EtherLogic Quick Start Guide

Introduction

ScadaWorks is a suite of software tools that help you quickly and easily configure your ICL controller for SCADA (Supervisory Control and Data Acquisition) and other applications.

The Quick Start Guide walks you through the essentials of installing the software, creating and running your first application.

Once you have the ScadaWorks software installed on your PC, additional information is available in the form of on-line help. The help files may be accessed directly through the ScadaBuilder Workbench "Help" menu. You can also get help on a specific setup parameter by pressing the F1 key when the parameter is selected. The same help can be accessed by clicking on the parameter with the "What's This" pointer (use the  button found in the upper right corner of most of the windows).

Technical Support

If you purchased your equipment and software through a distributor, please contact your distributor first for help.

If you purchased your equipment and software direct from ICL, contact us by phone or email using the information shown below:

Industrial Control Links, Inc.
12840 Earhart Avenue
Auburn, CA USA 95602
(800) 888-1893 (toll free in the USA)
(530) 888-1800
email: icl@iclinks.com
website: www.iclinks.com

Installing the Software

The ScadaWorks software suite consists of the components listed below. The minimum operating system requirement is Microsoft Windows 95.

Performing a "Typical" installation will install all of the components onto your development PC. A "Minimal" installation will only install the ScadaBuilder component. A "Custom" installation will allow you to select which components to install. ScadaWorks consists of the following software components.

ScadaBuilder

ScadaBuilder ties all the components together and provides the tools to help you configure ICL controller(s) for your applications.

ScadaBuilder Samples

A number of sample Projects are provided to illustrate the various features and functions of ScadaBuilder and ICL controllers. It is not required but is recommended that you install the samples.

ISaGRAF Support Kit

The ISaGRAF Support Kit allows you to develop control logic that will be executed on the target controller. This component consists of a workbench and a runtime kernel. The ISaGRAF Workbench is an IEC-61131-3 standards-compliant development tool that supports 6 development languages. The ISaGRAF Runtime Kernel and supplemental functions allow you to create ISaGRAF projects that take advantage of ICL controller hardware.

TCP/IP Support Kit

The TCP/IP Support Kit provides support for TCP/IP and related protocols over Ethernet or dialup/serial connections.

Voice Support Kit

The Voice Support Kit provides support for voice related features such as alarm dialing and the VUI (Voice User Interface). These features require the voice modem hardware option to be installed or connected to the unit.

Simple Application Walkthrough

To show you the basic steps you will use to develop a ScadaBuilder application, we will walk you through creating, transferring and running a simple application. This application sets up the target controller to do some simple control logic and to act as a Modbus slave. ("Modbus" is a serial communications protocol that is commonly used in industrial automation systems.)

After we go through the mechanics of using ScadaBuilder, we'll spend a little time talking about ScadaBuilder concepts and look more closely at the application you have created.

The hardware required for this tutorial is one ICL EtherLogic controller with some built-in analog inputs, digital inputs and digital outputs. The I/O on the controller will be mapped to registers that are accessible via Modbus.

We will assume at this point that you already have the following software components installed on your PC:

- ScadaBuilder Workbench
- ISaGRAF Support Kit
- TCP/IP Support Kit

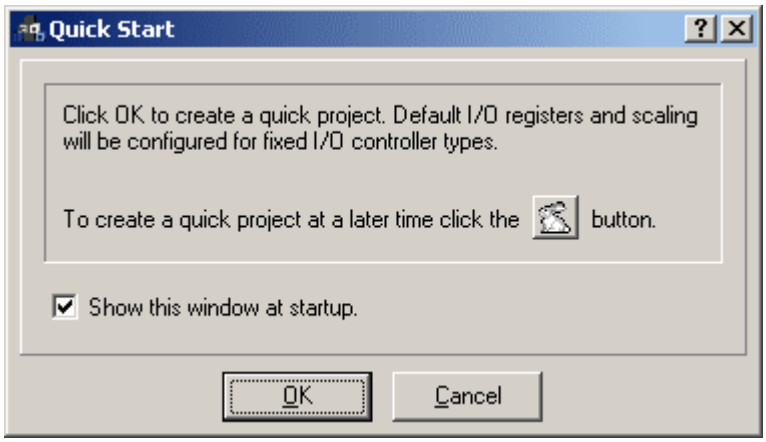
If you do not already have these components, they are available on the ScadaWorks CD.

Let's get started, shall we?

Creating a Project

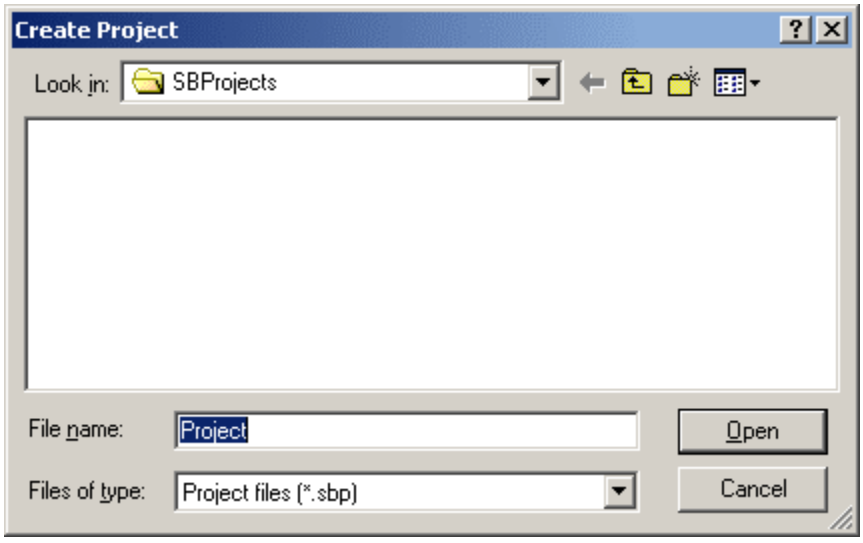
Follow the steps below to create a basic ScadaBuilder Project (which we will then build upon):

- Start the ScadaBuilder Workbench (normally on the Start menu under "Programs" | "ICL Tools" | "ScadaBuilder"). You will see the following Quick Start window:



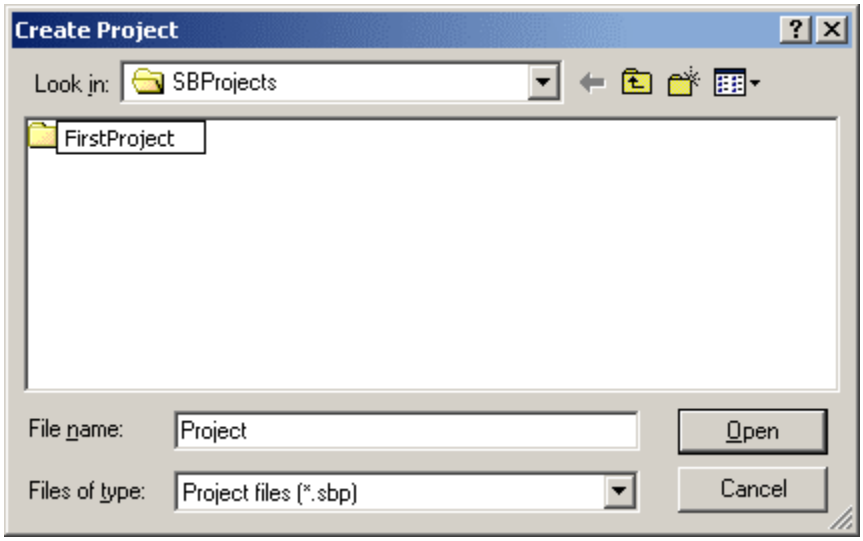
  This allows you to get started quickly with ScadaBuilder by helping you set up your first project.

Click "OK" to continue. ScadaBuilder then prompts you to select a directory for you project. You will see the following:

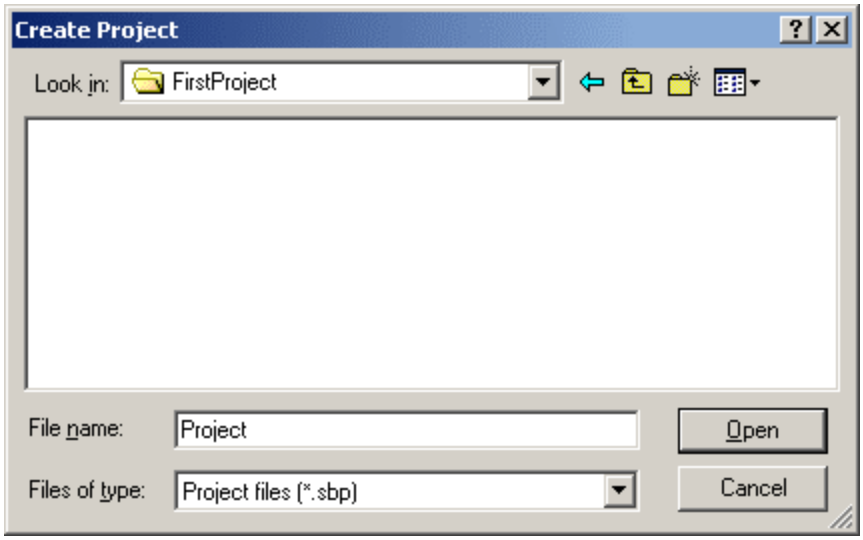


  You can store your projects wherever you like, but the default folder is "C:\SBProjects". Each project must be in its own folder (typically a sub-folder of "SBProjects").

- Create a sub-folder for your first project by clicking on the  button. Type in the name "FirstProject" as shown:

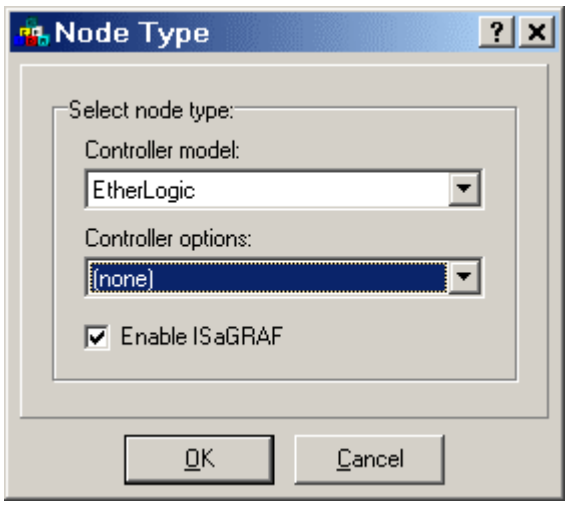


- Double-click the new folder name to go into that folder. (Note: if you are running on Windows 2000, you may get an error message and have to double-click again due to a pesky quirk in the operating system). The resulting window will look like this:

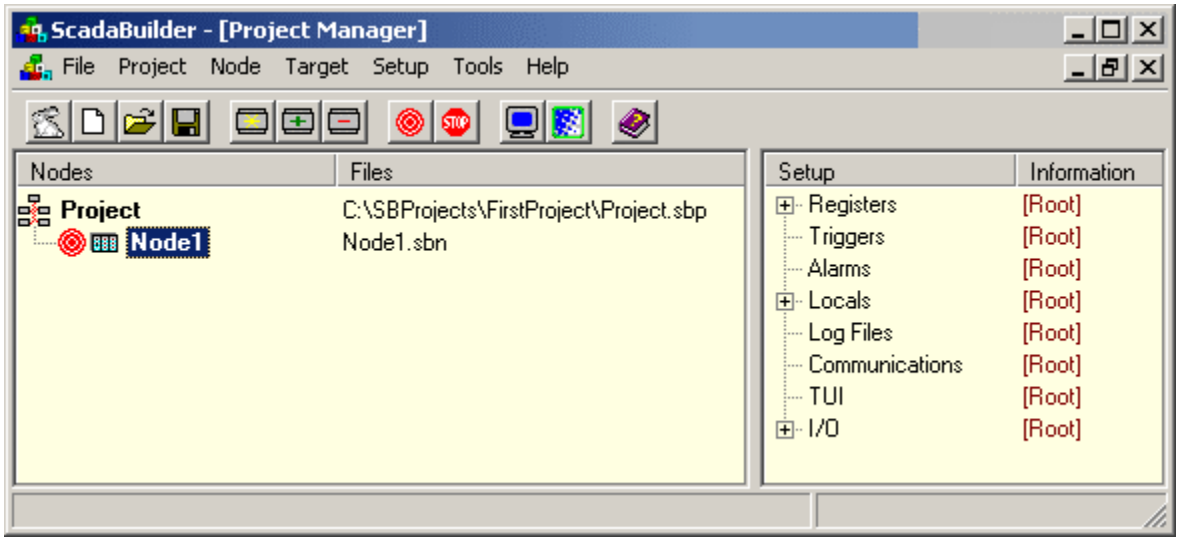


- Click "Open" to accept the default project file name ("Project").

