IED_STS 4	OYes ⊙No	and/or change individual
	DNPM_IED_1	IED_STS 5	OYes ⊙No	
7	DNPM_IED_1	IED_STS 6	OYes ⊙No	IED_STS 4
				IED_STS 5
				IED_STS 6
				IED_STS 7
				IED_STS 8
				IED_STS 9
				IED_STS 10
				IED_STS 11
				IED_STS 12
				IED_STS 13
				TED CTC 14
				Cancel Submi

#### 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.

### 3.12.2.3 Counters

Figure 3-105 RLL Accumulator I	Point Mapping
RLL Accumulator Doint M	anning

Point	Device Name	Point Name	Source Points				
0	DNPM_IED_1	IED_ACC_ 0	DNPM IED 1				
1	DNPM_IED_1	IED_ACC_ 1	SPARE				
2	DNPM_IED_1	IED_ACC_2	Select All points				
З	DNPM_IED_1	IED_ACC_ 3	IED_ACC_0 IED_ACC_1				
4	DNPM_IED_1	IED_ACC_ 4	IED_ACC_ 2				
5	DNPM_IED_1	IED_ACC_ 5	IED_ACC_ 3				
6	DNPM_IED_1	IED_ACC_ 6	IED_ACC_4				
7			IED_ACC_5				
	DNPM_IED_1	IED_ACC_7	IED_ACC_6				
			IED_ACC_ 7				
			IED_ACC_ 8				
			IED_ACC_ 9				
			IED_ACC_ 10				
			IED_ACC_ 11				
			IED_ACC_ 12				
			ΤΕΝ ΔCC 13				
			Cancel Submit				

#### 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.

# 3.12.2.4 Analog Outputs

# Figure 3-106 RLL Analog Output Point Mapping

Point	Device Name	Point Name	C Min 🔊	C Max 🔊	Source Points			
0	DNPM_IED_1	IED_AO_ 0	-2000	2000	DNPM IED 1	<b>V</b>		
1	DNPM_IED_1	IED_AO_1	-2000	2000	SPARE	<b>^</b>		
2	DNPM_IED_1	IED_AO_2	-2000	2000	Select All points			
3	DNPM_IED_1	IED_AO_3	-2000	2000	IED_AO_ 0 IED_AO_ 1			
4	DNPM_IED_1	IED_AO_4	-2000	2000	IED_AO_ 2			
5	DNPM_IED_1	IED_AO_5	-2000	2000	IED_AO_ 3			
6	DNPM_IED_1	IED_AO_6	-2000	2000	IED_AO_ 4			
_					IED_AO_ 5			
7	DNPM_IED_1	IED_AO_	-2000	2000	IED_AO_ 6			
		<			IED_AO_ 7			
	Click on Head	er to Change All			IED_AO_ 8			
	Change All	Х			IED_AO_ 9			
					IED_AO_ 10			
	Value	Set			IED_AO_ 11			
	and/or change	IED_AO_ 12						
and/or change individual values IED_AO_12						-		
	Cancel Submit							

#### 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.

### 3.12.2.5 Digital Outputs

# Figure 3-107 RLL Digital Output Point Mapping

	RLL Digital Output Point Mapping						
Point	Device Name	Point Name	Source Points				
0	MB_IED_1	IED_DO_ 0	MB IED 1				
1	MB_IED_1	IED_DO_1	SPARE				
2	MB_IED_1	IED_DO_ 2	Select All points				
3	 MB_IED_1	IED_DO_ 3	IED_DO_ 0 IED_DO_ 1				
4	MB_IED_1	IED_DO_ 4	IED_DO_ 2				
5	MB_IED_1	IED_DO_5	IED_DO_ 3				
6	 MB_IED_1	IED_DO_6	IED_DO_ 4				
			IED_DO_5				
7	MB_IED_1	IED_DO_7	IED_DO_6				
			IED_DO_ 7				
			IED_DO_ 8				
			IED_DO_ 9				
			IED_DO_10				
			IED_DO_ 11				
			IED_DO_ 12				
			IED DO 13				
			Cancel Submit				

#### 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.

# 3.12.2.6 SBO

#### Figure 3-108 RLL SBO Point Mapping

RLL SBO Point Mapping							
Point	Device Name	Point Name	State 🄊	Source Points			
0	DNPM_IED_1	IED_BO 0	⊙ Trip O Close	DNPM IED 1			
1	DNPM_IED_1	IED_BO 1	⊙ Trip O Close	SPARE			
2	DNPM_IED_1	IED_BO 2		Select All points			
3	DNPM IED 1	IED_BO 3		IED_BO 0 IED_BO 1			
4	DNPM_IED_1	IED_BO 4		IED_BO 2			
5	DNPM_IED_1	IED_BO 5	⊙ Trip O Close	IED_BO 3			
				IED_BO 4			
6	DNPM_IED_1		⊙ Trip O Close	IED_BO 5			
7	DNPM_IED_1		💿 Trip 🔘 Close	IED_BO 6			
	,			IED_BO 7			
	Click on Handar to C			IED_BO 8			
	Click on Header to C			IED_BO 9			
	Change All >			IED_BO 10			
	Value Se	t		IED_BO 11			
	and/or change indivi	dual values		IED_BO 12			
	genau						
				Cancel Submit			

#### 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.

# 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

#### RLL Logical Outputs Mapping

Туре	Number	Мар
Analog Inputs	8	MAP
Binary Inputs	8	MAP
Counters	8	MAP
Analog Outputs	8	MAP
Digital Outputs	8	MAP
SBO	8	MAP
		Back

#### Туре

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.

#### Мар

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

RLL Analog	Input	Point	Mappi	ng
------------	-------	-------	-------	----

Point	Device Name	Point Name	C Min 🤏	C Max 🔊	Source Points		
0	DNPM_IED_1	IED_ANALOG 0	-2000	2000	DNPM IED 1		
1	DNPM_IED_1	IED_ANALOG 1	-2000	2000	SPARE 🔶		
2	DNPM_IED_1	IED_ANALOG 2	-2000	2000	Select All points IED ANALOG 0		
3	DNPM_IED_1	IED_ANALOG 3	-2000	2000	IED_ANALOG U		
4	DNPM_IED_1	IED_ANALOG 4	-2000	2000	IED_ANALOG 2		
5	DNPM_IED_1	IED_ANALOG 5	-2000	2000	IED_ANALOG 3		
6	DNPM_IED_1	IED_ANALOG 6	-2000	2000	IED_ANALOG 4		
7	DNPM_IED_1	IED_ANALOG 7	-2000	2000	IED_ANALOG 5 IED_ANALOG 6		
					ED_ANALOG 7		
				Cli	ck on Header to Change All		
Change All X Value Set and/or change individual values							
	Cancel Submit						

#### 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.

### 3.12.3.2 Binary Inputs

Figure	3-111	RLL	Status	Input	Point	Mapping	a
inguic	2 1 1 1		Julus	mput	1 01110	mapping	1

RLL Status Input Point Mapping

Point	Device Name	Point Name	Invert 🔊	Source Points
0	DNPM_IED_1	COMM_STS	○Yes ⊙No	DNPM_IED_1
1	DNPM_IED_1	IED_STS 0	OYes ⊙No	4
2	DNPM_IED_1	IED_STS 1	OYes ⊙No	Click on any Header that has a
3	DNPM_IED_1	IED_STS 2	OYes ⊙No	hand to Change All
4	DNPM_IED_1	IED_STS 3	OYes ⊙No	Value • Yes • No Set
5	DNPM_IED_1	IED_STS 4	OYes ⊙No	and/or change individual
6	DNPM_IED_1	IED_STS 5	OYes ⊙No	values
7	DNPM_IED_1	IED_STS 6	OYes ⊙No	IED_STS 4
				IED_STS 5
				IED_STS 6
				IED_STS 7
				IED_STS 8
				IED_STS 9
				IED_STS 10
				IED_STS 11
				IED_STS 12
				IED_STS 13
				TED STS 14
				V 1ED_31314
				Cancel Submit

#### 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.

### 3.12.3.3 Counters

Figure 3-112 RLL Accumu	llator Point Mapping
RLL Accumulator P	oint Mapping

Point	Device Name	Point Name	Source Points			
0	DNPM_IED_1	IED_ACC_ 0	DNPM IED 1			
1	DNPM_IED_1	IED_ACC_1	SPARE			
2	DNPM_IED_1	IED_ACC_2	Select All points			
3	DNPM_IED_1	IED_ACC_ 3	IED_ACC_ 0 IED_ACC_ 1			
4	DNPM_IED_1	IED_ACC_4	IED_ACC_ 2			
5	DNPM_IED_1	IED_ACC_5	IED_ACC_ 3			
6	DNPM_IED_1	IED_ACC_6	IED_ACC_ 4			
			IED_ACC_ 5			
7	DNPM_IED_1	IED_ACC_7	IED_ACC_ 6			
			IED_ACC_ 7			
			IED_ACC_ 8			
			IED_ACC_ 9			
			IED_ACC_ 10			
			IED_ACC_ 11			
			IED_ACC_ 12			
			TED ΔCC 13			
			Cancel Submit			

