IODISPLAY™ USER'S GUIDE

FORM 1302-060608—JUNE, 2006

OPTO 22

43044 Business Park Drive • Temecula • CA 92590-3614
Phone: 800-321-OPTO (6786) or 951-695-3000
Fax: 800-832-OPTO (6786) or 951-695-2712
www.opto22.com

Product Support Services
800-TEK-OPTO (835-6786) or 951-695-3080
Fax: 951-695-3017
Email: support@opto22.com
Web: support.opto22.com

ioDisplay User's Guide
Form 1302-060608—June, 2006

The information in this manual has been checked carefully and is believed to be accurate; however, Opto 22 assumes no responsibility for possible inaccuracies or omissions. Specifications are subject to change without notice.

Opto 22 warrants all of its products to be free from defects in material or workmanship for 30 months from the manufacturing date code. This warranty is limited to the original cost of the unit only and does not cover installation, labor, or any other contingent costs. Opto 22 I/O modules and solid-state relays with date codes of 1/96 or later are guaranteed for life. This lifetime warranty excludes reed relay, SNAP serial communication modules, SNAP PID modules, and modules that contain mechanical contacts or switches. Opto 22 does not warrant any product, components, or parts not manufactured by Opto 22; for these items, the warranty from the original manufacturer applies. These products include, but are not limited to, OptoTerminal-G70, OptoTerminal-G75, and Sony Ericsson GT-48; see the product data sheet for specific warranty information. Refer to Opto 22 form number 1042 for complete warranty information.

Table of Contents

Chapter 7: Using Animated Graphics 7-1

CHAPTER 1

Welcome to ioDisplay

Welcome to ioDisplay™, Opto 22's human-machine interface (HMI), alarming, and trending software for Microsoft® Windows® operating systems. ioDisplay works with Opto 22 control programs (or *strategies*) running on Opto 22 industrial controllers.

ioDisplay lets you easily create graphical, on-screen operator interfaces to monitor and manage control applications running on Opto 22 industrial control engines. With ioDisplay, you can present real-time control engine information to the operator, set alarms to notify the operator of changing data, visually track trends in the data using graphs, and securely log data to local or remote computers. Additionally, you can configure the interface to allow a specified operator or group of operators to change values such as alarm thresholds.

ioDisplay Basic and ioDisplay Professional

Two versions of ioDisplay software are available: **Basic** and **Professional**. ioDisplay Basic provides all the HMI functions that are listed above and documented in this user's guide. ioDisplay Professional adds several capabilities that are important if you need to do the following:

- Import projects created in Opto 22's OptoDisplay application
- Take advantage of redundant Ethernet networking features in Opto 22 SNAP PAC controllers.
- Access Scratchpad variables on a SNAP Ultimate brain, SNAP-LCE controller, or SNAP PAC controller.
- Have an ioDisplay project connect to an Opto 22 controller that is running an OptoControl strategy.

This user's guide covers both ioDisplay Basic and ioDisplay Professional.

The "Basic" icon indicates features or functions that apply only to ioDisplay Basic.

The "Professional" icon indicates features or functions that apply only to ioDisplay Professional.

About This Guide

This user's guide teaches you how to use ioDisplay, including designing an ioDisplay project, configuring and connecting an Opto 22 control engine, and monitoring information in your operator interface.

This guide assumes that you are already familiar with Microsoft Windows on your personal computer, including how to use a mouse, standard menus and commands, and how to open, save, and close files. If you are not familiar with Windows or your PC, see the documentation from Microsoft and your computer manufacturer.

Here's what is in this user's guide:

Chapter 1: Welcome to ioDisplay—This introductory chapter.

Chapter 2: Quick Start—A short lesson to get you up and running with an ioDisplay project as quickly as possible. You'll use a sample project to learn how to work with graphics, assign dynamic attributes, and run a project.

Chapter 3: What Is ioDisplay?—An introduction to ioDisplay, basic design and programming concepts, and ioDisplay controls and windows.

Chapter 4: Working with Projects—An explanation of what ioDisplay projects are, the files they're made of, and how they're organized.

Chapter 5: Configuring Control Engines and Tags—Detailed procedures on configuring control engines and I/O from an ioControl strategy for use in an ioDisplay project.

Chapter 6: Working with Graphics—Detailed steps for working with graphics—including assigning animation attributes—and the windows in which graphics appear.

Chapter 7: Using Animated Graphics—Covers how to assign dynamic attributes to on-screen objects to create an animated, real-time display of I/O information.

Chapter 8: Working with Trends—Explains how to create and configure graphs to track data from I/O points over time.

Chapter 9: Configuring Trigger-Based Events—Explains how to use historical logging, application launching, sounds, recipes, and how to change window states based on events.

Chapter 10: Using ioDisplay Runtime—Describes how to customize configurable Runtime features and what you'll see during a Runtime project session.

Appendix A: ioDisplay Troubleshooting—Gives tips for solving problems you may encounter while building and using your ioDisplay project.

Appendix B: ioDisplay Errors—Explains warnings and error messages you may see while running a program in ioDisplay Runtime.

Appendix C: ioDisplay Files—Lists all ioDisplay files located in the ioDisplay directory.

Appendix D: ioDisplay Menu Reference—Lists commands and other menu bar options.

ioDisplay Index—Provides an alphabetical list of key words and their page locations.