- On the next window that appears, select the controller model and any options that represent the controller you have (or the one you are pretending to have). The window below shows a standard EtherLogic controller selected:



- Leave the "Enable ISaGRAF" option checked and click "OK." You will see a window that looks like this:



You have now successfully created a basic ScadaBuilder Project -- that wasn't too hard, was it?

Next, let's build on this by adding some simple control logic.

Control Logic

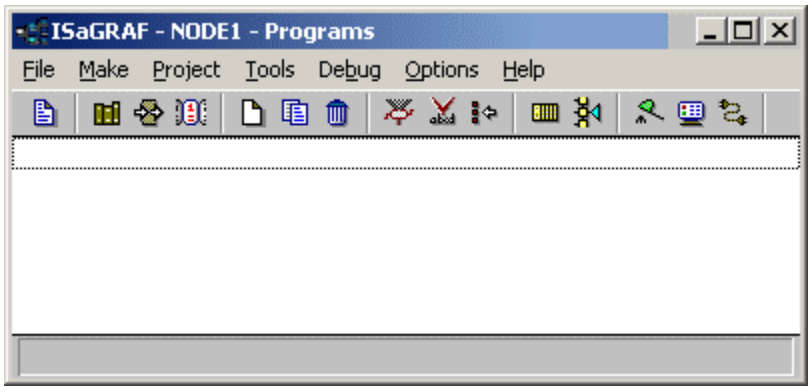
Let's add some basic logic. If you wanted to describe the desired logic as a sentence, you could write:

> "If digital input 1 or digital input 2 is ON, turn digital output 1 ON, otherwise turn digital output 1 OFF."

We will use ISaGRAF to represent this logic in a form that the controller can understand. ISaGRAF is a standards-compliant tool that is integrated with ScadaBuilder, and can be used to create logic of just about any level of complexity. With ISaGRAF, you have the choice of 6 different "languages" to tell the controller what you want it to do. For the most part, any of the languages can be used to accomplish a task, but the one(s) you choose will be based on your preferences and the types of tools that you are already familiar with. The languages are both graphical and text-oriented and you can use multiple languages within the same Node.

For this introduction, we will use "Ladder Diagram," or LD. Let's continue:

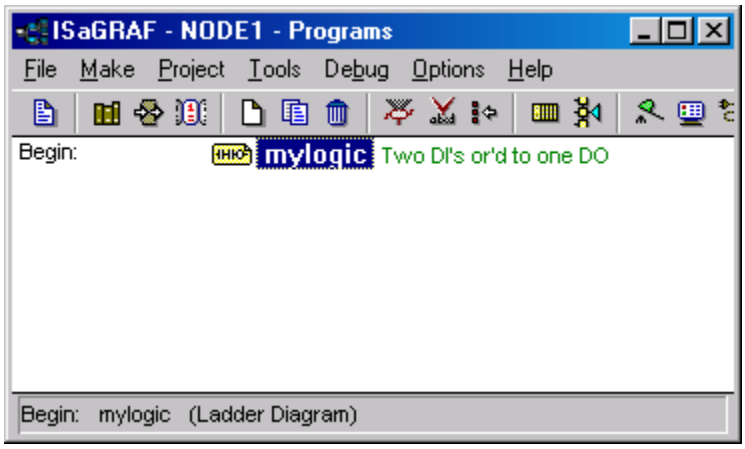
- Make sure that "Node1" is selected in the ScadaBuilder project window.
- Click the  button to bring up ISaGRAF. You should see an empty "Programs" window, like this:



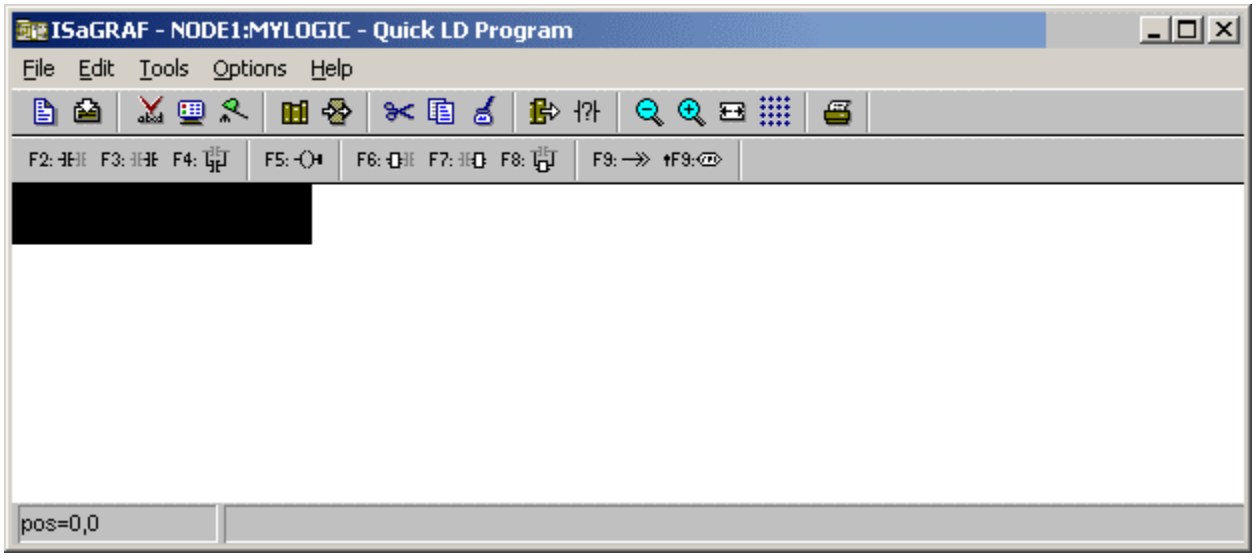
Multiple "Programs" can be created to represent different portions of your control logic. We will create just one simple Program.

- Choose "File" | "New" from the menu. The window below will appear. Type in a name and (optionally) a comment as shown. Make sure "Quick LD" and "Begin" are selected, then click "OK:"

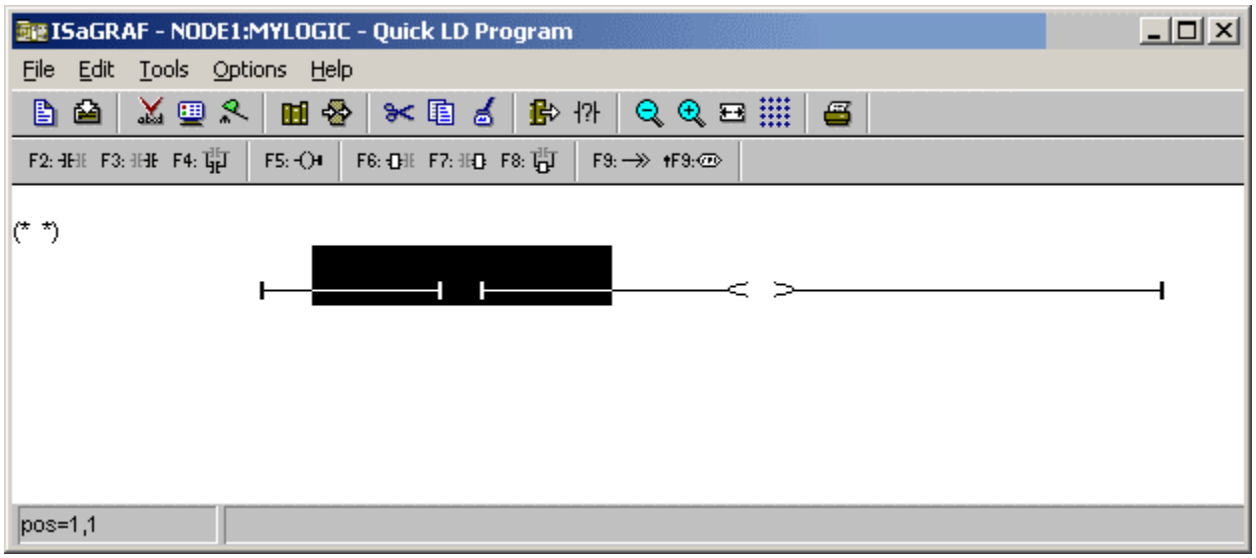
- Now your Programs window will look like this:



- Double-click on the "mylogic" program, to bring up a blank Quick LD program editor window:

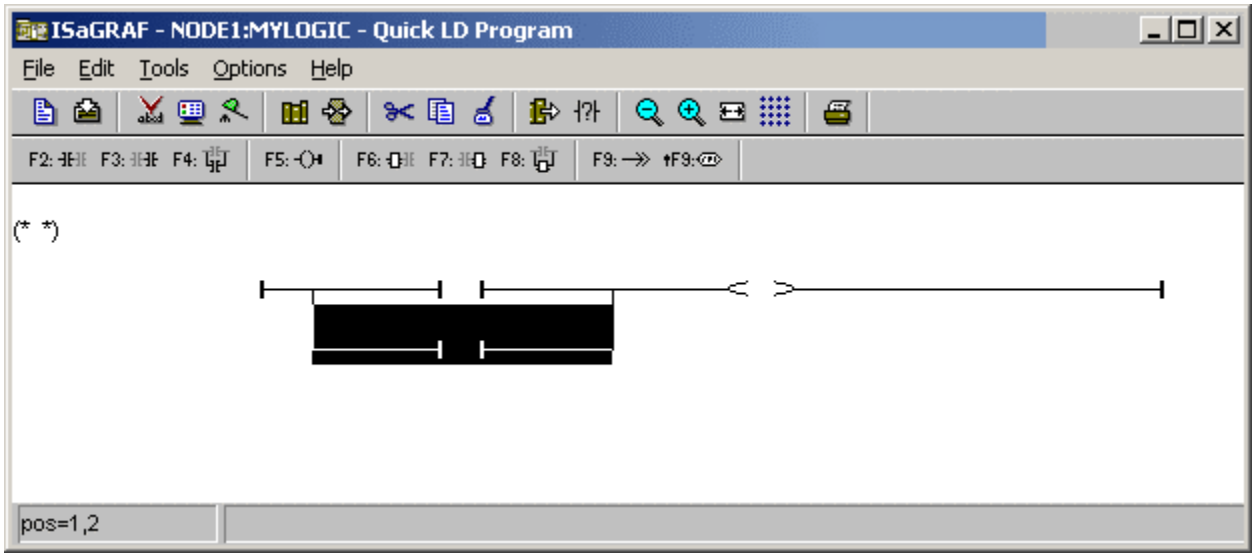


- Click on the F2: button (or press F2) to tell ISaGRAF that you would like to place a contact. The window will then look like this:



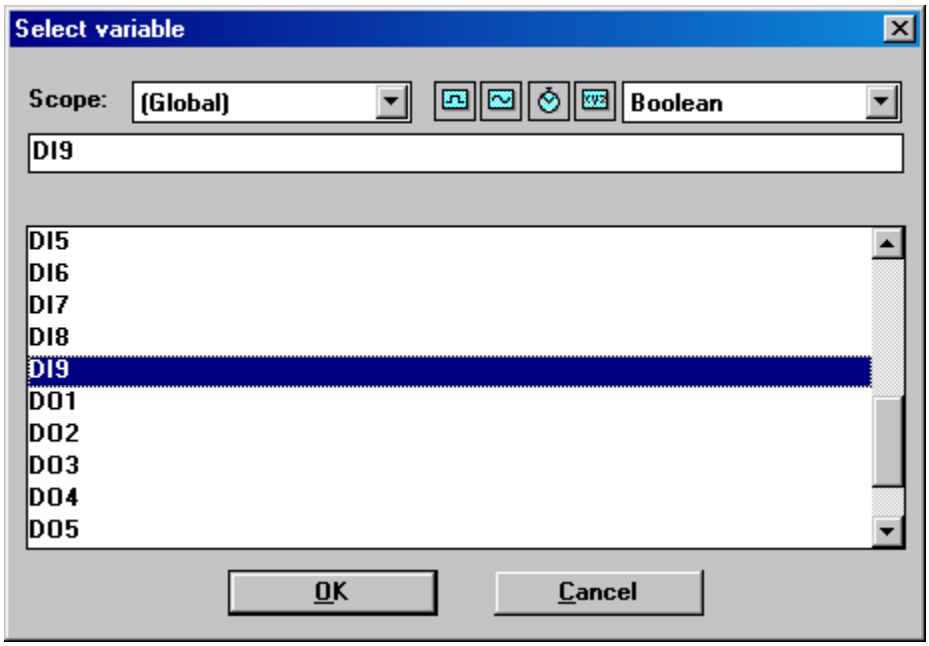
Notice that the Quick LD editor automatically makes a complete "rung," including a coil.

- Click the F4: button (or press F4) to add a parallel contact below the first contact. The result will look like this:

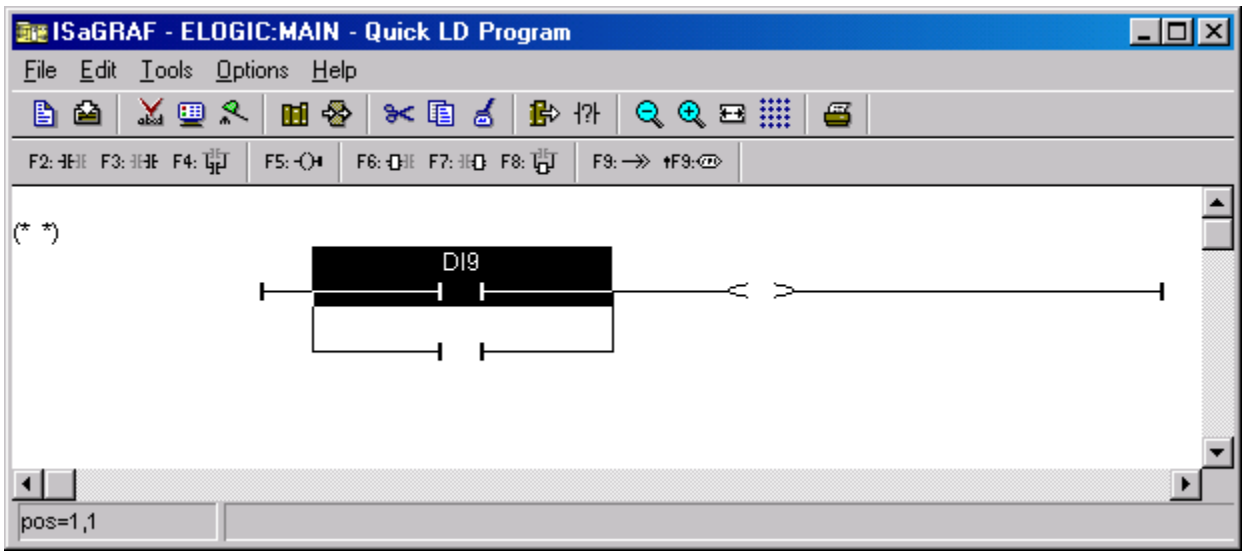


We are now done adding symbols and will assign registers to the contacts and coils.

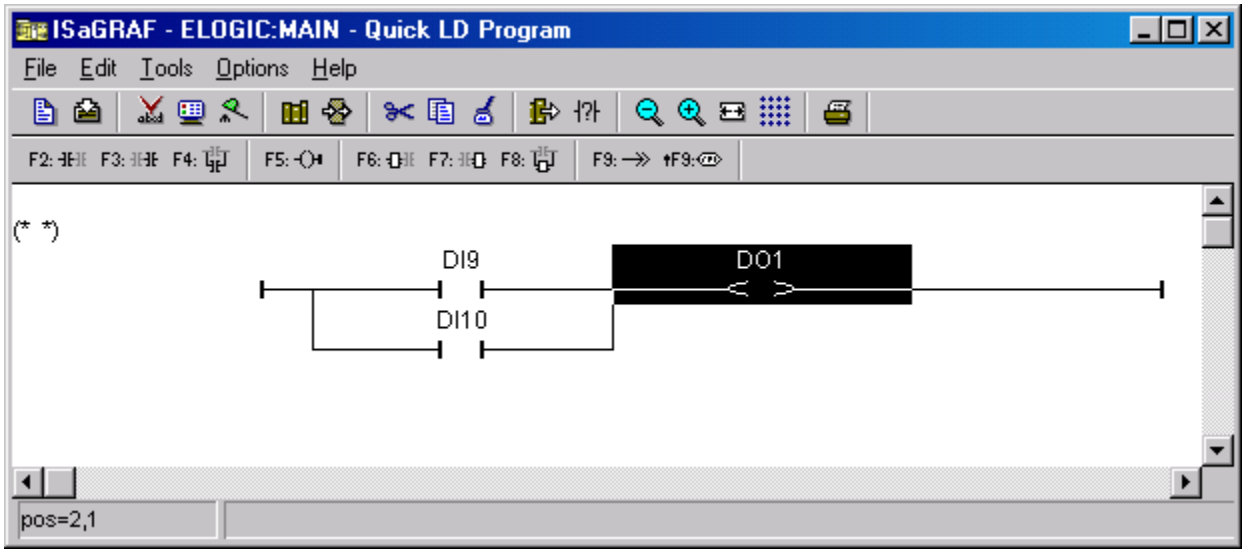
- Double click the contact at the top of the diagram. The following window will appear, which allows you to select a register (also known as a "variable" in ISaGRAF) to be associated with the contact:



- Make sure "DI9" is selected, then click "OK." The result should look like this:



- Repeat this process for DI10 and DO1, until your window looks like this:



We are using DI9 and DI10 for simplicity. On the EtherLogic controller, the first 8 Digital Inputs are tied to the Universal style and require certain modes to be set to operate properly. Please see the EtherLogic Hardware Reference Guide for more details on the Universal Inputs.

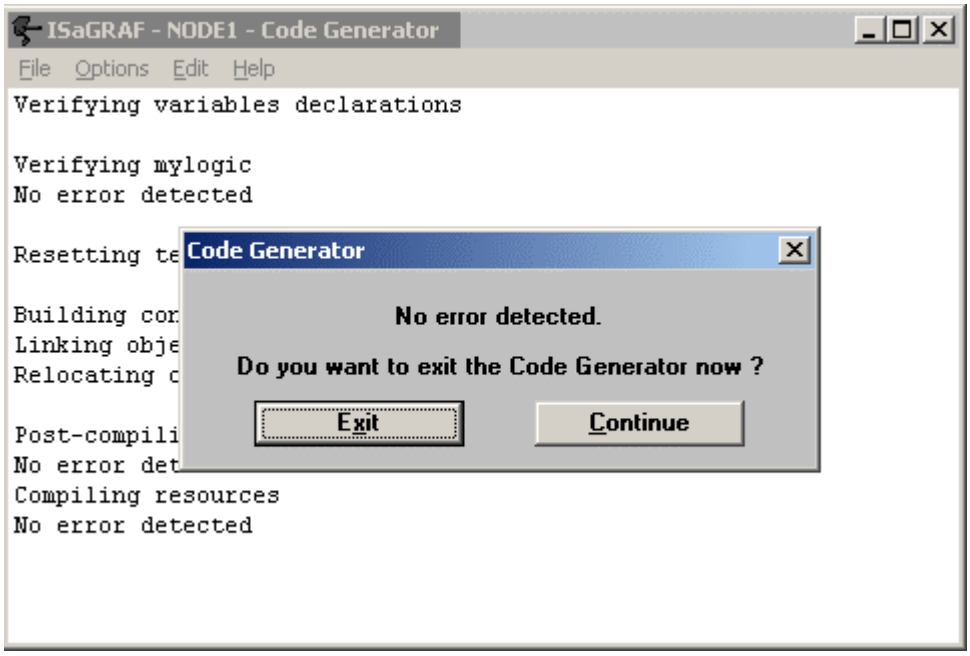
- Click the  button to save your work and then close the window.

You have now completed the control logic part of the application. Coming up next, we will "make" the application. After that, we will prepare the target controller, load the application and execute it to see if it works as expected.

Make Application

Before you try to load your application onto the target controller, you must first "make" it. This will take the application in its "source" form and compile it into a form necessary for execution. Both your ScadaBuilder setup and ISaGRAF control logic will be combined into a single file.

- From the ISaGRAF programs window, click the  button. After the application makes, you will see something like this:



- Click "Exit" to close the code generator window.
- You may now close the ISaGRAF programs window.

Your application is now in the form needed for execution on the target controller.

Whenever you make a change to the setup in the ScadaBuilder Workbench or change your ISaGRAF logic, you need to "make" the application before downloading it to the target if you want your changes to take effect.

Preparing the Target Controller

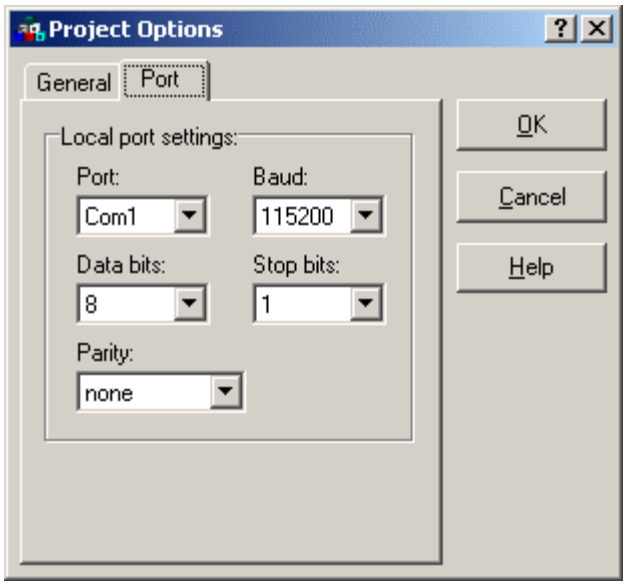
Now let's prepare the target controller so it is ready to accept and run your newly created application. Refer to the appropriate hardware reference manual for power supply requirements and other details.

Power and Communications Connections

- Attach a null-modem cable (available from ICL) from your development PC to the controller. Connect to the console (Com1) port on the controller.
- Provide power to the controller.