#### 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.

### 3.12.3.4 Analog Outputs

#### Figure 3-113 RLL Analog Output Point Mapping

RLL Analog Output Point Mapping							
Point	Device Name	Point Name	C Min 🥆	C Max 🔊		Source Points	
0	DNPM_IED_1	IED_AO_ 0	-2000	2000	DNP	M IED 1	<b>~</b>
1	DNPM_IED_1	IED_AO_1	-2000	2000		SPARE	<u></u>
2	DNPM_IED_1	IED_AO_2	-2000	2000		Select All points	
3	DNPM_IED_1	IED_AO_ 3	-2000	2000		IED_AO_ 0 IED AO 1	
4	DNPM_IED_1	IED_AO_4	-2000	2000		IED_AO_ 2	
5	DNPM IED 1	IED_AO_5	-2000	2000		IED_AO_ 3	
6	DNPM_IED_1	IED_AO_6	-2000	2000		IED_AO_ 4	
7	DNPM_IED_1	IED_AO	-2000	2000		IED_AO_5	
			-2000	2000		IED_AO_ 6	
	Click on Hoad	ler to Change All	<u>`</u>			IED_AO_ 7	
		er to change All				IED_AO_ 8	
	Change All	X				IED_AO_ 9	
	Value	Cot				IED_AO_ 10	
Value Set IED_AO_11							
and/or change individual values J IED_AO_ 12							
	IED 40 13						
	Cancel Submit						

#### 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.

# 3.12.3.5 Digital Outputs

# Figure 3-114 RLL Digital Output Point Mapping

RLL Digital Output Point Mapping								
Point	Device Name	Point Name		Source Points				
0	MB_IED_1	IED_DO_ 0		MB IED 1	<b>~</b>			
1	MB_IED_1	IED_DO_1		SPARE	<u>^</u>			
2	MB_IED_1	IED_DO_ 2		Select All points				
3	MB_IED_1	IED_DO_ 3		IED_DO_ 0				
			-	IED_DO_ 1				
4	MB_IED_1	IED_DO_ 4		IED_DO_ 2				
5	MB_IED_1	IED_DO_ 5		IED_DO_ 3				
6	MB_IED_1	IED_DO_6		IED_DO_ 4	≡			
			-	IED_DO_ 5				
7	MB_IED_1	IED_DO_ 7		IED_DO_ 6				
				IED_DO_ 7				
				IED_DO_ 8				
				IED_DO_ 9				
				IED_DO_ 10				
				IED_DO_ 11				
				IED_DO_ 12				
				TED DO 13	×			
				Cancel	Submit			

#### 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.

### 3.12.3.6 SBO

#### Figure 3-115 RLL SBO Point Mapping

RLL SBO Point Mapping											
Point	Device Name	Point Name	Point Name 🛛 State 🍽		Source Points						
0	DNPM_IED_1	IED_BO 0	💿 Trip 🔇	Close	DNPM IED 1						
1	DNPM_IED_1	IED_BO1	💿 Trip 🔇	Close	SPARE	<u> </u>					
2	DNPM_IED_1	IED_BO 2			Select All points						
3	DNPM_IED_1	IED BO 3			IED_BO 0						
4	DNPM_IED_1	IED_BO 4			IED_BO 1 IED_BO 2						
5	DNPM_IED_1	IED_BO 5			IED_BO 3						
					IED_BO 4						
6	DNPM_IED_1	IED_B	⊙ Trip (		IED_BO 5						
7	DNPM_IED_1	IET	💽 Trip 🤇		IED_BO 6						
	,	IED_BO 7									
	Click on Handar to C	IED_BO 8									
	Click on Header to C	IED_BO 9									
	Change All >	IED_BO 10									
	Value Se	IED_BO 11									
	and/or change indivi	IED_BO 12									
	J. J. J.	TED BO 13	<b>~</b>								
					Cancel	Submit					

#### 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.

# 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.

Relay Ladder Logic Load Template		
Load Template	X	
Load Template	×	
	Get	

#### 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.

	, , ,			
Save Template	X			
Replace Existing	×			
	(0R)			
Create New				
	Save			

#### Relay Ladder Logic Save Template

# **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.

RLL Data Display			
Туре	Number	View	
Analogs Inputs	4	View	
Binary Inputs	4	View	
Counters	4	View	
Analog Outputs	4	View	
Digital Outputs	4	View	
SBO	4		
		Back	

Figure 3-116 RLL Data Display

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
1 2 3	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

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	•
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
			Ba	:k

#### Figure 3-118 RLL Status Inputs Display

#### 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

RLL Accumulator Inputs Display			
	Page1 of 1 Go To G	ο	
Point	Point Name	Count	
1	RLL_ACC 0	0	
2	RLL_ACC 1	0	
2 3	RLL_ACC 2	0	
4	RLL_ACC 3	0	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
-	-	-	
		Back	

Figure 3-119 RLL Accumulator Inputs Display

#### 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.

3-87

# 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	•
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
			Ba	ck 🔵

#### 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.

RLL Command			
Туре	Number	Command	
Analogs Inputs	4		
Binary Inputs	4		
Counters	4		
Analog Outputs	4	Command	
Digital Outputs	4	Command	
SBO	4	Command	
Back			

Figure 3-122 RLL Command

# 3.14.1 Analog Outputs

#### Figure 3-123 RLL Analog Outputs Command

RLL Analog Outputs Command				
	Page 1 of 1	Go To G	0	
Point	Name	Range	Value	Operation
O	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**

KEE Digital Outputs Command								
Page 1 of 1 GoTo Go								
Point	Name	Poir	nt Operatio	ns				
0	RLL_DO 0	💿 Open	🔘 Close	Execute				
1	RLL_DO 1	🔘 Open	🔘 Close	Execute				
2	RLL_DO 2	🔘 Open	🔘 Close	Execute				
З	RLL_DO 3	🔘 Open	🔘 Close	Execute				
Open or	n RLL_DO 0 : Successful			Back				

#### 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

RLL SBO Command.							
Pnt	Name	P	oint Operat	ions			
0	RLL_SBO 0	💿 Trip	🔘 Close	Execute			
1	RLL_SBO 1	🔿 Trip	🔘 Close	Execute			
2	RLL_SBO 2	🔿 Trip	🔘 Close	Execute			
3	RLL_SBO 3	🔿 Trip	🔘 Close	Execute			
Trin on I	RULI SBOID : No response fro	om controlli	ina task	Back			

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.

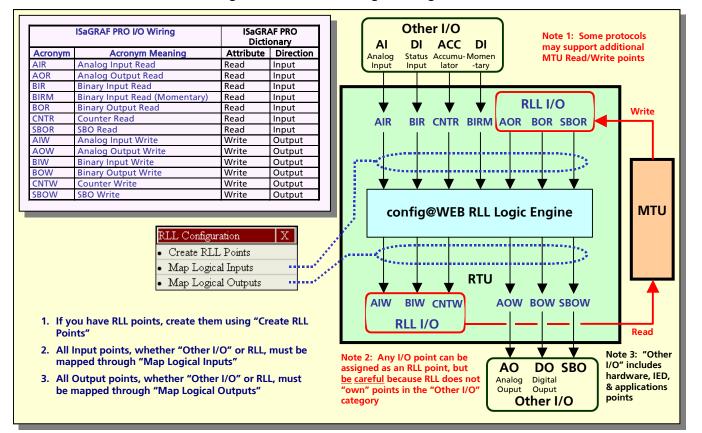
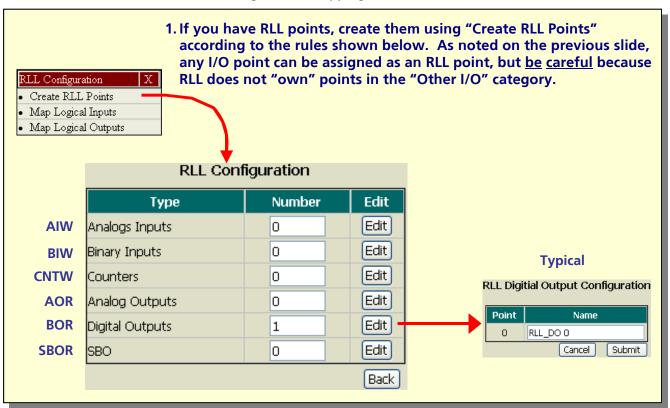
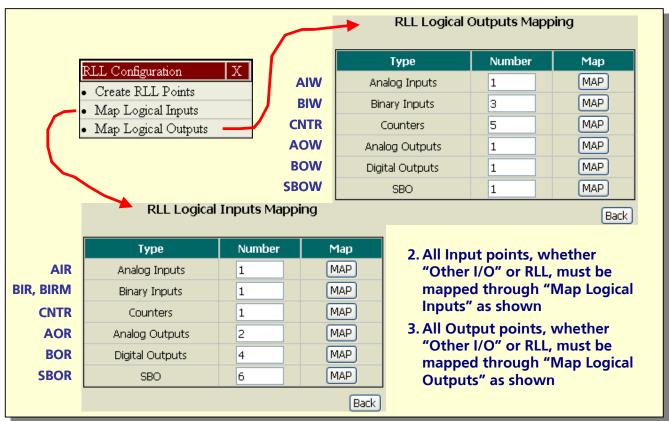


Figure 4-1 MTU/RTU Programming Model

#### 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.

DGE - [1: Resource1 - BIW_RLL_Status (* *)]	
File Edit Tools Debug Options Window Help	- U
😂 🖬   X 🖻 💼   🍳 🗠 🎒 🞒 🛍 🎽 💥 🕺 🥵 🛐 🖉	
▲   Ⅲ   Q,	
▶ 🖙 10: ⊣ 14 - + : 😭 ► 14 - I () - H 🖻	
AIR_Input_01        .     .       .     .       .     .       .     .       .     .       .     .       .     .       .     .       .     .       .     .       .     .	*
	•
Ready	

#### Variables 4.3.2



BIW_RLL_Status											
Name	Alias	Туре	0	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

#### Wiring 4.3.3

Figure 4-7 BIW\_RLL\_Status Program Wiring for Analog Inputs



#### Figure 4-8 BIW\_RLL\_Status Program Wiring for RLL Status Output

😑 👜 6: SAGE: BIW: BIW(\* \*)

+...