Document Conventions

The following conventions are used in this document:

- Italic typeface indicates emphasis and is used for book titles. (Example: "See the *ioControl User's Guide* for details.")
- Names of menus, commands, dialog boxes, fields, and buttons are capitalized as they appear in the product. (Example: "From the File menu, select Print.")
- File names appear either in all capital letters or in mixed case, depending on the file name itself. (Example: "Open the file TEST1.txt.")
- Key names appear in small capital letters. (Example: "Press SHIFT.")
- Key press combinations are indicated by hyphens between two or more key names. For example, SHIFT+F1 is the result of holding down the shift key, then pressing and releasing the F1 key. Similarly, CTRL+ALT+DELETE is the result of pressing and holding the CTRL and ALT keys, then pressing and releasing the DELETE key.
- "Click" means press and release the left mouse button on the referenced item. "Right-click" means press and release the right mouse button on the item.
- Menu commands are referred to with the Menu→Command convention. For example, "File→Open Project" means to select the Open Project command from the File menu.
- Numbered lists indicate procedures to be followed sequentially. Bulleted lists (such as this one) provide general information.

Other ioDisplay Resources

Documents and Online Help

To help learn and use ioDisplay, the following resources are provided:

- **Online Help** is available in ioDisplay, ioControl, and in most of the utility applications. To open online Help, choose Help→Contents and Index in any screen.
- ***ioDisplay User's Guide*** (this document)

Online versions (Adobe® Acrobat® format) of ioDisplay documentation are provided on the ioDisplay CD and are also available from the Help menu in ioDisplay. To view a document, select Help→Manuals, and then choose a document from the submenu.

ioDisplay and ioProject resources are also available on the Opto 22 Web site at www.opto22.com. You can conveniently access this and other sections of the Opto 22 Web site from ioDisplay's Help menu. Select Help→Opto 22 on the Web, and then select an online resource from the submenu.

Product Support

If you have any questions about ioDisplay, you can call, fax, or e-mail Opto 22 Product Support.

Phone: 800-TEK-OPTO (835-6786)
951-695-3080
(Hours are Monday through Friday,
7 a.m. to 5 p.m. Pacific Time)

Fax: 951-695-3017

Email: support@opto22.com

Opto 22 Web site: support.opto22.com

NOTE: Email messages and phone calls to Opto 22 Product Support are grouped together and answered in the order received.

When calling for technical support, be prepared to provide the following information about your system to the Product Support engineer:

- Software and version being used
- Control engine firmware version
- PC configuration (type of processor, speed, memory, operating system)
- A complete description of your hardware and operating systems, including:
  - types of I/O units installed
  - type of power supply
  - third-party devices installed (e.g., barcode readers)
- Specific error messages seen.

Installing ioDisplay

ioDisplay installation is easy and quick. Insert the ioDisplay CD in your CD-ROM drive, and the installation wizard should appear. If the wizard does not appear, start Windows Explorer and navigate to your CD-ROM drive. Double-click Setup.exe to begin installation.

NOTE: If you run ioProject applications in Microsoft Windows XP, make sure to use the Windows Classic theme. Otherwise, a Microsoft bug with how themes are handled may cause the system to crash.

If you have trouble installing ioDisplay or need 3.5-inch disks rather than a CD, contact Opto 22 Product Support at 800-835-6786 or 951-695-3080.

System Requirements

Installation Requirements

Here's what you need to install and run ioDisplay:

- A computer with at least the minimum processor required for your version of Microsoft Windows (1 GHz Pentium®-class or better recommended) and Ethernet capability
- VGA or higher resolution monitor (Super VGA recommended). Minimum size: 800x600 with small fonts.
- Mouse or other pointing device
- Installed Windows printer (optional)
- Microsoft Windows XP or Windows 2000® (with SP4) workstation operating system
- At least 128 MB RAM (256 MB recommended)

  If your ioDisplay project uses many basic trends, SuperTrends, or XY Plots, we strongly recommend adding RAM beyond the amount suggested here. See "Memory Requirements for SuperTrend Pens" on page 8-7 for more information on memory requirements.
- At least 50 MB available space on hard drive.

NOTE: If your are using an operating system such as Windows XP that supports multiple monitors, you can display the operator interface you create in ioDisplay on more than one monitor as long as all monitors have identical video cards.

CHAPTER 2

Quick Start

Introduction

The quickest way to become familiar with ioDisplay is by working through a simple example. Our example will use a cookie factory to show you how easy it is to use ioDisplay. You'll learn how to start ioDisplay, open and save a project, and assign an ioControl strategy to the project. Then you'll bring in a bitmap, add some animation attributes, and watch your project in action. We'll repeat this process to fine-tune the visuals and we'll end up with a final "working" cookie factory.

NOTE: If you can't access an Opto 22 control engine at the moment, you can still do everything in the Quick Start up to the point of running your project.