Development PC Port Setup

- From the ScadaBuilder Workbench menu, choose "Project" | "Options."
- Click the "Port" tab to set the port parameters as they apply to your development PC. You will see the following window:

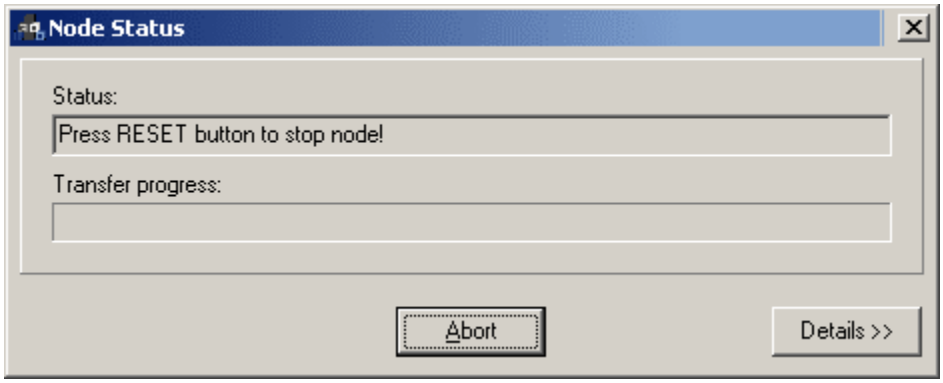


- For the "Port" setting, select the Com port that you have connected the null-modem cable to. The other settings should be selected as shown. (Remember, you are configuring the settings of your development PC serial port, NOT the port on the target controller.)

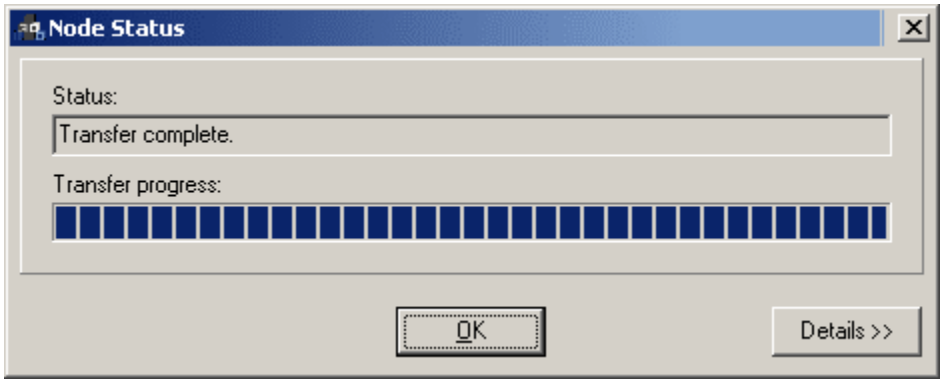
ISaGRAF Setup

Your controller would normally be set up at the factory to run ISaGRAF applications, but we should update the configuration to make sure that the controller matches the software that is installed on your development PC. Follow these steps:

- In the ScadaBuilder Workbench, make sure the "Node1" is selected.
- Choose "Target" | "Send ISaGRAF Startup Config." Press the reset button (or power-cycle) on the controller if prompted:



- The startup configuration sets the controller up so that it will automatically run ISaGRAF on startup, and will automatically run any loaded application). When the transfer is complete, click "OK" to close the status window:



- Next, choose "Target" | "Send ISaGRAF Runtime Kernel." Press reset again if prompted. The Runtime Kernel is part of the ISaGRAF Support Kit, and is the "engine" that executes your ScadaBuilder/ISaGRAF application. When the transfer is complete, click "OK" to close the status window. (This transfer typically takes a little over a minute.)
- Press the reset button on the controller one more time to automatically start the newly loaded Runtime Kernel.

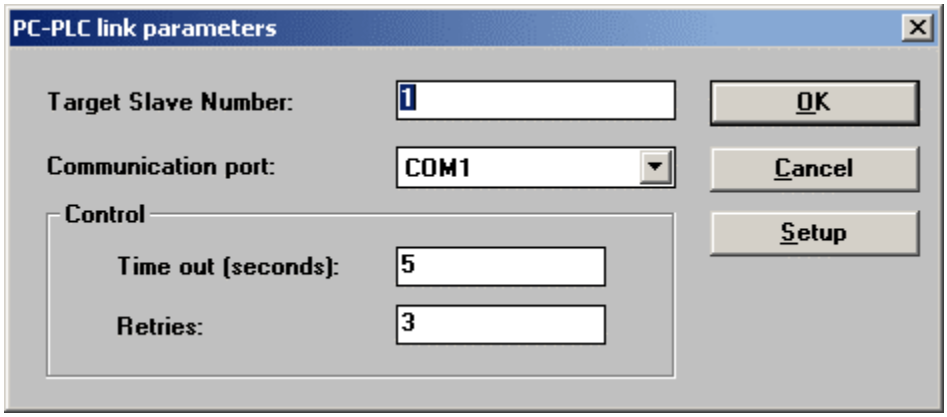
If you encounter any errors, double check to see that the cable is correct and that the communications port on the host PC is selected properly ("Project" | "Options" | "Port"). The PC com port baud rate must match that of the target Com1 (console) port. The most common settings are 115200 baud and 9600 baud. Units produced since about February 2002 have a factory default setting of 115200 baud. Earlier units had a default of 9600 baud. (The baud rate setting can also be changed with a utility called "syscfg" -- contact ICL for details).

Once you send the Runtime Kernel and Startup Config to the target, you don't need to do this step each time. You only need to send it again if you start working with a different target or you get an updated ISaGRAF Support Kit.

Debugger Link Setup

Now we will set the parameters for the link that the ISaGRAF Debugger uses to communicate with the target. Normally you will use the same PC port that you used to transfer the Runtime Kernel and Startup Configuration (set via "Project" | "Options" above).

- Click the  button to open the ISaGRAF Programs window.
- Choose "Debug" | "Link setup" to bring up the following window:

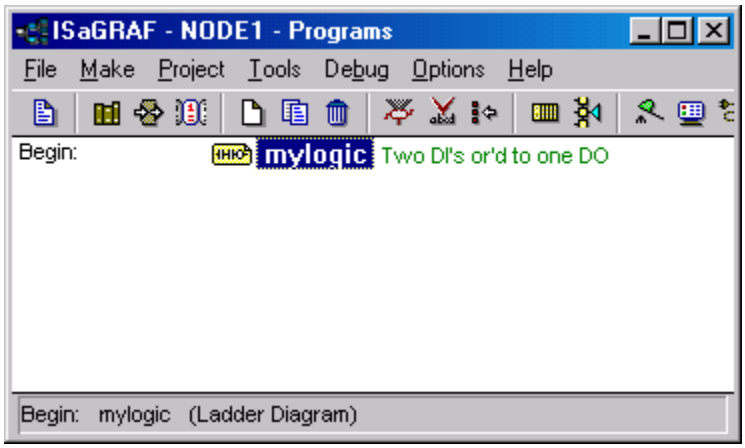


- For the "Port" setting, select the same PC Com port as previously (the one that has the null-modem cable connected). The other settings should be left at their defaults, as shown. (Once again, remember that you are configuring the settings of your development PC serial port, NOT the port on the target controller.)

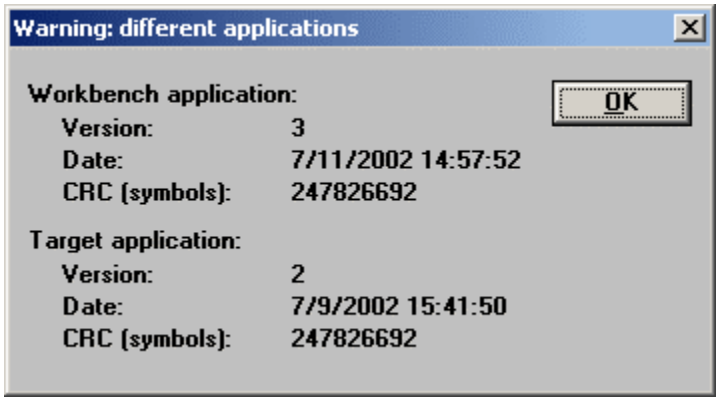
Send to Target/Run

We are now ready to send the application to the target and run it.

- From the ISaGRAF Programs window (shown below), click the button to start the ISaGRAF Debugger.

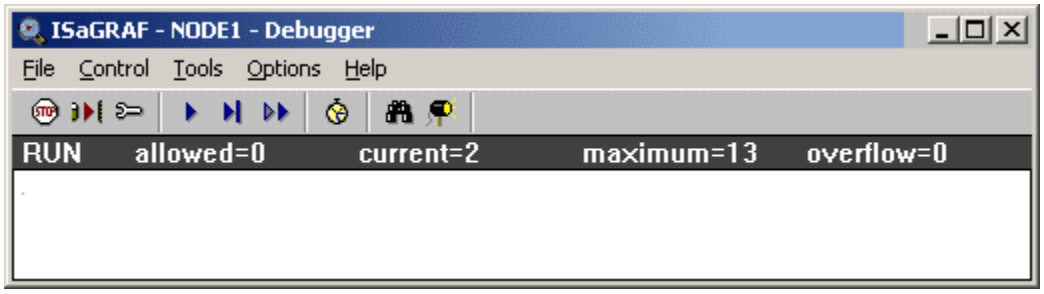


- (The Debugger allows you to send the application to the target controller and to monitor its operation.) If an ISaGRAF application has already been loaded on the target, you may see a window similar to this:



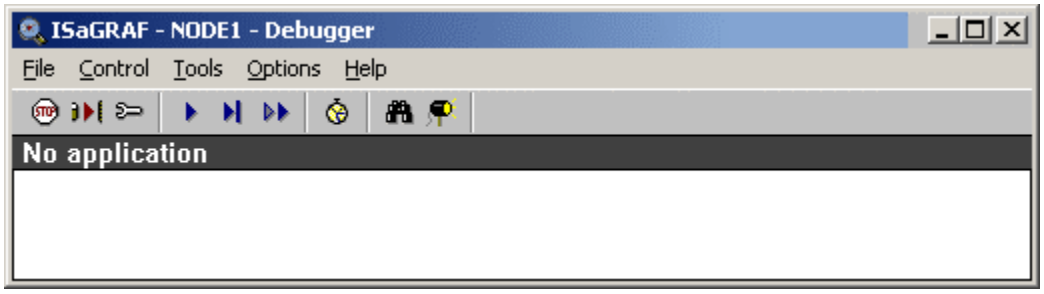
- If this window comes up, click "OK" to close it.