🚴 %QX6.0=BIW\_RLL\_Status\_Out@BIW\_RLL\_Status

# 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	
------------	--------------------------------------	--

RLL Cor	nfiguration X		RLL Conf	iguration	
• Create	e RLL Points 👘 🔸		Туре	Number	Edit
• Map I	ogical Inputs.		Analogs Inputs	0	Edit
• Map I	ogical Outputs.		Binary Inputs	1	Edit
DLI	Status Configuratio		Counters	0	Edit
KLL	status coringuratio		Analog Outputs	0	Edit
Point	Name		Digital Outputs	0	Edit
0	RLL_STS 0		SBO	0	Edit
	Cancel Sub	omit			Back

R.

		•	
LL Configuration X	RLL Logical	Outputs Mapp	bing
Create RLL Points	-		
Map Logical Inputs	Туре	Number	Мар
Map Logical Outputs	Analog Inputs	O	MAP
	Binary Inputs	1	MAP
	Counters	0	MAP
	Analog Outputs	0	MAP
	Digital Outputs	0	MAP
	SBO	0	MAP
			Back

#### Figure 4-10 RTU Configuration – Map Logical Outputs

**RLL Status Input Point Mapping** 

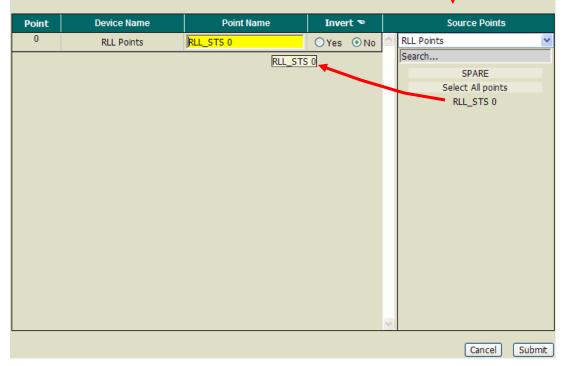


Figure 4-11 Shows how to map the two analog input points from the hardware analog points.

RIT Co	onfiguration	X		RLL LO	ogical	Input	s Марр	bing	
	e RLL Points			Туре		Nur	nber	Мар	
Мар.	Logical Inputs		A	nalog Inputs	:	2		MAP	
• Map ]	Logical Outputs		В	inary Inputs		0		MAP	
				Counters		0		MAP	
			An	alog Output	s	0		MAP	
			Di	gital Outputs	5	0		MAP	
				SBO		0		MAP	
								Bac	-10
							/		
		R	LL Analog	Input Point N	4apping		•		
Point	Device Name	Point N	lame	C Min 🔊	C Max	<b>~</b>		Source Points	
Point 0	Device Name Hardware Analogs	Point N	lame	C Min マ -2000		<b>~</b>	Hardware	e Analogs	~
			ANALOG	-2000 1 -2000			Hardware	e Analogs SPARE	~
0	Hardware Analogs	ANALOG 1		-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points	¥
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2	v
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6 ANALOG 7	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6 ANALOG 7	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6 ANALOG 7	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6 ANALOG 7	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6 ANALOG 7	~
0	Hardware Analogs	ANALOG 1	ANALOG	-2000 1 -2000		2000	Hardware	e Analogs SPARE Select All points ANALOG 1 ANALOG 2 ANALOG 3 ANALOG 4 ANALOG 5 ANALOG 6 ANALOG 7	~

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.

#### 4-9

# Figure 4-12 Monitoring the Analogs

Analog Inputs (AI) Display
----------------------------

4

	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

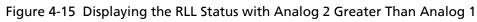
However, the Analog Display shows only the inputs to the program. Display the pseudo status point as shown below. Figure 4-14 shows the results if Analog 2 is equal to, or less than, Analog 1. Figure 4-15 shows the results if Analog 2 is greater than Analog 1.

RLL Data Display							
Туре	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 4-13 RLL Data Display

	RLL Status Inputs Display								
	Page1 of 1 Go	To 🗾 GO							
Point	Point Name	Point Status	Point State	•					
1	RLL_STS 0		OPEN	٠					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	· _	-					
			Ba	ck 🛛					

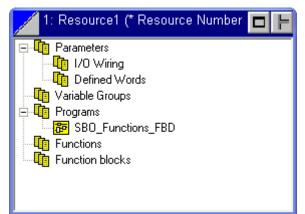
#### Figure 4-14 Displaying the RLL Status with Analog 2 Less Than Analog 1



	RLL Status Inputs Display								
	Page1 of 1 Go	To 🗾 GO							
Point	Point Name	Point Status	Point State	•					
1	RLL_STS 0		CLOSED	•					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-					
			Ba	:k					

# 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.

#### B DGE - [1: Resource1 - SBO\_Functions\_FBD (\* \*)] 🚰 File Edit Tools Debug Options Window Help a 法 陥 💼 📭 🗠 (番) 🖻 🔒 🚧 🔟 🔄 🔬 🔍 🧟 # | Q 🗨 🖂 🖂 A - h k e I: H → : 🔐 🕨 +ト ま () -નિ ٠ RS 0R SET SBOR\_SBO\_Input Q1 RESE BIW\_STS\_Output SBOR\_SBO\_Reset All three inputs and the output are pseudo points. Because SBOs (even pseudo SBOs) are momentary, RS latches the input from BOR\_DO\_Control SBOR\_SBO\_Input. SBOR\_SBO\_Reset resets the latch. BOR\_DO\_Control allows an independent input (non-momentary) to also flip the output. Output BIW\_STS\_Output is a pseudo status. • NUM Ready

#### Figure 4-17 SBO\_Functions\_FBD Program

# 4.4.2 Variables



SBO_Functions_FB	D									
Name	Alias Type	0	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

🖻 – 🛄 6: SAGE: BIW: BIW(\* \*) -

🗄 🗄 %QX6.0=BIW\_STS\_Output@SBO\_Functions\_FBD

🚊 👜 7: SAGE: BOR: BOR(\* \*)

- 🚵 %IX7.0=BOR\_DO\_Control@SBO\_Functions\_FBD

🖃 👜 11: SAGE: SBOR: SBOR(\* \*)

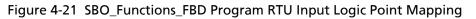
÷

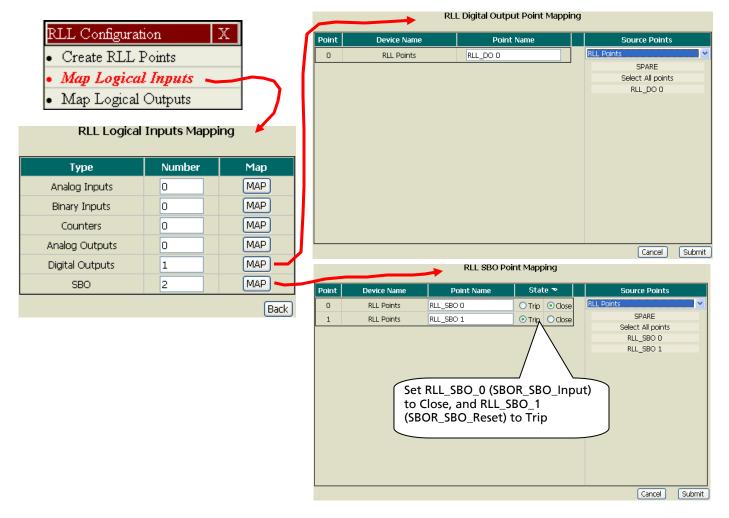
- 🚴 %IX11.0=SBOR\_SBO\_Input@SBO\_Functions\_FBD
- 🚴 %IX11.1=SBOR\_SBO\_Reset@SBO\_Functions\_FBD

# 4.4.4 RTU Mapping

### Figure 4-20 SBO\_Functions\_FBD Program RTU RLL Point Mapping

RLL Configuration	X			RLL	Status Configuration
<ul> <li>Create RLL Point</li> <li>Map Logical Inputs</li> </ul>		[		Point	Name
<ul> <li>Map Logical Inputs</li> <li>Map Logical Output</li> </ul>			L	0	RLL_STS 0
	¥				Cancel Submit
RLL Con	figuration	F	RLL Digi	itial Output Configuration	
Туре	Number	Edit	1	A	
Analogs Inputs	O	Edit		Point	Name
Binary Inputs	1	Edit		0	RLL_DO 0
Counters	0	Edit			Cancel Submit
Analog Outputs	0	Edit	/	RLL	. SBO Configuration
Digital Outputs	1	Edit			
SBO	2	Edit		Point	Name
				0	RLL_SBO 0
		Back		1	RLL_SBO 1
					Cancel Submit





#### Figure 4-22 SBO\_Functions\_FBD Program RTU Output Logic Point Mapping

RLL Configuration	X				RLL Status Input Po	int Mapping		
			Point	Device Name	Point Name	Invert 🔊	Source Points	
<ul> <li>Create RLL Point</li> </ul>	ts		0	RLL Points	RLL_STS 0	OYes ⊙No	RLL Points	~
<ul> <li>Map Logical Inpu</li> </ul>	ıts				RLL_STS	50	Search	
Map Logical Or	utputs 🗕						Select All points RLL_STS 0	
RLL Logical (	Outputs Mapp	ing					_	
Туре	Number	Мар						
Analog Inputs	0	MAP						
Binary Inputs	1	MAP -						
Counters	0	MAP						
Analog Outputs	0	MAP						
Digital Outputs	0	MAP						
SBO	0	MAP				N		
		Back					Cancel	Submit