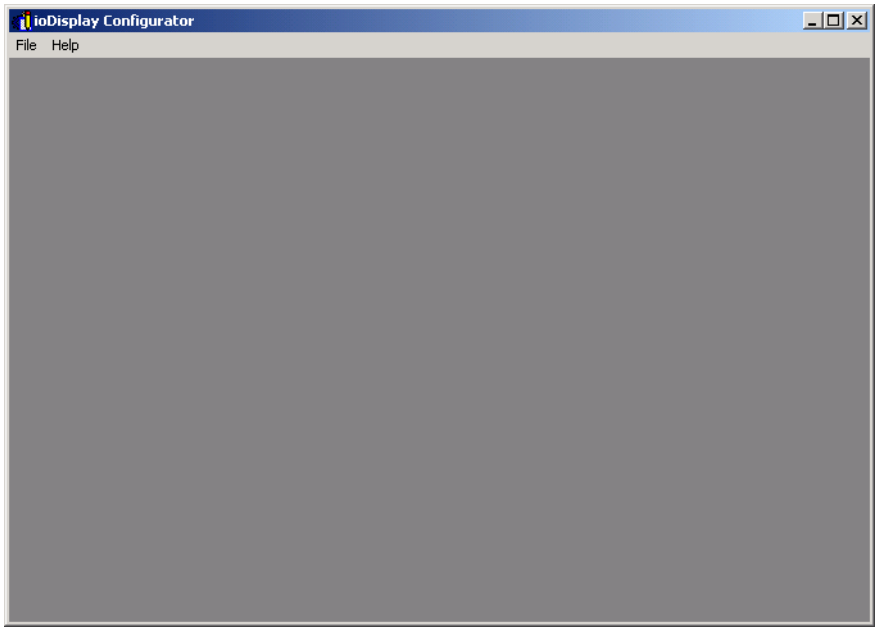
In This Chapter

Opening the Project

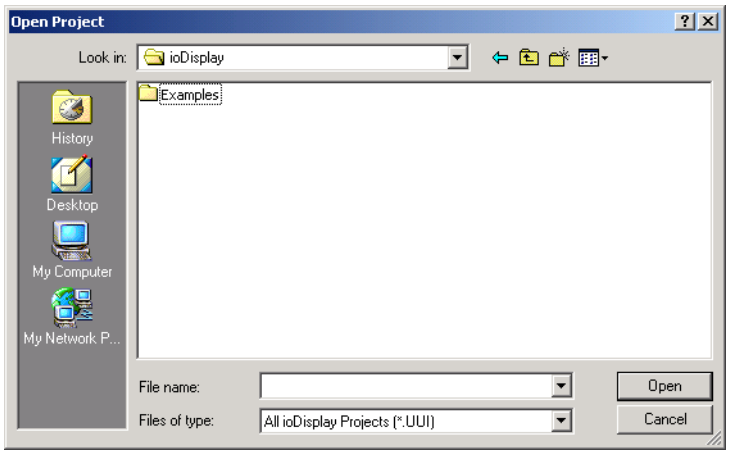
Let's start by opening our sample project. ioDisplay projects contain windows, graphics, and other information needed to produce an animated operator interface.

1. To start ioDisplay, click the Windows Start button and select Programs→Opto 22→ioProject Software→ioDisplay Configurator.

The ioDisplay main window opens:



2. Select File→Open Project to open the sample project.

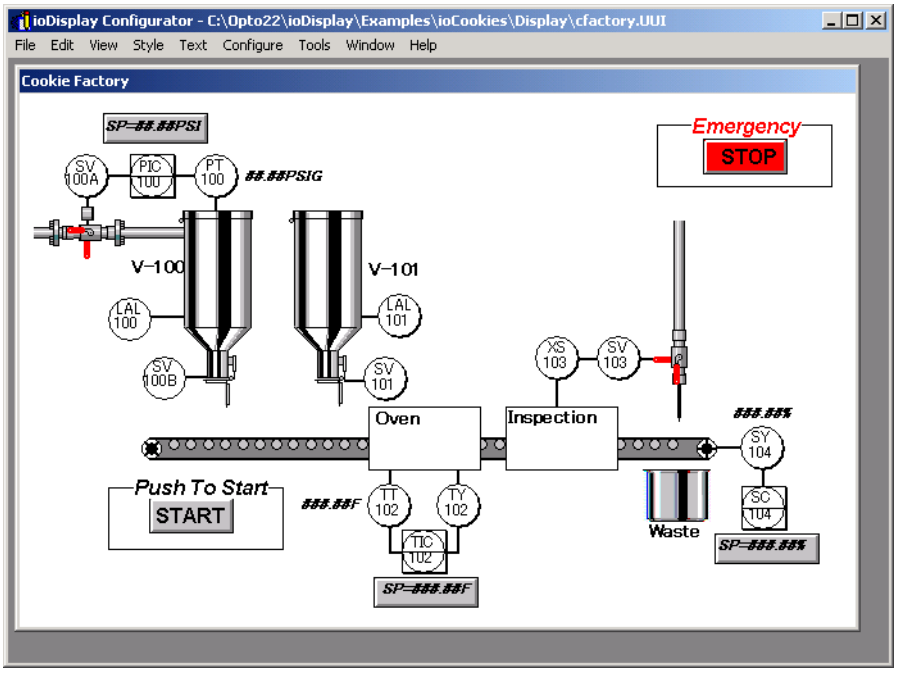


3. In the Open Project dialog box, navigate to Program Files\Opto22\ioProject Software\ioDisplayExamples.

4. In the Examples directory, double-click the ioCookies subdirectory, and then double-click the Display subdirectory to reach the project file we will use.

5. Double-click the project file cfactory.uui to open it.

The main window should look like the example shown below.