- Next, you will see a window similar to this:

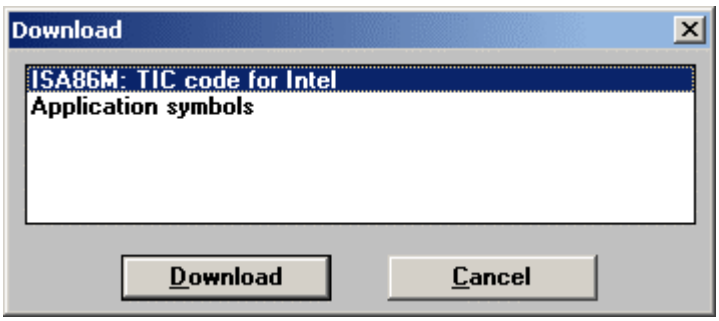


or this:

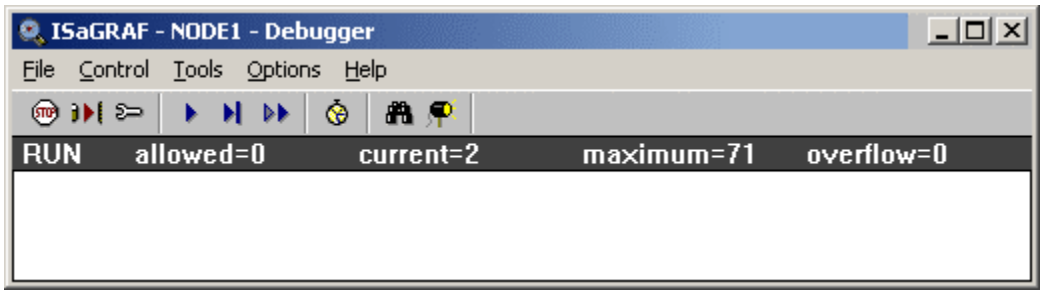
- Unless the window already say "No application," this means an application is running. Click the  button to stop it and choose "Yes" at the prompt. Now the Debugger window should look like this:

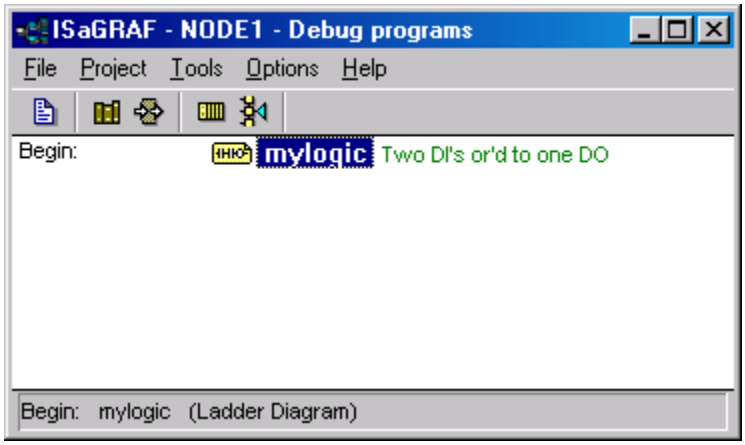


- Click the  button to request that the Debugger download your application, then click the "Download" button on the following window:



- Your will see a progress bar as the application is transferred, then the following windows once the application has started running:



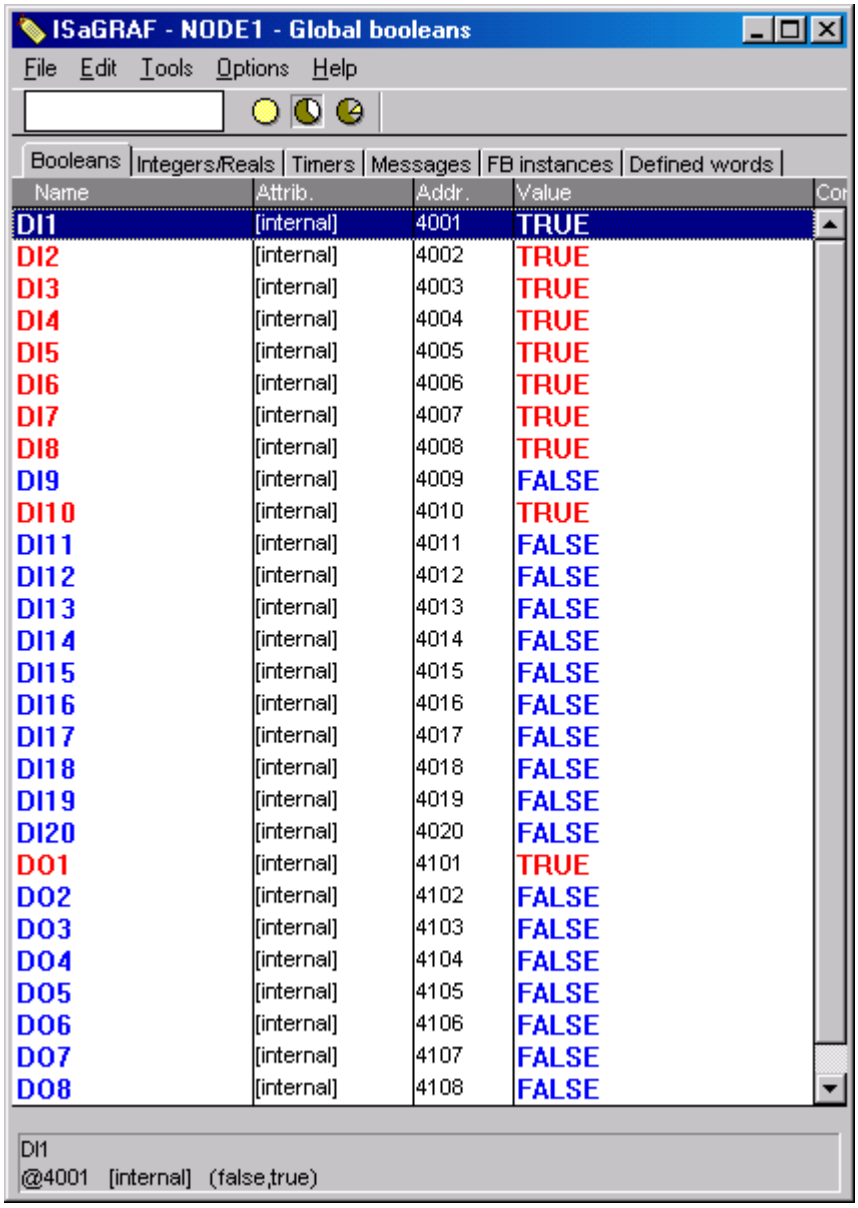


You should also see the Status LED blinking on the front of the controller.

Test

Now, let's see if the application you created is acting as expected.

- Click on the  button of the "Debug programs" window to bring up a register window:



Live data is shown, allowing you to monitor your running application. The tabs at the top of the window allow you to select different types of registers to view. In the case of this application, all the registers you see are connected to I/O channels. (Note that on an EtherLogic controller, the Universal

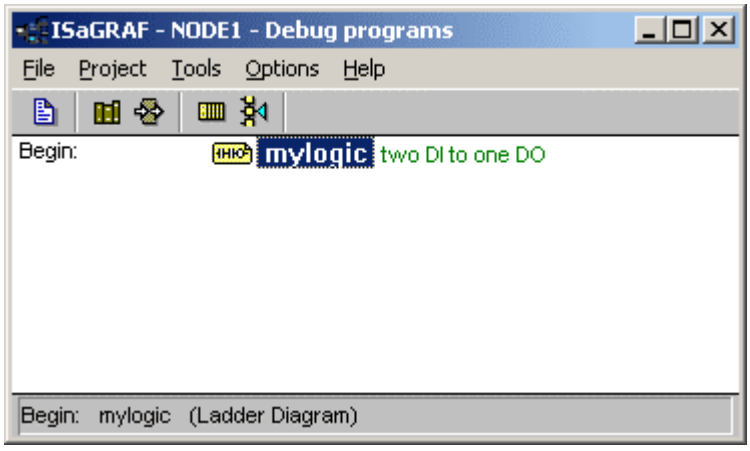
Inputs show up as both analog inputs and digital inputs in the register window -- see the EtherLogic Hardware Reference Guide for details.)

- You should be able to connect signals to the input terminals on the controller and have the corresponding value appear in the register window. To turn on DI 9 or DI 10 on the EtherLogic, apply 12 – 24 VDC to the corresponding terminal and common.

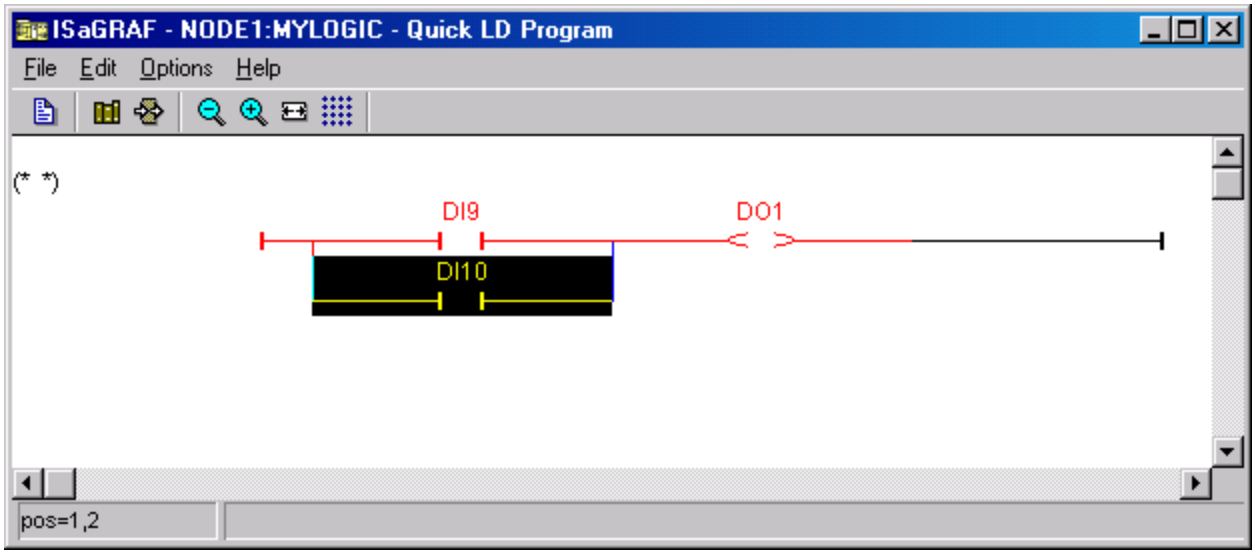
The operation of digital output 1 should reflect the logic that you put into ISaGRAF:

DI 9	**DI 10**	**DO 1**
FALSE	FALSE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
TRUE	TRUE	TRUE

- Another way to view the running logic of your application is to double click on the  icon corresponding to the logic in the "Debug programs" window:



- This will bring up a "live" window that shows the current state of the logic:



- Try exercising the logic by changing the state of the digital inputs. You should see the corresponding change shown in the window.

Now let's set up the controller to do some simple communications.

Modbus Setup

We will now add the Modbus protocol setup to our sample application.

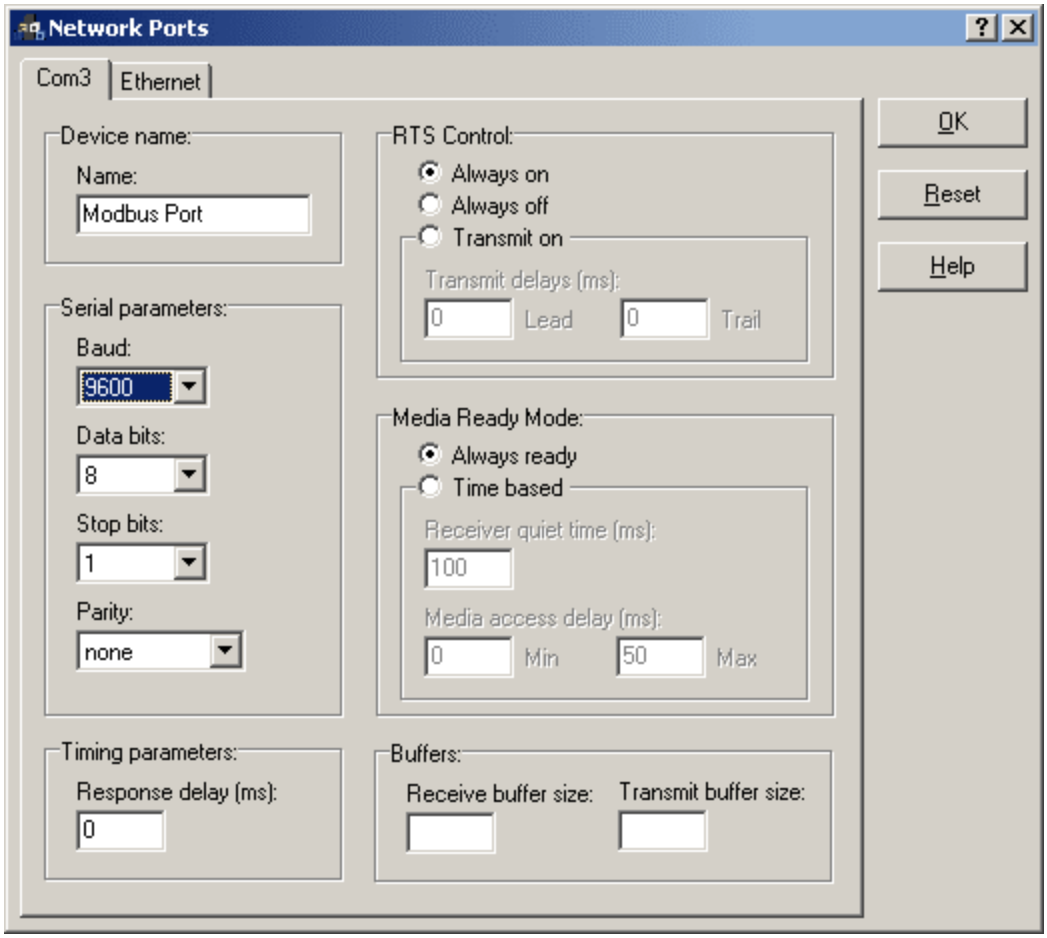
With Modbus protocol, there are "masters" and "slaves." There is only one master on a given Modbus network. The master originates messages to the slaves and the slaves respond. Slaves do not originate messages.