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

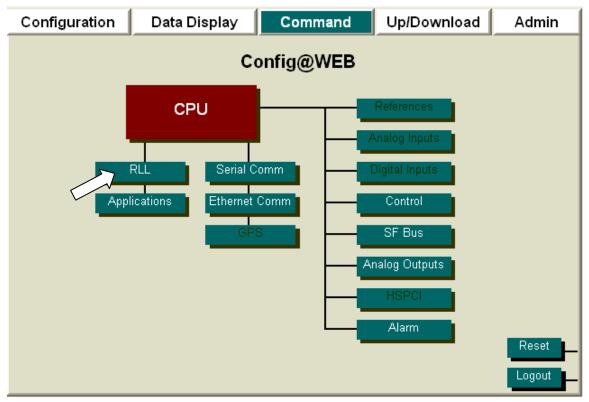


Figure 4-23 Commanding SBOR\_SBO\_Input

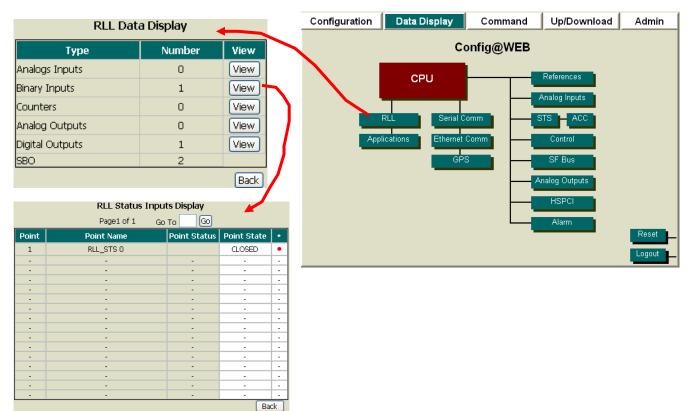
Figure 4-24 Commanding SBOR\_SBO\_Input

RLL Command						
Туре	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

RLL SBO Command.							
Pnt	Name	Pa	oint Operat	tions			
0	RLL_SBO 0	🔘 Trip	💿 Close	Execute			
1	RLL_SBO 1	🔘 Trip	🔘 Close	Execute	<b>\</b>		
Trip on F	RLL_SBO 1 : Successful			Back			



#### 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

RLL SBO Command.							
Pnt	Name	Point Operations					
0	RLL_SBO 0	O Trip O Close Execute					
1	RLL_SBO 1	💿 Trip 🔿 Close Execute					
Trip on F	RLL_SBO 1 : Successful	Back					

### Figure 4-28 Resulting RLL Status Inputs Display

RLL Data	Display			Configuration Data Display Command Up/Download Admin
Туре	Number	View		Config@WEB
Analogs Inputs	0	View		CPI I References
Binary Inputs	1	View		
Iounters	0	View		Analog Inputs
Analog Outputs	0	View		RLL Serial Comm
Digital Outputs	1	View		Applications Ethernet Comm Control
5BO	2			GPS SF Bus
		Back		Analog Outputs
RLL Status	Inputs Display			HSPCI
Page1 of 1	Go To 📃 GO	_		Alarm
Point Point Name	Point Status	Point State	•	Reset
1 RLL_STS 0		OPEN	•	
	-	-	-	Logout
	-	-		
	-	-	- I	
	-	-	÷	
		-	-	
	-	-	-	
- ·	-	-	-	

Back

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.

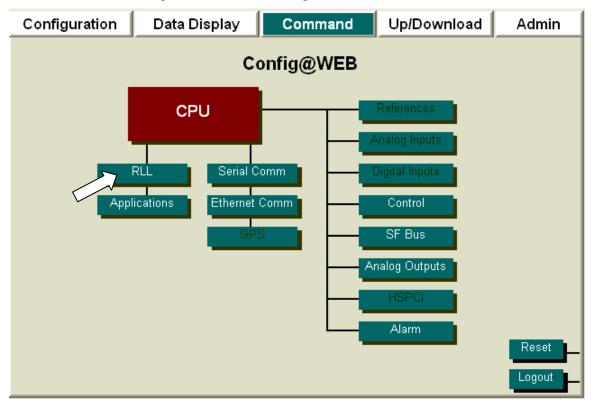


Figure 4-29 Commanding the BOR\_DO\_Control

Figure 4-30 Commanding the BOR\_DO\_Control

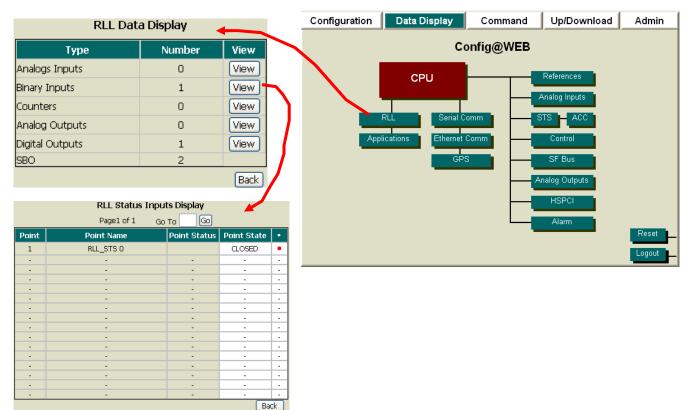
RLL Command						
Туре	Number	Command				
Analogs Inputs	0					
Binary Inputs	1					
Counters	0					
Analog Outputs	0	Command				
Digital Outputs	1	Command				
SBO	2	Command				
		Back				

To execute the BOR\_DO\_Control command properly, you must Close the point, as shown in Figure 4-31.

Figure 4-31 Commanding the BOR_DO_Control							
RLL Digital Outputs Command							
	Page 1 of 1	GoTo	Go				
Point	Name	Point Operations					
0	RLL_DO 0	🔘 Open	💿 Close	Execute			

Figure 4-31 Commanding the BOR DO Control

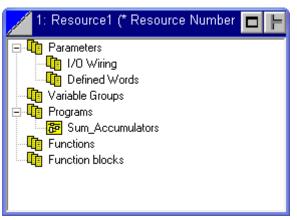
Unlike an SBO momentary input that has to be latched, the DO input that closes the status point will stay closed. If you need the status point to open again, you must open the RLL\_DO\_0 point.



#### 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.

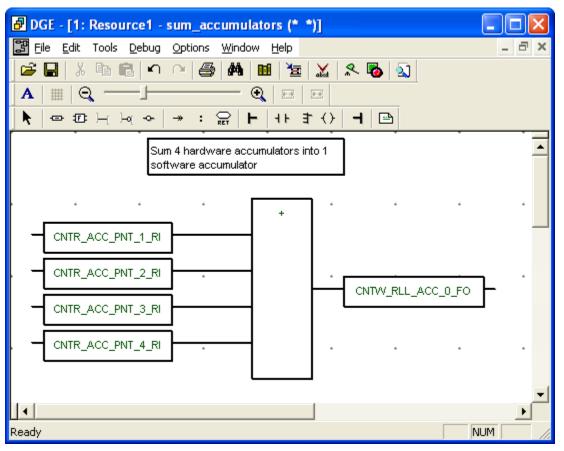


Figure 4-34 Sum\_accumulators Program

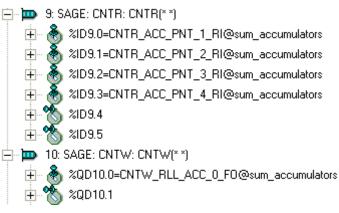
# 4.5.2 Variables



sum_accumulators												
Name	Alias	Туре	0	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_F0		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

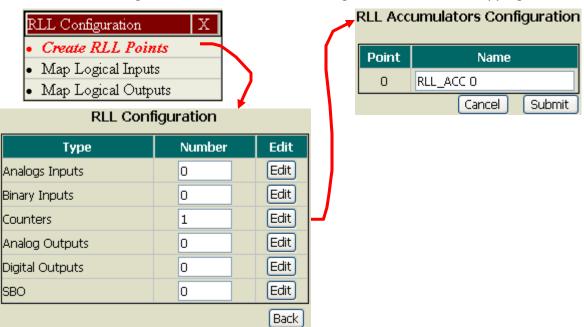
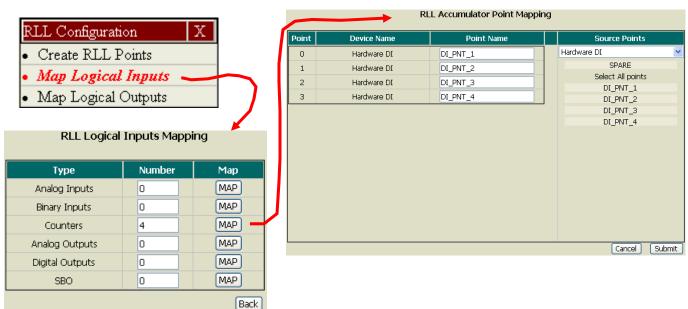


Figure 4-37 Sum\_accumulators Program RTU RLL Point Mapping



#### Figure 4-38 Sum\_accumulators Program RTU Input Logic Point Mapping

#### Figure 4-39 Sum\_accumulators Program RTU Output Logic Point Mapping

					RLL Accumulator Point Mapp	ing
RLL Configura	ation	X	Point	Device Name	Point Name	Source Points
• Create RLL	Points		0	RLL Points	RLL_ACC 0	RLL Points
Map Logical	1 Topute					SPARE Select All points
	-					RLL_ACC 0
Map Logica	ul Outputs					
Dillaria	Outroute Mana					
RLL LOGICAI	Outputs Mapp	oing 📕				
Туре	Number	Мар				
Analog Inputs	0	MAP				
Binary Inputs	0	MAP				
Counters	1	MAP				
nalog Outputs	0	MAP				
igital Outputs	0	MAP				Cancel Subn
SBO	0	MAP				
		Back				