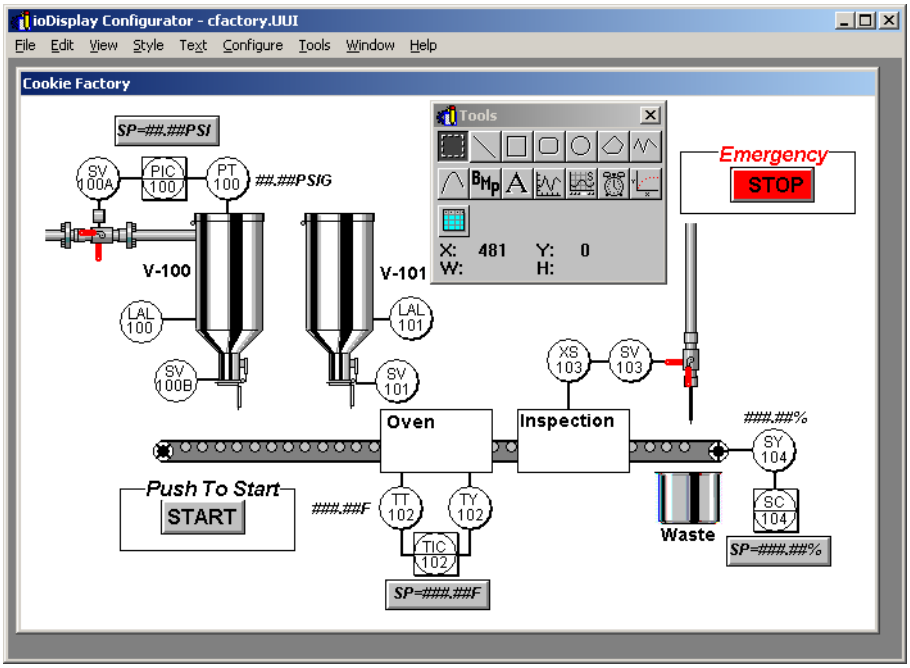
Examining the Project

This particular ioDisplay project models a simple cookie factory that uses the following components:

- A tank of pre-mixed cookie dough
- A tank of chocolate chips
- An oven
- A visual inspection station
- Some plant air to blow out bad cookies
- A conveyor belt to move material between the different components.

At the start of the process, a measured amount of dough is dropped onto the conveyor belt. The dough moves first under the chip tank to receive some chips, and then into the oven to be baked. The next stop is an inspection station, where rejected cookies are blown off the belt. The good cookies go to shipping. Should anything go wrong, we also have some alarms built in to stop the process if necessary.

The window for the cookie factory project is shown below.



This window is called a *draw window,* because in ioDisplay Configurator this is where you can create and position graphic objects and other elements for your operator interface. This is also where you assign animation characteristics, or *dynamic attributes,* to graphic objects. Floating on top of the window is a toolbox with all the tools you need to draw graphic objects.

Configuring a Control Engine

We'll start out by configuring a control engine so that our graphics are tied to actual values in an ioControl strategy. The strategy will be downloaded to your control engine later so we can actually see things running. We'll briefly go through the configuration process, which is covered in greater detail in Chapter 5, "Configuring Control Engines and Tags."

If the Control Engine Already Exists

If you already completed the Tutorial chapter in the *ioControl User's Guide,* a control engine named "Cookie Controller" should be available. Follow the instructions below to check if this control engine already exists.

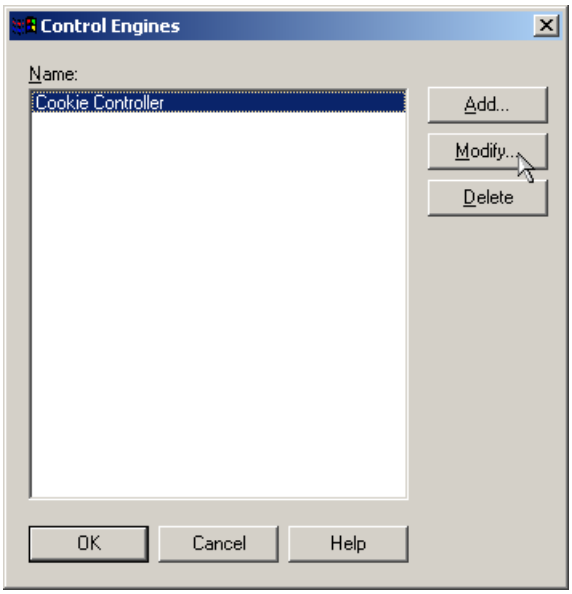
1. Choose the Configure→Control Engine(s) menu item.

   The Control Engines dialog box appears.

2. Do the following:

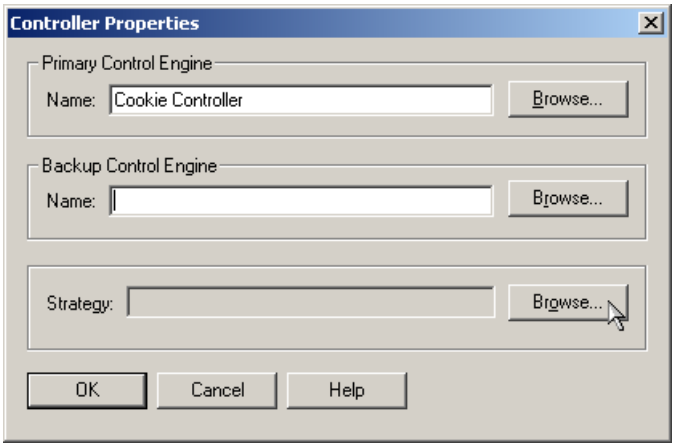
- If the control engine "Cookie Controller" is listed in this dialog box, continue with step 3 below.
- If the control engine "Cookie Controller" is *not* listed in this dialog box, follow the steps in "Adding a Control Engine" on page 2-6.

3. Select "Cookie Controller" in the Control Engines dialog box and click Modify.

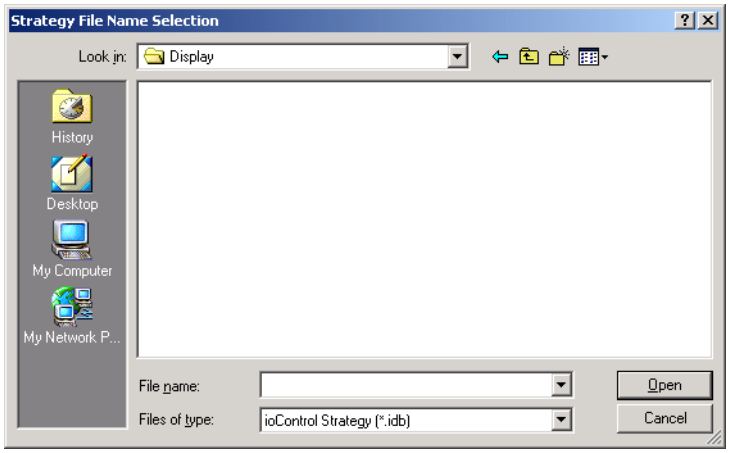


4. In the Controller Properties dialog box that appears, click Browse in the Strategy section.

NOTE: This dialog box is slightly different in ioDisplay Professional, but the Strategy section and corresponding Browse button are the same.