We are going to make our controller a slave on the "Com3" serial port.

First let's define the characteristics of the port:

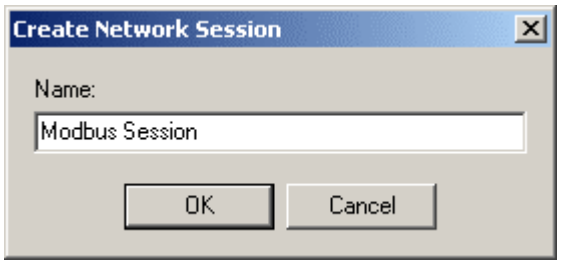
- Make sure that "Node1" is selected in the ScadaBuilder project window so that we can access its setup parameters. You should see a "target" icon  next to the Node.
- Choose "Setup" | "Network Ports" from the menu.
- If the "Com3" tab is not already selected, click on it to select settings for the Com3 serial port.
- Type the name as shown below for the port. Each Network Port that that you want to use must be given a name.
- Select a baud rate of 9600.

- Leave all the other settings at their default as shown below, then click "OK."

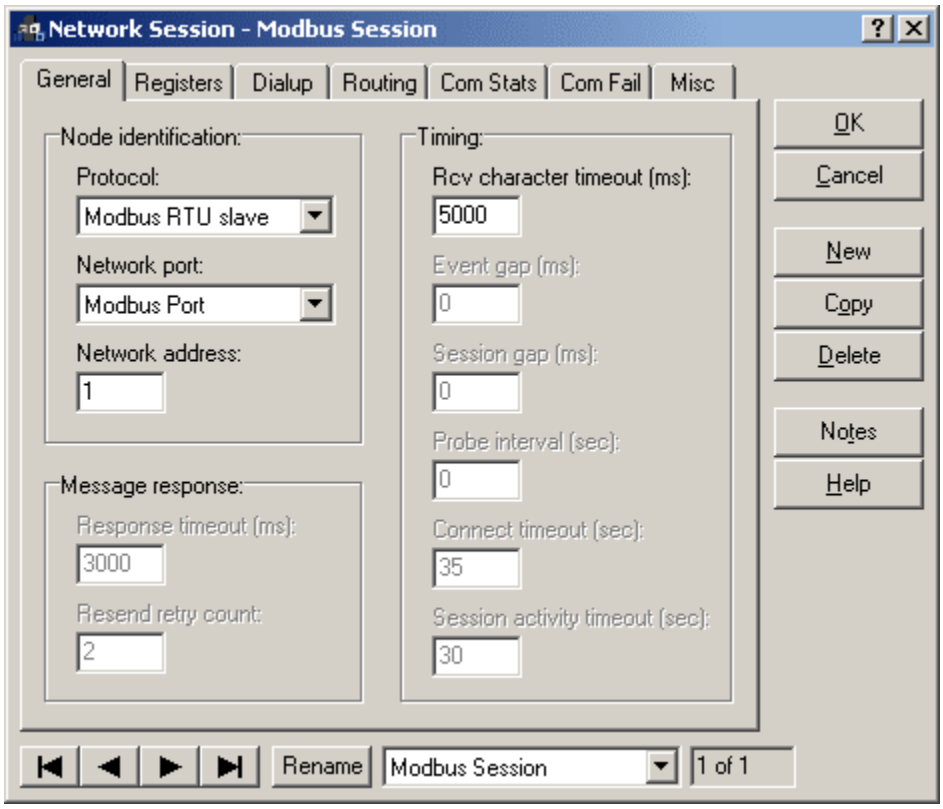


Next, you will create a Network Session. A Network Session specifies a protocol and related parameters. Network Sessions are applied to a specific Network Port. That makes the protocol run on the port. For this sample application, you will create a Network Session for the Modbus RTU Slave protocol.

- Choose "Setup" | "Network Sessions" from the menu.
- Enter the name as shown below, then click "OK":



- Set all the parameters as shown below on the "General" tab (you do not need to change settings on any of the other tabs):



  The "Network Address" is used to unique identify the slave. When the master has a message to send to the slave, it uses this address.

- Click "OK" to accept your changes.

You have now completed the Modbus slave setup.

- You should be able to communicate with the controller using a Modbus master (such as a SCADA or MMI software package running on a PC). Set the communications parameters to match the settings on Com3 of the controller (9600 baud, Modbus address 1). Use a null-modem cable between a test PC and Com3 of the controller.

  The I/O should be available in the following Modbus registers:

  - digital inputs -- starting at Modbus register 104001
  - digital outputs -- starting at Modbus register 004101
  - analog inputs -- starting at Modbus register 304201
  - analog outputs -- starting at Modbus register 404301

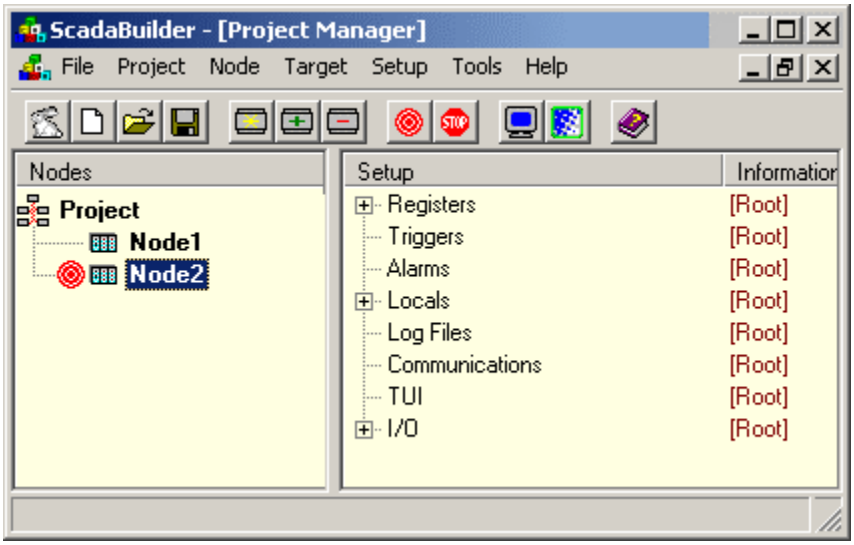
Now that you are done creating and testing a specific application, let's step back a little and talk about some general ScadaBuilder concepts that you will need to know as you venture off on your own.

ScadaBuilder Concepts

Projects and Nodes

ScadaBuilder lets you work with "Projects" and "Nodes." A ScadaBuilder Project is a collection of one or more Nodes. Each Node represents one physical ICL controller and its corresponding ScadaBuilder setup.

The example Project below has two Nodes ("Node1" and "Node2"):



To work on a specific Node, you must first click on it to select it. The currently selected Node has a "target" icon to the left of its name. In the example above, "Node2" is selected.

The options under the "Setup" menu apply to the selected Node. If a Node is not selected, you can't modify any Node setup items (the items of the "Setup" menu will be disabled).

Setting up a Node consists of creating and modifying configuration items using the "Setup" menu.

Some operations can be performed using the toolbar buttons at the top of the Workbench window. To see what the function of each button is, move your mouse pointer over the button of interest and wait a moment for the handy "tool-tip" to appear.

Setup items are shown on the right side of the Project Manager window. You can expand/contract a branch of the tree by clicking on the "+" and "-" symbols. To view or edit a setup item, double-click on it in the tree view. If no setup items exist, you may create one by double-clicking the corresponding tree root or using the "Setup" menu. A newly created Node will contain mostly the roots of the setup tree, as shown above.

To access the ISaGRAF tools for developing control logic, select a node and click the ISaGRAF  button or choose "Tools" | "ISaGRAF Workbench" from the menu.

Registers

ScadaBuilder uses "registers" to store information. Registers can store different types of information, such as integers, floating point/real numbers and on/off Boolean values. Values can be written to registers from various sources, including analog inputs, digital inputs and communications protocols. Likewise, registers can be read by analog outputs, digital outputs and communications. Registers are handy little places to store things and are essential to everything in ScadaBuilder -- without them, you can't do anything.

I/O Mapping

The link between registers and physical inputs and outputs (I/O) is created though "I/O mapping." When a register is mapped to an input, the input state/value is automatically stored in the associated register. When a register is mapped to an output, the value in the register is automatically reflected on the output.

I/O Scaling

ScadaBuilder allows you to scale analog inputs and outputs to engineering or "real world" units. For instance, an analog input could be scaled to represent flow in gallons per minute, liters per minute or just about any other unit of measure you can dream up.

What the Quick Project Feature Does

When you create a Project for a fixed I/O count controller or RTU (such as an EtherLogic or ScadaFlex Plus) using the "Quick Project" feature, a Project with one Node is created. In the Node setup, I/O registers are automatically created and mapped to the appropriate I/O channels. An I/O Scaling entry is created for each supported analog I/O mode, and the default Scaling entry is applied to each analog I/O channel.

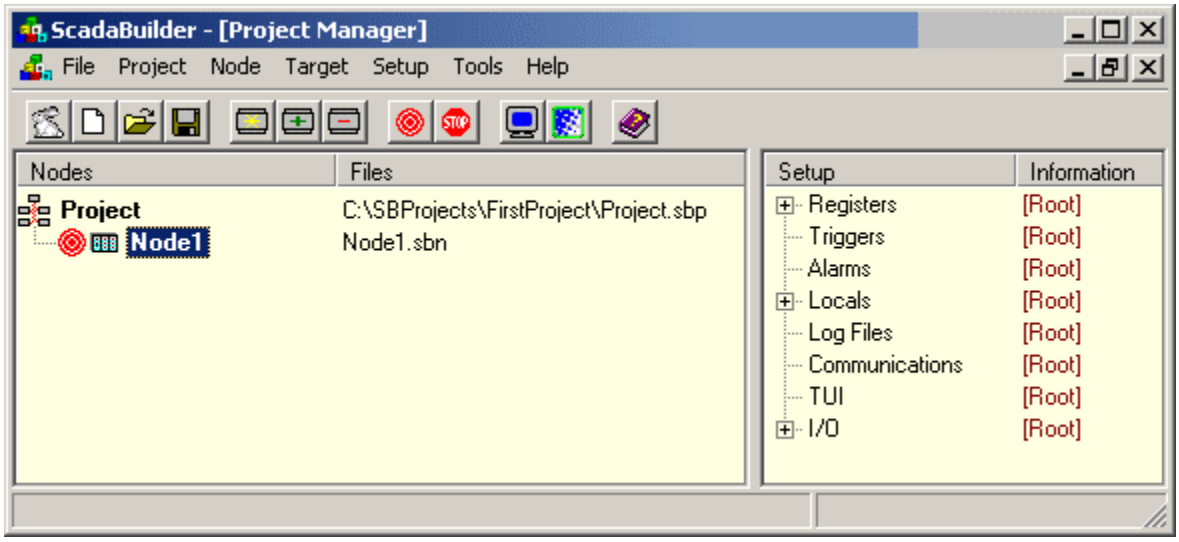
The Quick Project feature (or "ScadaBunny," as we affectionately call it sometimes!) can be run by clicking on the  button on the toolbar.