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

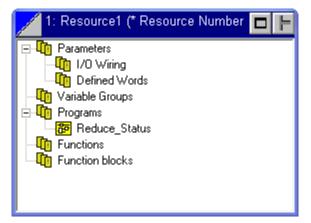
RLL Dat	a Display 🛛 ┥		Configuration Data Display Command Up/Download Admin
Туре	Number	View	Config@WEB
Analogs Inputs	0	View	CPU
Binary Inputs	0	View	
Counters	1	View	Analog Inputs
Analog Outputs	0	View	RLL Serial Comm
Digital Outputs	0	View	Applications Ethernet Comm Control
SBO	0		GPS SF Bus
		Back	Analog Outputs
RLL Accumulate	or Inputs Display		HSPCI
	ю то 👘 😡		Alarm
Point Point Nam		Count	Reset
1 RLL_ACC (		13	Logout
· ·			
· ·			
		-	
		-	
· ·			
· ·			
· · ·		-	
· · ·		-	
· ·		-	
· · ·			
		Back	

#### 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.

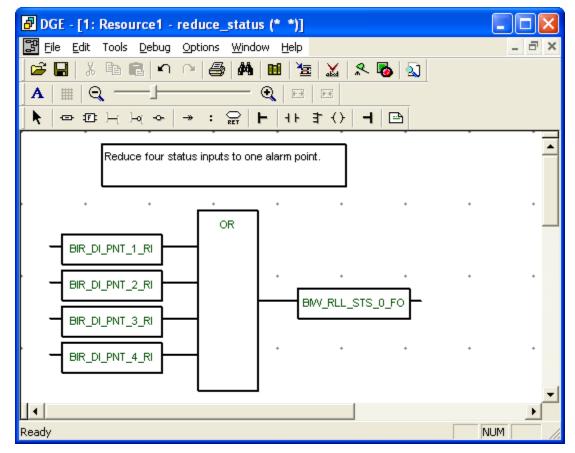


Figure 4-42 Reduce\_Status Program

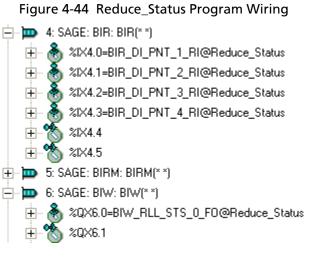
# 4.6.2 Variables



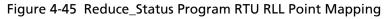
reduce_status												
Name	Alias	Туре	0	Ini	Di	Group	Attribute	Scope	Direction	Retain	Wiring	Address
BIR_DI_PNT_1_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.0	
BIR_DI_PNT_2_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.1	
BIR_DI_PNT_3_RI		BOOL				None	Read	reduce_status	Input	No	%IX4.2	
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	

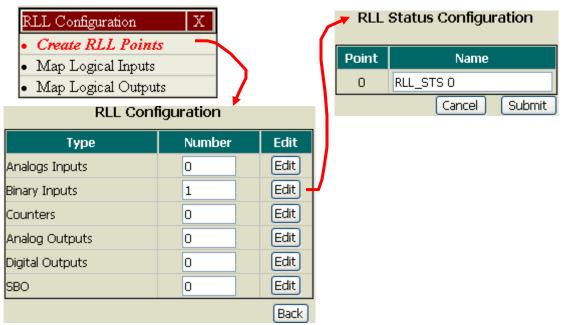
4-24

# 4.6.3 Wiring



# 4.6.4 RTU Mapping



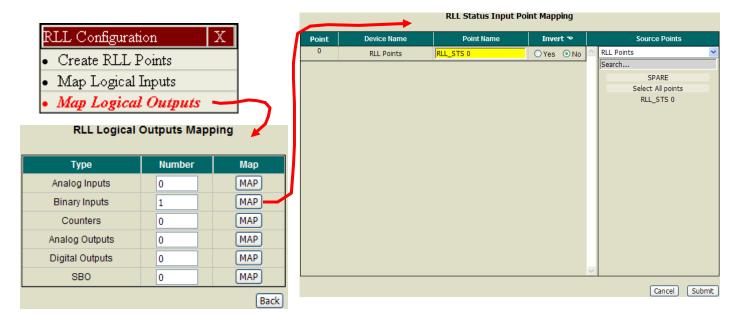


RLL Status Input Point Manning

#### Figure 4-46 Reduce\_Status Program RTU Input Logic Point Mapping

	· -	-			RLL Status Input	Point Mapping		
LL Configurati	ion 💚	C	Point	Device Name	Point Name	Invert 🔊	Source Points	
Create RLL I	Points		0	Hardware DI	DI_PNT_1	OYes ⊙No	Ardware DI	
			1	Hardware DI	DI_PNT_2	OYes ⊙No	Search	
<b>Map Logical</b> Map Logical	l Inputs 🚤		2	Hardware DI	DI_PNT_3	OYes ⊙No	SPARE	
Mon Logical	Outrouta		3	Hardware DI	DI_PNT_4	OYes ⊙No	Select All points DI PNT 1	
tviap Logical	Outputs						DI_PNT_1 DI_PNT_2	
							DI_PNT_3	
RLL Logica	il Inputs Mappii	ng 📕					DI_PNT_4	
, in the second s		Ŭ					DI_PNT_5	
							DI_PNT_6	
Type	Number	Map					DI_PNT_7	
Analog Inputs	0	MAP					DI_PNT_8 DI_PNT_9	
Analog Inputs	U	MAP					DI_PNT_10	
Binary Inputs	4	[MAP]					DI_PNT_11	
Countara	0	MAP					DI_PNT_12	
Counters	U						DI_PNT_13	
Analog Outputs	0	MAP					DI_PNT_14	
District October							DI_PNT_15	
Digital Outputs	0	MAP					DI_PNT_16	
SBO	0	MAP						
							Cancel	Sub
		Back						

### 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.

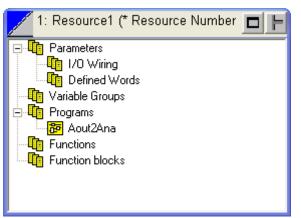
## 4.6.5 RTU Display

F	RLL Data Display			Configuratio	on Data Dis	splay	Comma	and	Up/Download	Admin
Туре	Numb	er Viev				Co	nfig@W	EB		
Analogs Inputs	0	View						_		
Binary Inputs	1	View			CPL	, t			References	
Counters	0	View	51				F	A	nalog Inputs	
Analog Outputs	0	View	=		RLL	Serial Co	mm 📕	ST	IS 🗕 ACC	
Digital Outputs	0	View	<u> </u>		Applications	Ethernet C	omm		Control	
SBO	0	(101	-			GPS			SF Bus	
		Bac	- -			GPS		_		
		_					F	An	alog Outputs	
	RLL Status Inpu						-	_	HSPCI	
	Page1 of 1 Go		D : 1 OL 1				L	_	Alarm	
Point	Point Name	Point Status	Point State	•						Reset
4	BUL STS O									Reset
1	RLL_STS 0	-	CLOSED -	•						Logout
		-		•						
-	- - -	-	-	•						
-	- - -		-							
	- - -	-	- - - -	•						
	- - - -	- - -	- - - - -							
	- - - - -	- - - -	- - - - - -	•						
	- - - - - - - - -	- - - - - - - - - - -	- - - - - - - - - -							
	- - - - - - - - - - -	- - - - - - - - - - - - - -	- - - - - - - - - - - - -							
	- - - - - - - - -	- - - - - - - - - - -	- - - - - - - - - -							
	- - - - - - - - - - - - - - - -	- - - - - - - - - - - - - - - - - - -								
	- - - - - - - - - - - - - - - - - - -		- - - - - - - - - - - - - - - -							
•         •           •         •	- - - - - - - - - - - - - - - - - - -	- - - - - - - - - - - - - - -								

#### 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 Als and Als 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.

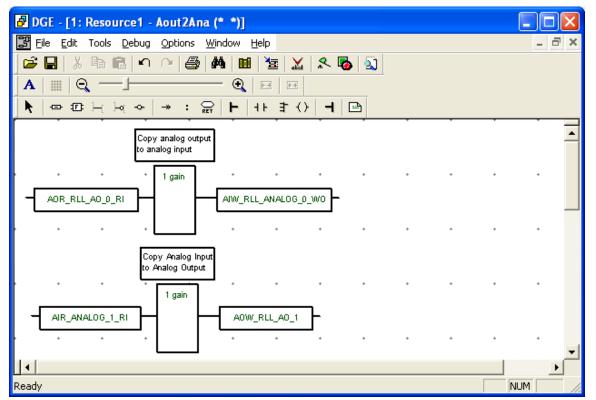
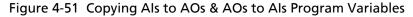


Figure 4-50 Copying Als to AOs & AOs to Als Program

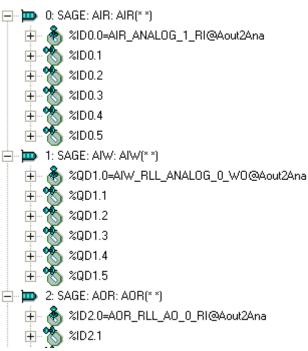
## 4.7.2 Variables



Aout2Ana												
Name	Alias	Туре	0	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	