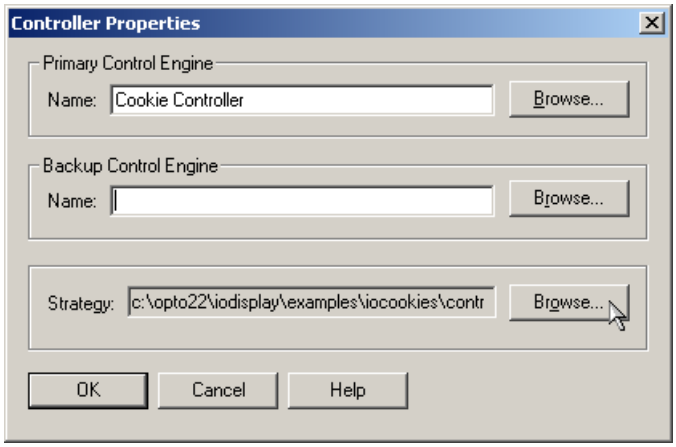
The Strategy File Name Selection dialog box appears. This dialog box is much like the Open Project dialog box we used to find our cookie factory project.



5. Change directories to Program Files\Opto22\ioProject Software\ioDisplayExamples\ioCookies\Control directory.

6. Select the cookies.idb file, and then click Open.

   The Controller Properties dialog box should now show both the name of the control engine and the ioControl strategy that will be used.

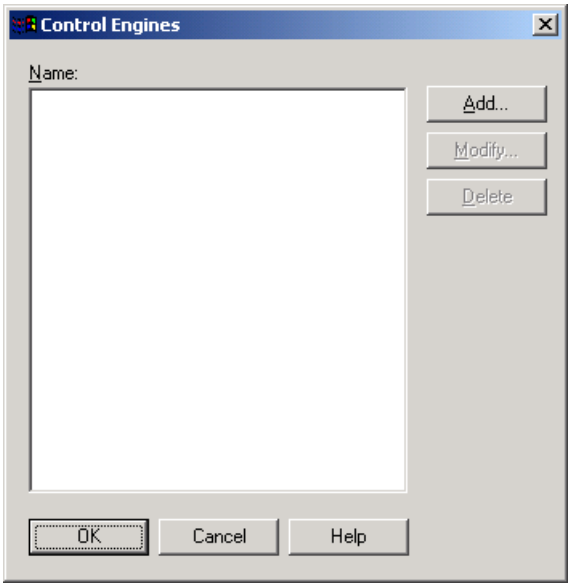


The ioDisplay project is now configured to use your control engine for this Quick Start exercise. Skip the next section, "Adding a Control Engine," and continue with the steps in "Adding a Dynamic Attribute" on page 2-11.

Adding a Control Engine

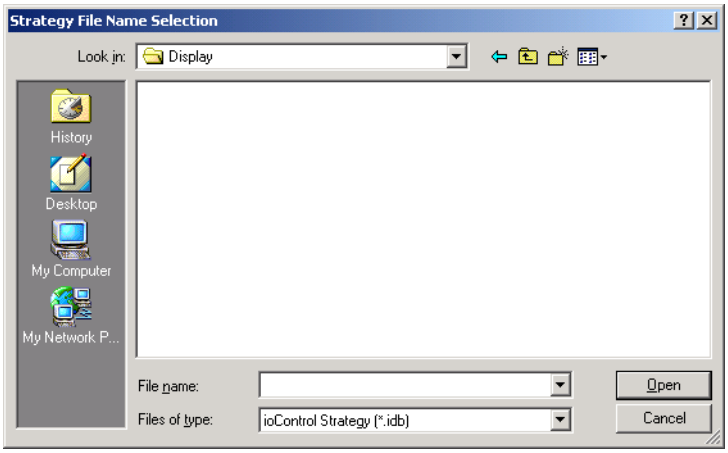
1. Choose the Configure→Control Engine(s) menu item.

The Control Engines dialog box appears:



2. Click the Add button.

   The Strategy File Name Selection dialog box appears. This dialog box is much like the Open Project dialog box we used to find our cookie factory project.