Now, armed with some new concepts, let's take a tour around the application you have created in order to more fully understand it.

Exploring the Setup

Let's take a moment to explore some of the items that the Quick Project feature created for us in our sample application.

- Click on "Node1" in the Project window, as shown below, to make sure it is selected:

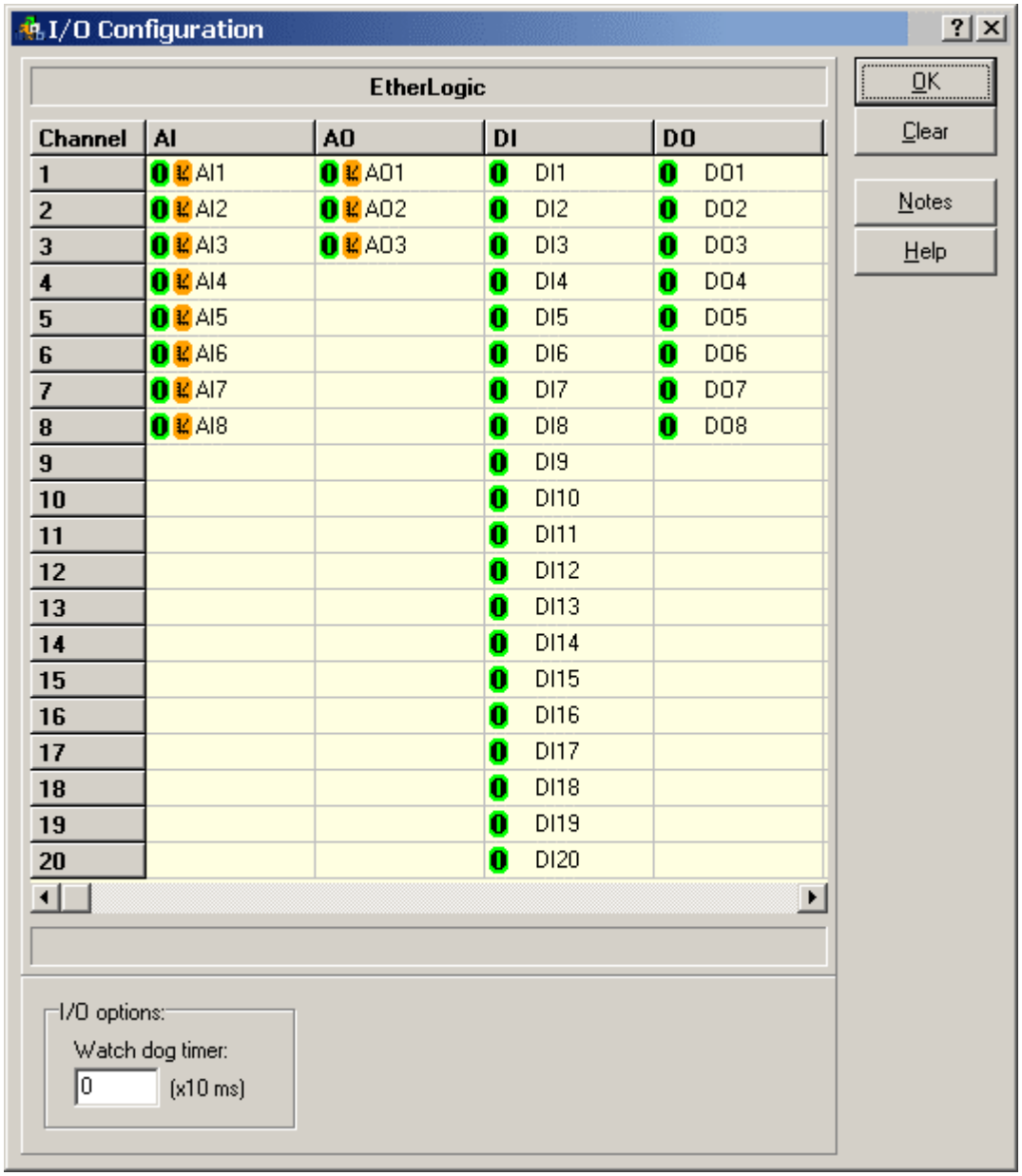


Registers

- Now click on the "+" sign to the left of "Registers" (in the right-hand window pane), to expand the register tree. Click on the "+" signs next to the "Boolean" and "Integer" branches as well to fully expand that branch of the tree. (Feel free to resize the window and move the dividing line that separates the two panes of the window.) The items listed are register names, such as "DI1" and "AI5." You will see that a number of registers have been automatically created for you -- one for each I/O channel. (If you wanted to change any of the register definitions, you would double-click the register type of interest, then double-click the specific register of interest when the ISaGRAF register window appears.)

I/O Configuration

- Next, expand the "I/O" section of the tree, then double-click "Configuration" to open the following window:



This window shows (and allows you to change) how I/O channels are mapped to registers. The columns show the I/O types, and the rows show the channels. For instance, the register "AI1" is mapped to analog input 1. We just picked simple register names -- you could call them something more suited to your application (like "Pressure" or "Flow").

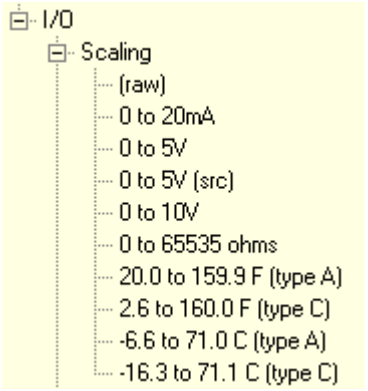
The icon indicates that a given channel has been mapped to a register. If the icon were present, this would indicate that the channel is not mapped. If a channel has been scaled, the icon appears.

To change the mapping or scaling, click on the channel you are interested in and then click the "Map" or "Scale" button that appears on the right side of the window.

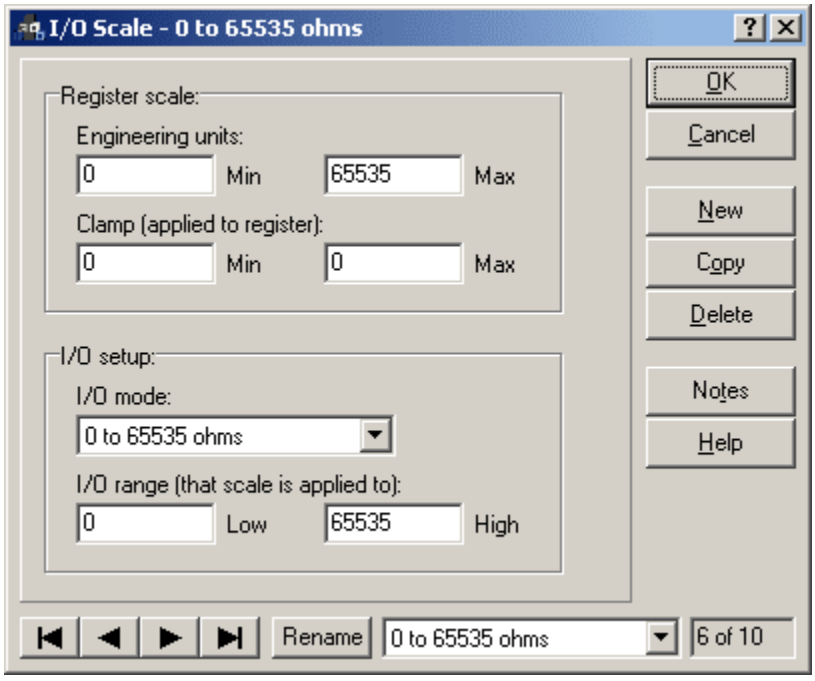
Scaling

Now let's take a closer look at scaling.

- Close the I/O Configuration window.
- Click the "+" sign next to the "Scaling" entry in the setup tree. You will see the following list of scaling entries:



- One Scaling entry has been automatically created for each analog I/O mode supported by the EtherLogic. The names are arbitrary, but have been chosen to indicate the mode.
- Double-click on the "0 to 65535 ohms" entry to see its parameters:



- Just for the fun of it click on the ? button, then click on the "Max" parameter. You will see a help window appear for that parameter. Almost every window has this feature -- use it to your advantage to find out what a particular parameter means.

  This Scaling entry is a simple 1:1 type -- the units that come out of the EtherLogic are already in Ohms when 0 to 65535 ohm mode is selected (with the "I/O mode" setting). "What use is it then?" you might understandably ask. Although the actual scaling in this case is not very exciting, it turns out that the Scaling entry also selects the operational mode of the I/O channel -- if we did not attach a Scaling entry to a channel, it would by default operate in "raw" mode, which would give us numbers in the range of 0-1023 (10 bits) for EtherLogic. Also, these Scaling entries where created in order for you to use as a starting point -- you can easily change them to suit your needs. For example, if you wanted the engineering units to be 0 to 100 percent, you would plug those numbers in to the "Engineering units" "Min" and "Max" parameters. You could also click the "Rename" button to choose a more meaningful Scaling entry name for your application (such as "Valve opening - percent").

  If you want to create another Scaling entry with the same mode and different engineering units or other parameters, use the "Copy" button. Use the "New" button to create a Scaling entry "from scratch."

  That completes our tour of ScadaBuilder. There are many more features that we haven't touched on, but you should know enough now to at least be able to get started. Feel free to explore some more on your own, and don't forget about the ? button to access parameter-specific help. Help is also available from the "Help" menu and many windows have a "Help" button.

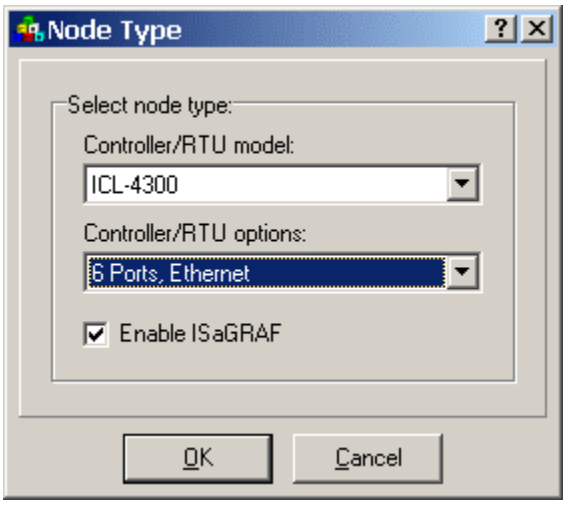
Appendix A - Adding Ethernet Support

There are two ways that a project node can be selected for Ethernet support.

- During project creation with the "Scada Bunny", the "Node | Type..." menu selection can be used to select a Controller/RTU option that includes Ethernet.
- If the project has been created without Ethernet, Ethernet can be added using the "Node | Options..." menu selection can be used to select a Controller/RTU option that includes Ethernet.