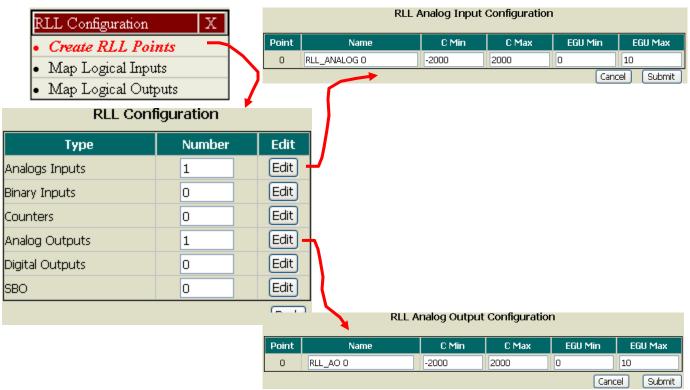
## 4.7.3 Wiring

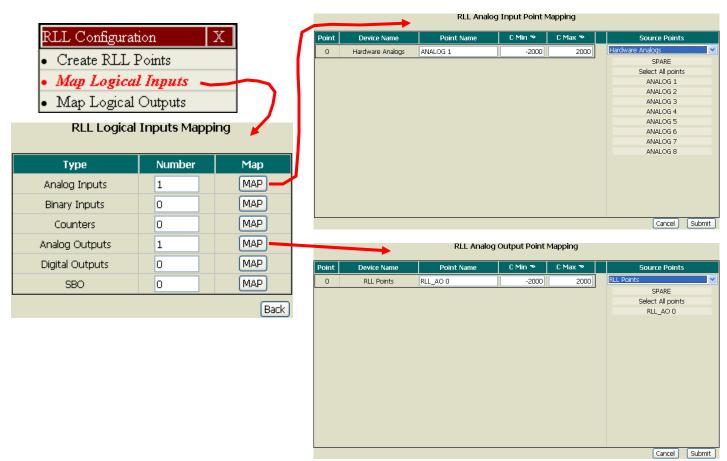




## 4.7.4 RTU Mapping

#### 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.

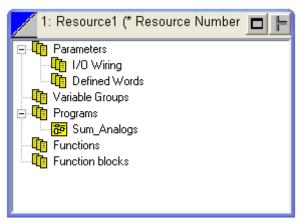
	RLL Analog Ou	itputs [	Display	
	Page1 of 1	Go Ti	o 🗾 GO	
Point	Point Name	Р	oint Status	Point Value
0	RLL_AO 0			4.342
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	-
-	-		-	
				Back

Figure 4-56 Exercising AOR to AIW

RLL Analog Outputs Command										
	Page 1 of 1		Go To	Go						
Point	Name		Range	Value	Operation					
0	RLL_AO O	0.00	00 to 10.000	7.65	Execute					
RLL_AO	0 : Success			1.	Back					
				- 7						
	RLLA	Analog	Inputs Displa	w						
		-	· · · · ·							
	Page1		Go To	Go						
Point	Point Name		Point Status	Point Value	Point Counts					
0	RLL_ANALOG 0			7.650	null					
-	-		-	-						
-	-		-	-	-					
-	· · ·		-	-	-					
-	-		-	-	-					
-	-		-	-	-					
-	-		-	-	-					
-	-		-	-	-					
-	-		-	-						
-	-		-	-						
-				-	-					
-	-		-	-	-					
-	-		-	-	-					
-	-		-	-	-					
					Back					

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.

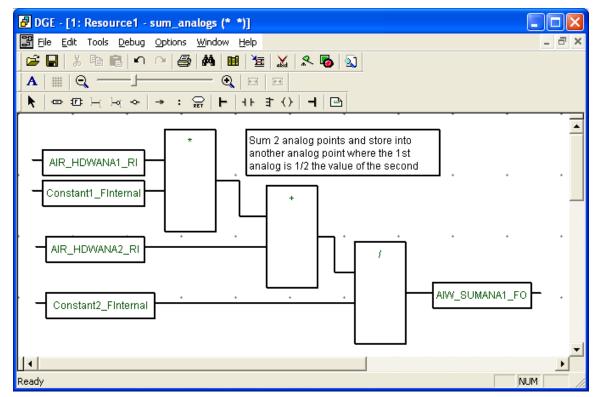


Figure 4-58 Summing Analog Points Program

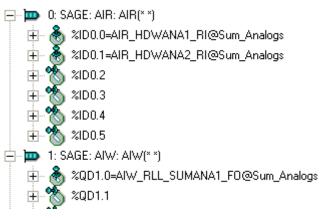
#### 4.8.2 Variables



Sum_Analogs												]	
Name	Alias	Туре	0	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

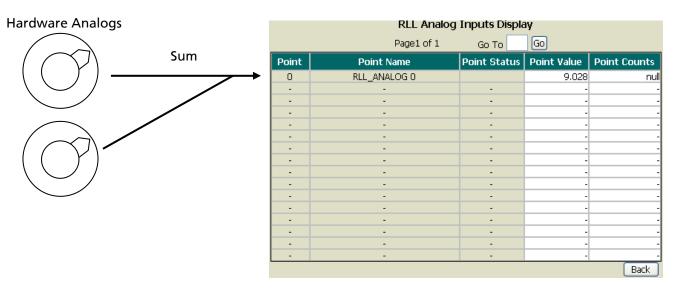
RLL Configuration	X		RLL	Analog Input	Configuration	ו	
Create RLL Poin		Point	Name	C Min	C Max	EGU Min	EGU Max
• Map Logical Input		O RL	L_ANALOG 0	-2000	2000	0	10
<ul> <li>Map Logical Outp</li> </ul>			/			Cano	el Submit
	iguration						
Туре	Number	Edit					
Analogs Inputs	1	Edit					
Binary Inputs	0	Edit					
Counters	0	Edit					
Analog Outputs	0	Edit					
Digital Outputs	0	Edit					
SBO	0	Edit					
		Back					

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

RLL Configurat	tion	X	Point	Device Name	Point Name	C Min 🔊	C Max 🔊	Source Points
• Create RLL	Pointe		0	Hardware Analogs	ANALOG 1	-2000	2000	Hardware Analogs
			1	Hardware Analogs	ANALOG 2	-2000	2000	SPARE
<ul> <li>Map Logica</li> <li>Map Logical</li> </ul>	l Inputs 🖕							Select All points ANALOG 1
	<u> </u>							ANALOG 2
<ul> <li>Map Logical</li> </ul>	Outputs							ANALOG 3
								ANALOG 4
RLL Logica	al Inputs Mapp	ing 🖌						ANALOG 5 ANALOG 6
								ANALOG 7
Туре	Number	Map						ANALOG 8
Analog Inputs	2	MAP						
Binary Inputs	0	MAP						
Counters	0	MAP						
Analog Outputs	0	MAP						Cancel Submi
Digital Outputs	0	MAP						
SBO	0	MAP						
		Back						

## 4.8.5 RTU Display

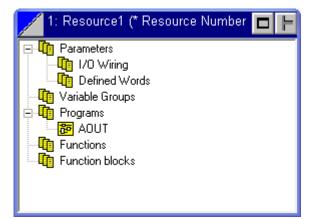
To see the program in operation you must exercise the hardware analogs, then observe the RLL analog output.



#### 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.

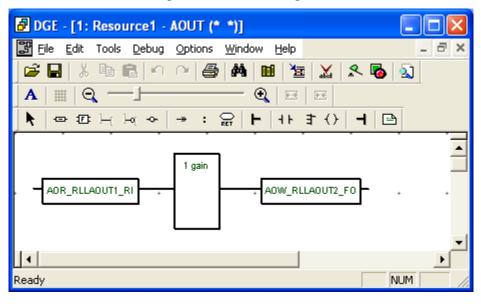


Figure 4-65 AOUT Program

#### 4.9.2 Variables

AOUT													
Name	Alias	Туре	0	Init. value	Dimensi	Group	Attribute	Scope	Direction	Retain	Wiring	Address	С
AOR_RLLAOUT1_RI		DINT				None	Read	AOUT	Input	No	%ID2.0		
A0W_RLLA0UT2_F0		DINT				None	Write	AOUT	Output	No	%QD3.0		

#### Figure 4-67 AOUT Program Wiring

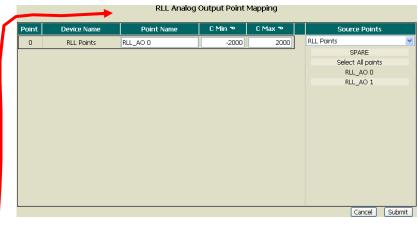
— 2: SAGE: AOR: AOR(\* \*) ሕ %ID2.0=AOR\_RLLAOUT1\_RI@AOUT ÷. ÷ 为 %ID2.1 <u>08</u> ÷ %ID2.2 প ÷ ¥ID2.3 08 ÷ %ID2.4 ÷ 🖄 %ID2.5 🚊 -- 📴 3: SAGE: AOW: AOW(\* \*) 🗄 🖓 %QD3.0=AOW\_RLLAOUT2\_FO@AOUT ÷ 🆄 %QD3.1

## 4.9.4 RTU Mapping

#### Figure 4-68 AOUT Program RLL RTU Mapping

RLL Configuration	X		RLL #	Analog Outpu	t Configuratio	n	
Create RLL Poin	its	Point	Name	C Min	C Max	EGU Min	EGU Max
<ul> <li>Map Logical Input</li> </ul>		0	RLL_AO 0	-2000	2000	0	10
<ul> <li>Map Logical Dutp</li> </ul>		1	RLL_AO 1	-2000	2000	0	10
	figuration		1			Can	icel Submit
	ilguration						
Туре	Number	Edit					
Analogs Inputs	0	Edit					
Binary Inputs	0	Edit					
Counters	0	Edit					
Analog Outputs	2	Edit					
Digital Outputs	0	Edit					
SBO	0	Edit					
		Back	<				

#### RLL Configuration Х Point 0 Create RLL Points Map Logical Inputs Map Logical Outputs **RLL Logical Inputs Mapping** Number Туре Мар MAP Analog Inputs 0 Binary Inputs 0 MAP MAP 0 Counters Analog Outputs 1 MAP MAP Digital Outputs 0 SBO 0 MAP Back



#### Figure 4-70 AOUT Program Logical Output RTU Mapping

RLL Configurati	on 3	C	-		RLL Analog	) Output Point	Mapping		
		<b>`</b>	Point	Device Name	Point Name	C Min 🔊	C Max 🔊	Source Points	
<ul> <li>Create RLL P</li> </ul>	oints		0	RLL Points	RLL_AO 1	-2000	2000	RLL Points	~
<ul> <li>Map Logical I</li> </ul>	Innute							SPARE Select All points	
	-							RLL_AO 0	
Map Logical     RLL Logical	Outputs — Outputs Mapp	ing						RLL_AO 1	
Туре	Number	Мар							
Analog Inputs	0	MAP							
Binary Inputs	0	MAP							
Counters	0	MAP						Cancel	Submit
Analog Outputs	1	MAP							
Digital Outputs	0	MAP							
SBO	0	MAP							
		Back							

#### Figure 4-69 AOUT Program Logical Input RTU Mapping

## 4.9.5 RTU Display

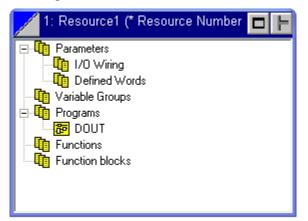
When the first RLL point is commanded (AOR), the Display shows both RLL points (AOR & AOW) changing to the commanded value.