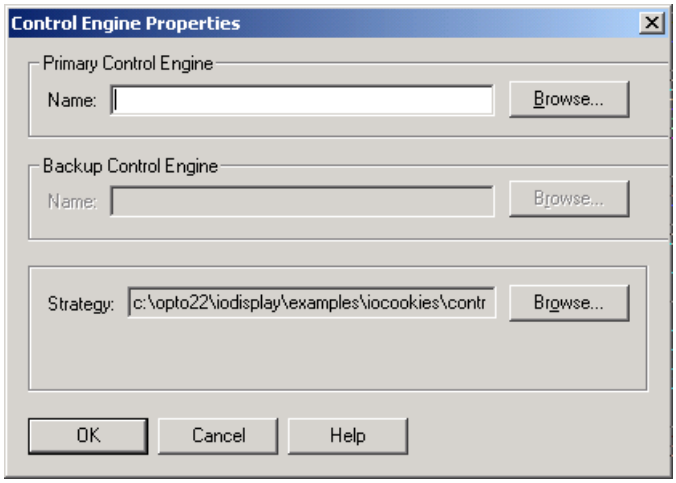


3. Change directories to Program Files\Opto22\ioProject Software\ioDisplayExamples\ioCookies\Control directory.

4. Select the cfactory.idb file, and then click Open.

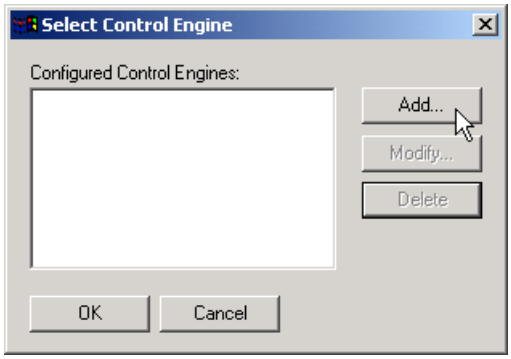
   The Control Engine Properties dialog box appears. Notice that the ioControl strategy you just picked is shown in the Strategy field.

NOTE: This dialog box is slightly different in ioDisplay Professional, but the Strategy section and corresponding Browse button are the same.



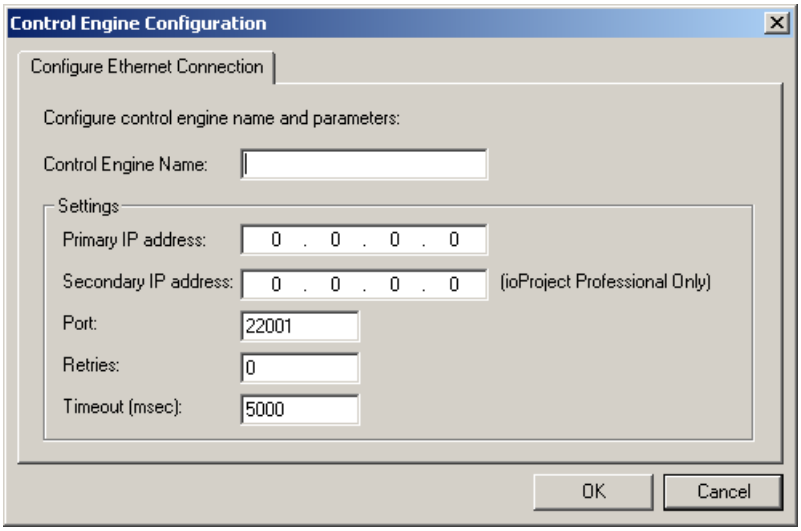
5. To define your control engine in the Primary Control Engine field, click the Browse button in the Primary Control Engine group.

   The Select Control Engine dialog box opens:



6. Click Add to add a control engine.

The Control Engine Configuration dialog box appears:



7. Enter Cookie Controller as the control engine name.

The name can contain letters, numbers, spaces, and most other characters except colons and square brackets. Spaces cannot be used as first or last characters.

8. Enter the control engine's IP address.

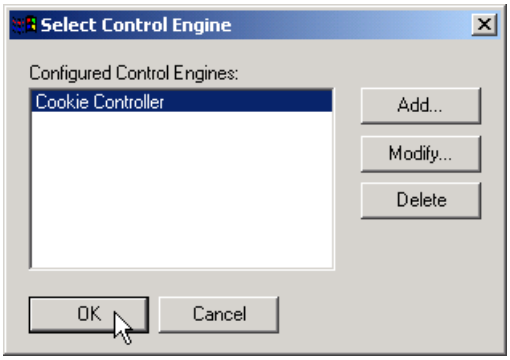
On a control engine the IP address assigned to the device is usually written on a sticker on the side of the unit. If an IP address has not been assigned to the control engine, see the user's guide for that device for configuration instructions.

Pro

You can also configure a secondary IP address for the control engine, where the second IP address is another control engine running the same strategy. If a SNAP PAC control engine is used, the second IP address can be the controller's second Ethernet interface. See the *Redundancy Technical Note* (Opto 22 form 1597) for more information on using SNAP PAC controllers to segment Ethernet networks.

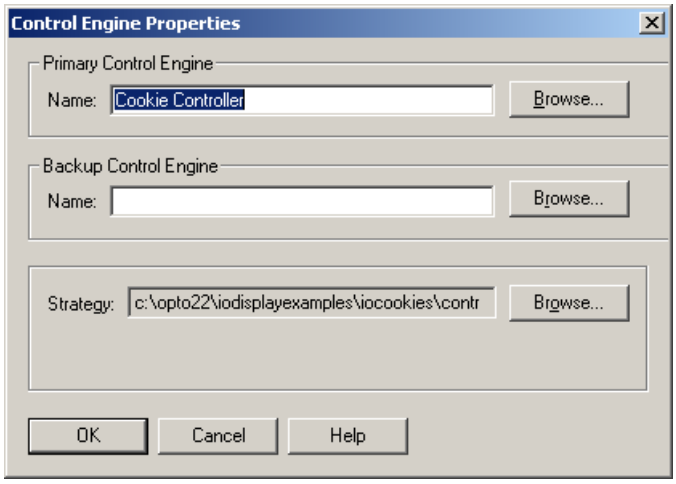
9. Make sure that you have not changed the values in the Port, Retries, and Timeout, and then click OK.

The newly configured control engine appears in the Select Control Engine dialog box.