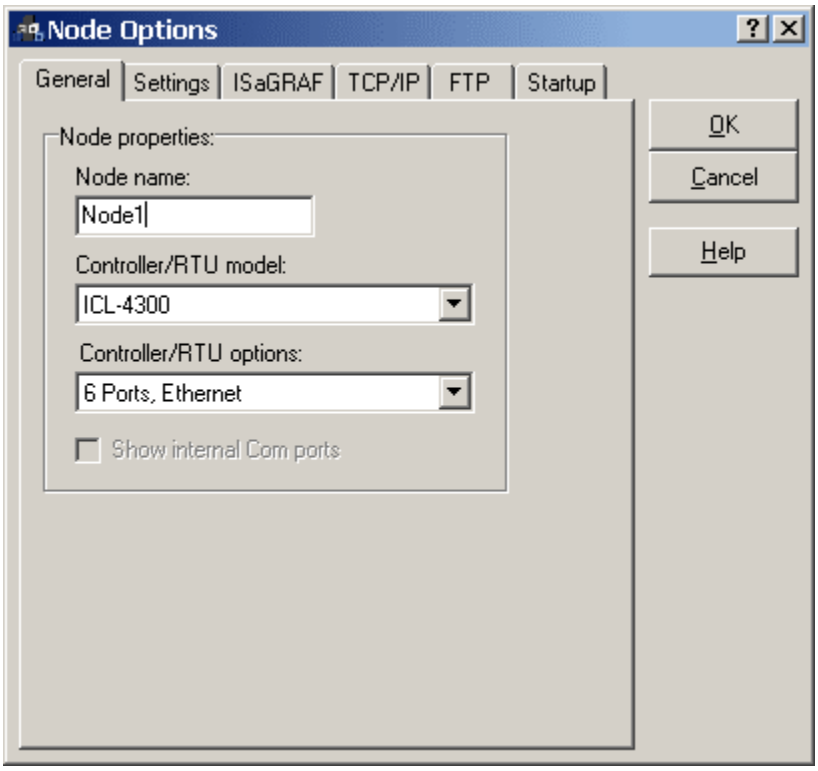
To provide include Ethernet support during the creation of the project, select the appropriate "Controller/RTU options" when ScadaBuilder prompts for the Node Type. An example of the Node Type window for the ICL-4300 is shown below

- Click OK to accept the options.



- To configure an existing project for Ethernet support, Click on the Node and choose "Node | "Options...".

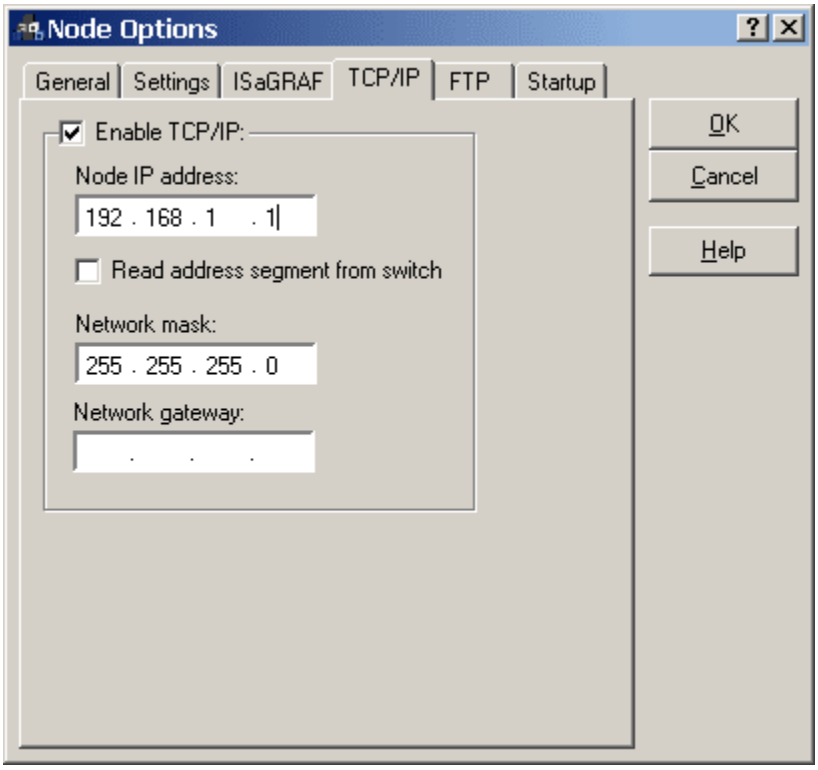
The following window will appear.



- From the "General" Tab, verify that the "Controller/RTU Options" has a controller options selected that include Ethernet. It may be necessary to change this selection to a model that includes Ethernet.

Once the Node is configured as an Ethernet capable device, the TCP/IP must be configured.

- Select the "TCP/IP" Tab from the Node Options window as shown below.



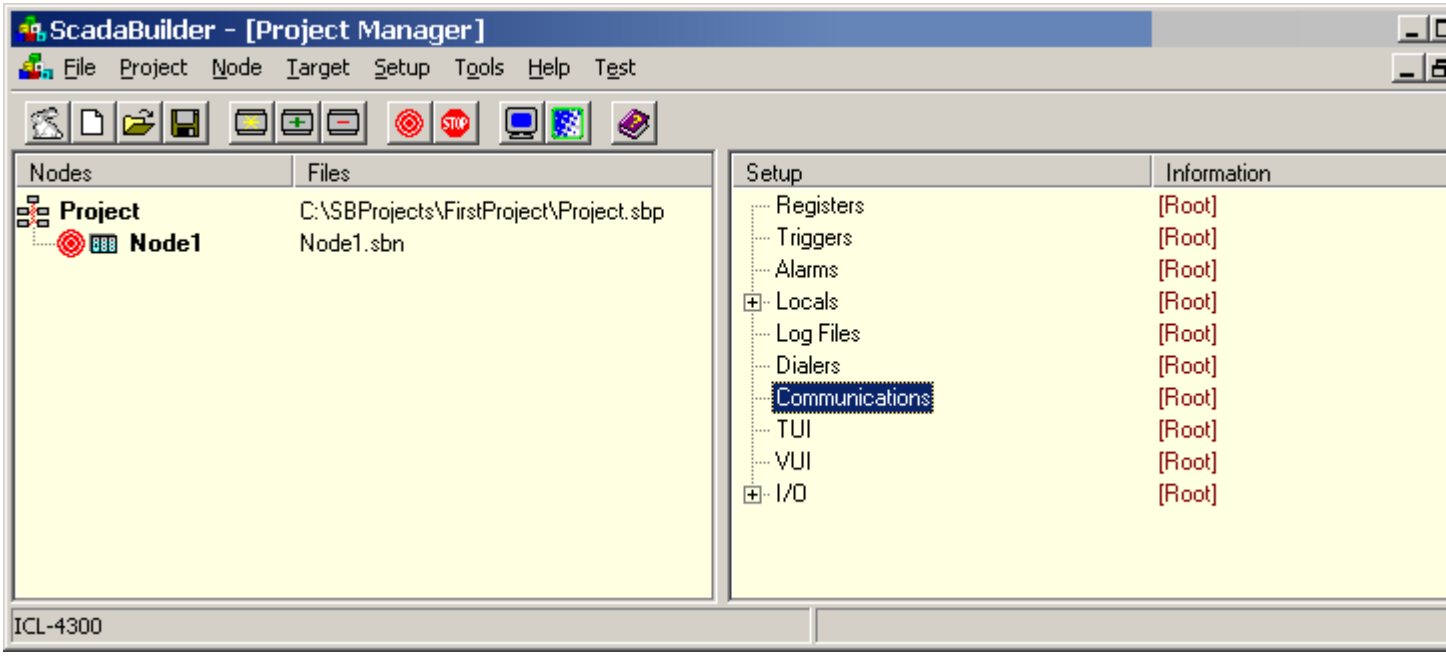
- Click on the "Enable TCP/IP" check box and verify that a " ✓ " appears in the check box. This will cause the IP configuration entries to become enabled.

The Node IP address and Network subnet mask must be configured. At this point it may be necessary to consult your IT department for a compatible IP configuration for your LAN.

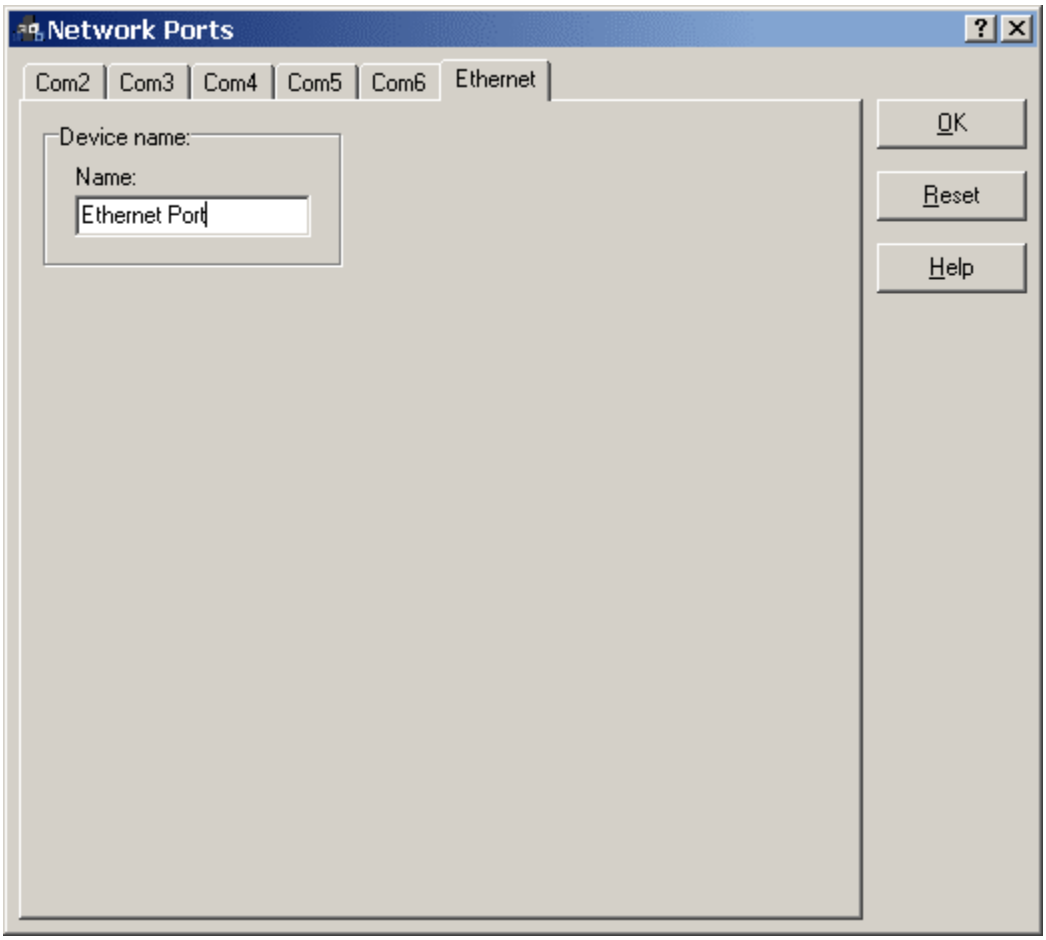
- Enter an IP address and Network mask for the ICL-4300.
- Click on the "OK" button to save the IP configuration.

Ethernet is configured and available for use as a network communication protocol.

To use the configured Ethernet protocol in a project, invoke the ScadaBuilder Project Manager window as shown below.



- Click on the Node and choose "Setup | "Network Ports...". Alternatively, you can double-click on "Communications" in the project tree-view. The following window will appear.



- Select the "Ethernet" tab and enter a device name for the port.
- Click on "OK" to accept the entry

Congratulations, Ethernet is configured and available for use in your project.