Figure 4-71 AOUT Program RTU Display

		RLL An	alog Outputs (	Comma	ind			
		Page 1 of 1	Go To	G	io			
Point		Name	Range		٧a	lue	Opera	ation
0		RLL_AO 0	0.000 to 10.0	000		3.77	Exe	tute
1		RLL_AO 1	0.000 to 10.0	000		0.000	Exec	ute
L_AO	0 : Succe	ss						Back
		RLLA	Analog Outputs	s Displa	v			
				_	-			
		Page1		То	Go			
	Point	Point		Point S	itatus	Point		
	0	RLL_					3.770	
	1	RLL_	AO 1				3.770	
	-	· · · · · ·	•	-				
	-		-	-				
	-		- -	-				
	-			-			-	
	-	-	-	-			-	
	-		-	-			-	
	-		-	-			-	
	-			-			-	
	-	· · · · ·	•	-			-	
	-		•	-				
	-			-				
	-			-				
						ſ	Back	

## 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.

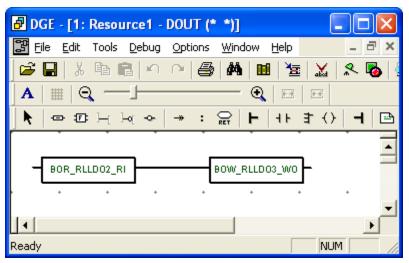


Figure 4-73 DOUT Program

#### 4.10.2 Variables



DOUT													
Name	Alias	Туре	0	Init. value	Dimensi	Group	Attribute	Scope	Direction	Retain	Wiring	Address	Comn
BOR_RLLD02_RI		BOOL				None	Read	DOUT	Input	No	%IX7.0		
BOW_RLLD03_W0		BOOL				None	Write	DOUT	Output	No	%QX8.0		

## 4.10.3 Wiring

#### Figure 4-75 DOUT Program Wiring

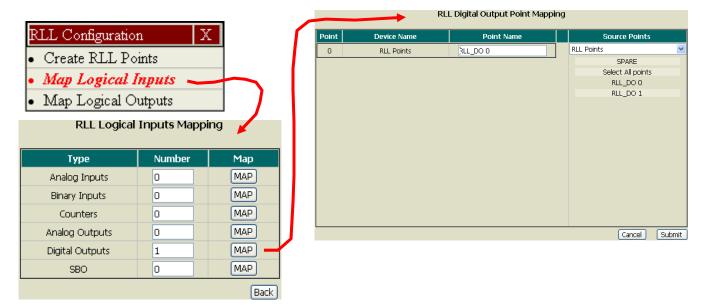


## 4.10.4 RTU Mapping

#### RLL Digitial Output Configuration RLL Configuration Х Create RLL Points Point Name Map Logical Inputs RLL\_DO 0 0 Map Logical Outputs RLL\_DO 1 1 **RLL Configuration** Cancel Submit Type Number Edit Edit Analogs Inputs 0 0 Edit Binary Inputs Edit 0 Counters Edit Analog Outputs 0 2 Edit Digital Outputs Edit SBO 0 Back

#### Figure 4-76 DOUT Program RLL RTU Mapping





RLL Configuration	on X			R	LL Digital Output Point Mapp	ing	
			Point	Device Name	Point Name	Source Points	
<ul> <li>Create RLL P</li> </ul>	oints		0	RLL Points	RLL_DO 1	RLL Points	~
• Map Logical L	nnuts					SPARE	_
	-					Select All points RLL DO 0	_
• Map Logical	Outputs 🚤					RLL_DO 1	
RLL Logical (	Outputs Mapp	ing					
Туре	Number	Мар					
Analog Inputs	0	MAP					
Binary Inputs	0	MAP					
Counters	0	MAP					
Analog Outputs	0	MAP	]			Cancel	Submit
Digital Outputs	1	MAP -					
SBO	0	MAP					
		Back					

#### Figure 4-78 DOUT Program Logical Output RTU Mapping

## 4.10.5 RTU Display

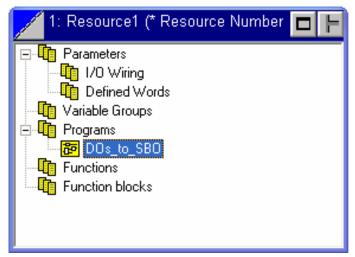
When the first RLL point is commanded (BOR), the Display shows both RLL points (BOR & BOW) changing to the commanded value.

Figure 4-79 DOUT Program RTU Display

RLL Digital Outputs Command								
		Page 1 of 1 🔅	юТо		Go			
F	Point	Name		Poir	nt Opera	ntions		
	0	RLL_DO 0	0	Open	📀 Clos	e Execute	-	
	1	RLL_DO 1	0	Open	O Clos	e Execute		
C	lose or	n RLL_DO 0 : Successful				Back		
		RLL Digital Ou	tpu	its Disi	plav			
		Page1 of 1	Go		Go			
Point		Point Name	00	_	Status	Point State	•	
1		RLL_DO 0				CLOSED	•	
2		RLL_DO 1				CLOSED	•	
-		-			-	-	-	1
-		-			-	-	-	
-		-			-	-	-	
-		-			-	-	-	
-		-			-	-	-	
-		-			-	-	-	
-		-			-	-		
-		-			-	-	-	1
-		-			-	-	-	
-		-			-	-	-	
-		-			-	-	-	
-	_	-			-	-	-	
-		-			-	-	-	]
						Ba	:k	

## 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.

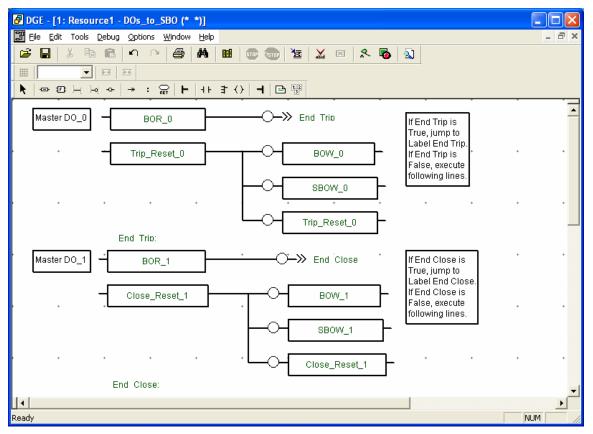


Figure 4-81 DOs to SBO Program

## 4.11.2 Variables

Figure 4-82 DOs to SBO Variables

ISaGRAF PRO - [DOs_to_SBO]           Eile         Edit         Debug         Tools         Options										_ 5
- 	<b>3</b>   <b>1</b>   <b>2</b>   <b>3</b>   <b>1</b>   <b>1</b>	- 🛠 🐻								
추 뿜 🛍 🖿 🗷 🔬 🛛 🎟 👘		t								
📐 😽 🏘 🔸	DOs_to_SBO									
BM_ Variables ÈA Resource1 (Config1)	Name	Alias	Туре	0	Init	Dim	Group	Attribute	Scope	Direction
Any Group	BOW_0		BOOL				None	Write	Global	Output
All variables	Trip_Reset_0		BOOL				None	Free	Global	Internal
Global variables	SBOW_0		BOOL				None	Write	Global	Output
DOs_to_SBO	BOW_1		BOOL				None	Write	Global	Output
	BOR_0		BOOL				None	Read	Global	Input
	BOR_1		BOOL				None	Read	Global	Input
	Close_Reset_1		BOOL				None	Free	Global	Internal
	SBOW_1		BOOL				None	Write	Global	Output
	··· 🛆		$\wedge$					$\wedge$		$\wedge$
	٦٢		][					][		ון
	_		_					_		_
	<									
LVENT-OBLNWYS.rd-bennieg Resource	e 1: Config1\Resource:	1 (* Resource N	umber 1 *)							NUM

## 4.11.3 Wiring

#### Figure 4-83 DOs to SBO Wiring

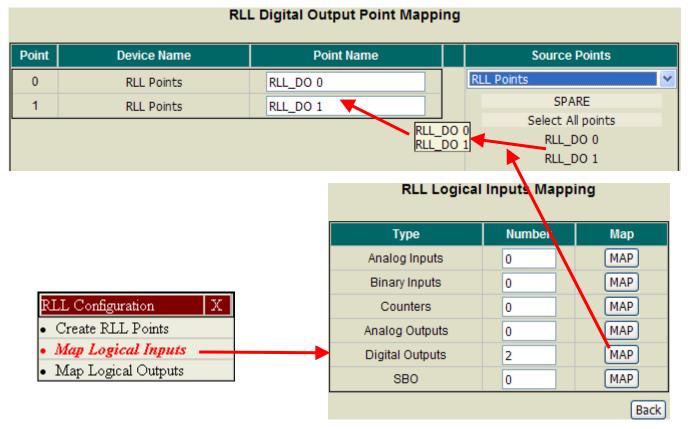
🔀 ISaGRAF PRO - [DOs_to_SBO(* *) - I/O Wiring - 1: Resource1(* Res	sour	ce Number 1*) ]		
🕮 Eile Edit Options Window Help				_ @ ×
응 명 🗰 🖼 🖏				
🖻 🖩 🍯 🐚 👗 🗠 🗠 📽 📽 🕷 🧇				
- 💬 7: SAGE: BOR: BOR(* *)	^	Unwired varia	bles - Type: Dl	NT - Direction: Ir
🗄 🗄 %IX7.0=BOR_0		Name	Alias	Comment
<ul> <li>⊕</li></ul>				
B 8: SAGE: BOW: BOW(* *)				
& xQX8.0=BOW_0 <				
⊕ % %QX8.1=BOW_1     √     ⊕ % %QX8.2				
È ┣ 12: SAGE: SBOW: SBOW(* *)				
🗄 💩 %QX12.0=SBOW_0				
⊕ %QX12.1=SBOW_1 <				
the <sup>4</sup> / <sub>2</sub> % 20×12.2				