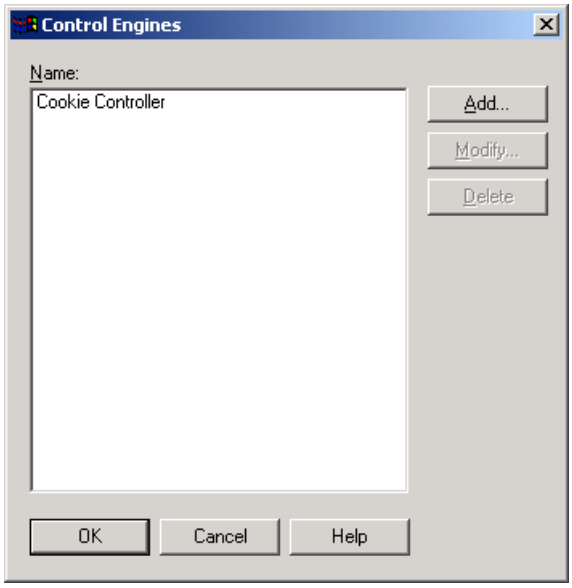
10. Click the new Cookie Controller control engine to select it, and then click OK.

The Control Engine Properties dialog box appears with the new control engine listed in the Primary Control Engine group.



11. Click OK.

The Control Engines dialog box appears with the new control engine Cookie Controller.

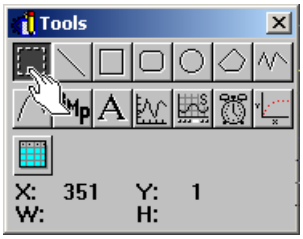


12. Click OK to finish configuring the control engine.

Adding a Dynamic Attribute

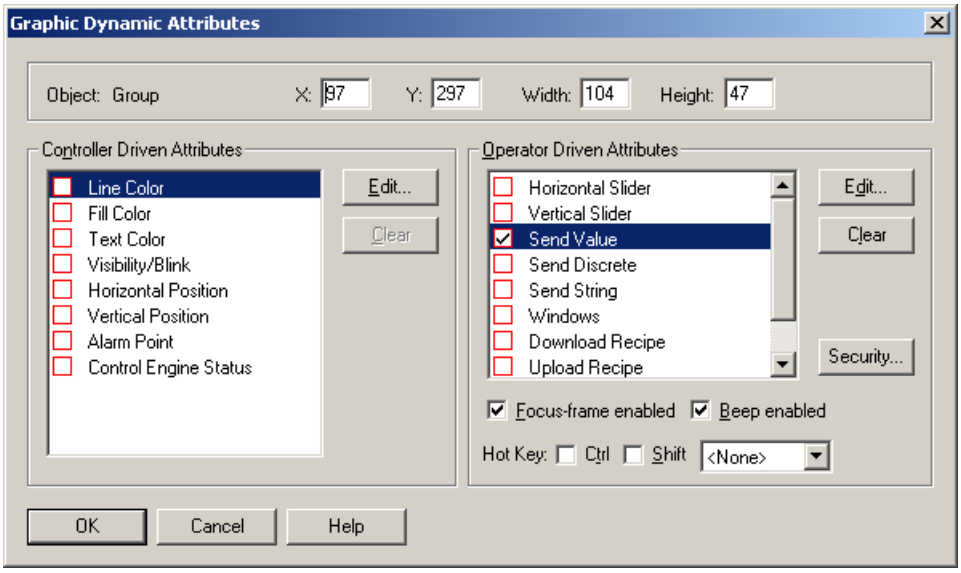
Let's assign an attribute to the Start button that will start the cookie factory display.

1. Click the Select tool in the ioDisplay Configurator toolbox as shown below.



2. Double-click the Start button in the Cookie Factory window.

   Some small black boxes called *sizing handles* appear around the button. They indicate that the button is currently selected. The Graphic Dynamic Attributes dialog box also opens.

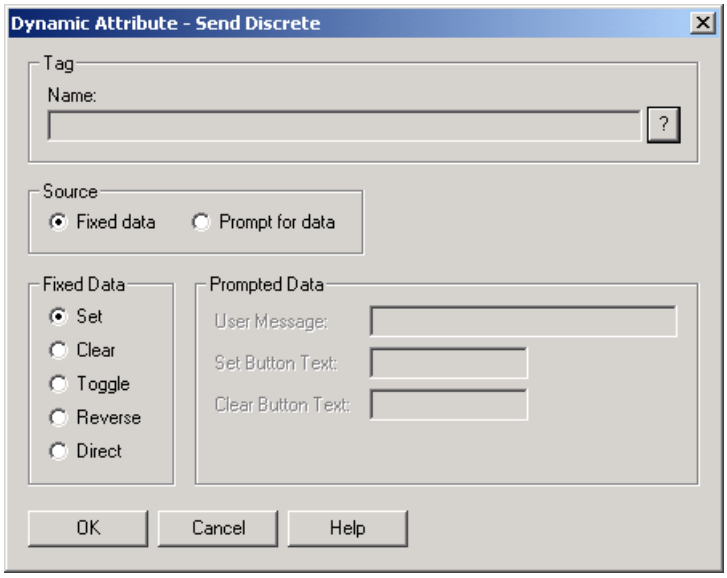


   Notice that the dialog box has two separate groups of attributes: Controller Driven Attributes and Operator Driven Attributes. *Control Engine-driven attributes* are attributes that are driven by tag values from the ioControl strategy we assigned to the project. *Operator-driven attributes* are driven by an operator's interaction with a graphic object in ioDisplay.

   We're going to choose the operator-driven attribute Send Discrete to send a discrete value to the control engine. The ioControl strategy interprets the value as a signal to start the cookie factory.