## 4.11.4 RTU Mapping

Figure 4-84 DOs to SBO Create RLL Point

RLL Configuration	X		RLL Dig	itial Outp	put Con	figurat
Create RLL Points     Map Logical Inputs			Point		Name	
<ul> <li>Map Logical Outputs</li> </ul>			0	RLL_DO	0	
			1	RLL_DO	1	
RLL Cor	nfiguration		1		Cancel	Submi
Туре	Number	Edit				
Analogs Inputs	0	Edit				
Binary Inputs	0	Edit				
Counters	0	Edit				
Analog Outputs	0	Edit				
Digital Outputs	2	Edit				
SBO	0	Edit				
		Back				

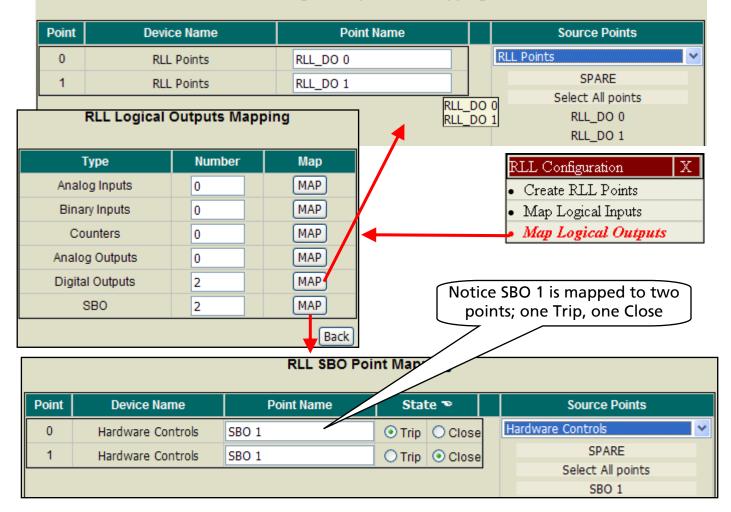
Figure 4-85 DOs to SBO Logical Input RTU Mapping



#### S2200-AAA-00003

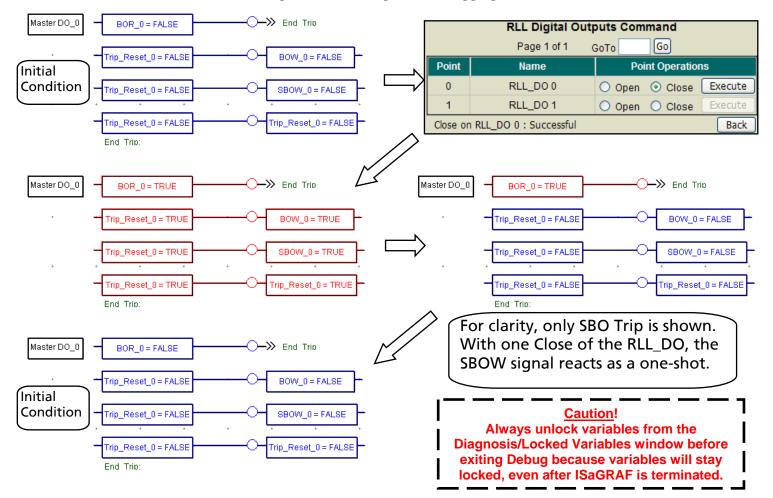
#### Figure 4-86 DOs to SBO Logical Output RTU Mapping

RLL Digital Output Point Mapping



## 4.11.5 Testing With Debugging

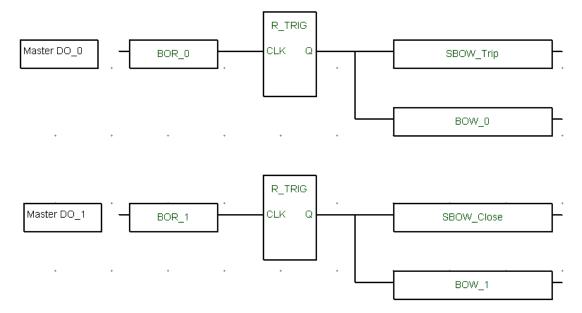
Figure 4-87 Testing With Debugging



## 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
С	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.
СТЅ	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;
	FRF = Data Value – Data Min
ГС	Data Max – Data Min
FS	Full Scale
FTP	File Transfer Protocol – A TCP/IP application used for
CDC	transferring files from one system to another
GPS GUI	Global Positioning System
H	Graphical User Interface
	Hexadecimal (base 16), as in XXXXh
HEX HDLC	Hexadecimal (base 16), as in XXXXh High-level Data Link Control
HSPCI	High Speed Pulse Counter Input
Hz	
I/O	Hertz, frequency in cycles per second
ID	Input/Output 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
20	programming languages
LED	Light Emitting Diode
LRC	
LNC	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
Тх	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).
ХТ	External Termination (panel, module or assembly)

## APPENDIX B

#### D

Downloading & Recovering Code to/from Target						
Import/Export Templates 3-83						
Installation						
Installation Package Contents 2-1						
Installation Procedure						
Installing ISaGRAF PRO 2-2						
Installing Telvent-Provided Components 2-3						
PC Requirements2-1						
RTU Requirements 2-1						
Introduction						
How to Determine Your Number-Of-Points						
Supported1-3						
Multiple Programs in the RTU 1-2						
Overview1-1						
Points Supported 1-2						
Reference Documents 1-2						
ISaGRAF Program Maintenance						
0						
Operation						
Changing the IP Address						
Changing the IP Address 3-54 Changing the Program Cycle Time						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address						
Changing the IP Address3-54Changing the Program Cycle Time3-55Changing Variable Attributes3-29Compiling the Project and Downloading tothe RTUthe RTU3-38Configuring the RTU for RLL3-41Creating Simple Programs3-1Drivers for Inputs to Logic3-45air (Analog Input Read)3-45bir (Binary Input Read)3-45birm (Binary Input Read)3-45bor (Binary Output Read)3-45contr (Counter Input Read)3-46sbor (SBO Read)3-46						
Changing the IP Address3-54Changing the Program Cycle Time3-55Changing Variable Attributes3-29Compiling the Project and Downloading tothe RTUthe RTU3-38Configuring the RTU for RLL3-41Creating Simple Programs3-1Drivers for Inputs to Logic3-45air (Analog Input Read)3-45bir (Binary Input Read)3-45birm (Binary Input Read)3-45bor (Binary Output Read)3-45contr (Counter Input Read)3-46sbor (SBO Read)3-46Drivers for Outputs from Logic3-46						
Changing the IP Address3-54Changing the Program Cycle Time3-55Changing Variable Attributes3-29Compiling the Project and Downloading tothe RTUthe RTU3-38Configuring the RTU for RLL3-41Creating Simple Programs3-1Drivers for Inputs to Logic3-45aor (Analog Input Read)3-45bir (Binary Input Read)3-45bir (Binary Input Read)3-45bor (Binary Output Read)3-45contr (Counter Input Read)3-46sbor (SBO Read)3-46Drivers for Outputs from Logic3-46						
Changing the IP Address3-54Changing the Program Cycle Time3-55Changing Variable Attributes3-29Compiling the Project and Downloading tothe RTUthe RTU3-38Configuring the RTU for RLL3-41Creating Simple Programs3-1Drivers for Inputs to Logic3-45air (Analog Input Read)3-45bir (Binary Input Read)3-45birm (Binary Input Read)3-45bor (Binary Output Read)3-45contr (Counter Input Read)3-46sbor (SBO Read)3-46Drivers for Outputs from Logic3-46						

bow (Binary Output Write)	
cntw (Counter Write)	
sbow (SBO Write)	. 3-46
Introduction	
Managing Multiple Programs for Differer	nt
RTUs	
Multiple Programs in the RTU	. 3-44
Opening an Existing Project	3-4
Program Debug	
Program Simulation	
Removing/Adding/Modifying Drivers	
RTU Communications Settings	
Starting a New Function Block Diagram (F	BD)
Program	
Starting a New Ladder Diagram (LD) Prog	jram
Starting a New Project	
Testing Your Programs	. 3-44
Р	
Program Maintenance	. 3-62
Programming Principles & Examples	
BIRM, BIR Notes	4-4
Converting 2 DOs from Master to 1 SBO	
Trip/Close to IED	
Copying AO to AI & AI to AO	
Copying AOR to AOW	
Copying BOR to BOW	
Hardware AI to RLL (Pseudo) Points	
Introduction	
MTU/RTU Programming Model	
Reducing Status Points	
Summing Accumulator Points	
Summing Analog Points	
Lising Input SPO BLL (Proudo) Points	1 1 1

Using input SBO KLL (Pseudo) Points	4-11
R	
RLL Command Output	3-89
RLL Configuration	3-63
RLL Data Display	3-84
Т	
Templates, Import/Export	3-83