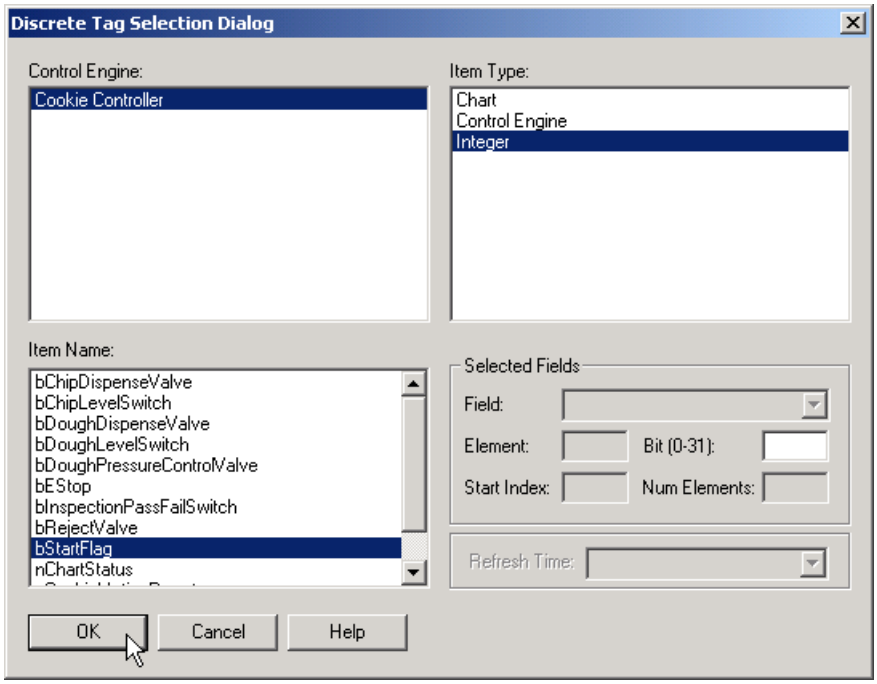
3. In the Graphic Dynamic Attributes dialog box, select Operator Driven Attribute→Send Discrete, and then click Edit.

The Dynamic Attribute - Send Discrete dialog box opens:



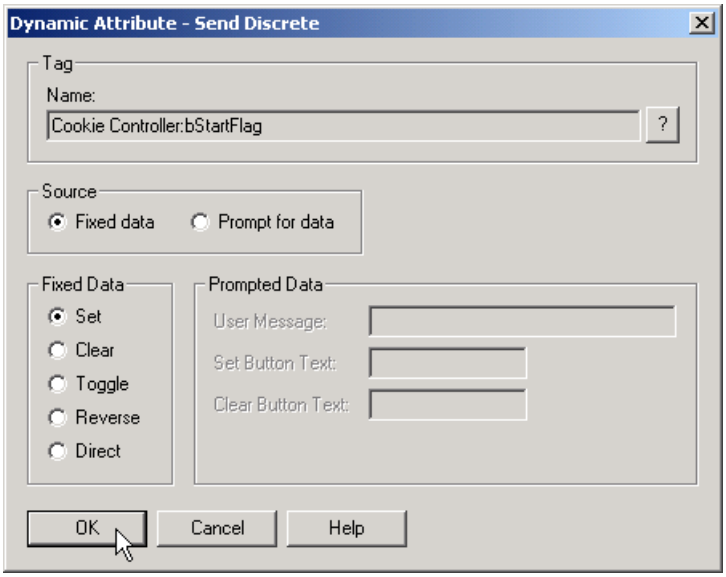
4. Click the Tag Selection button  to configure the tag we want to connect to in the ioControl strategy.

   The Tag Selection dialog box appears. Notice that the Cookie Control Engine is highlighted in the Control Engine list.



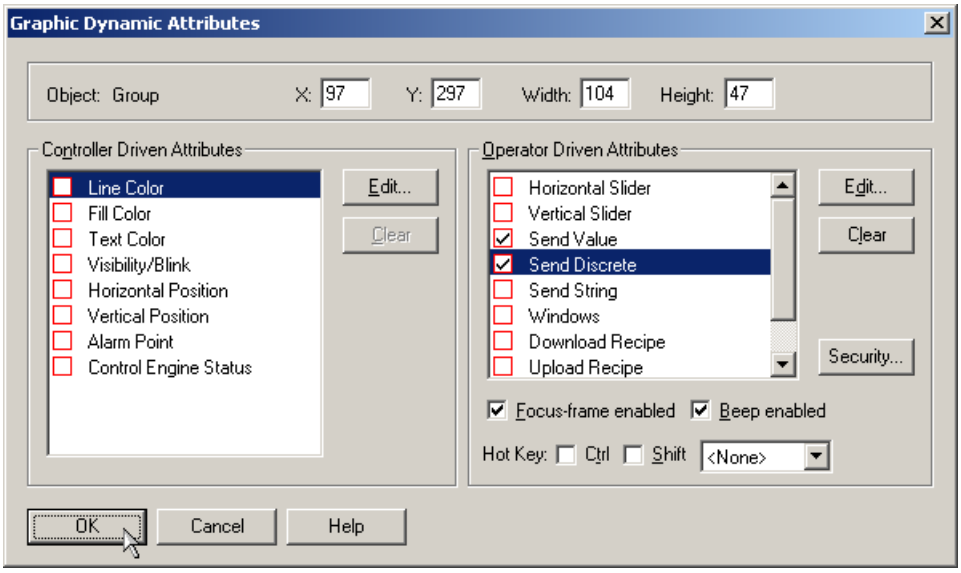
5. Select Integer in the Item Type group and bStartFlag in the Item Name group, and then click OK.

The Dynamic Attribute - Send Discrete dialog box appears with the new tagname listed in the Tag group.



6. Click OK.

   The Graphic Dynamic Attributes dialog box appears with a check mark next to Send Discrete in the Operator Driven Attributes list.



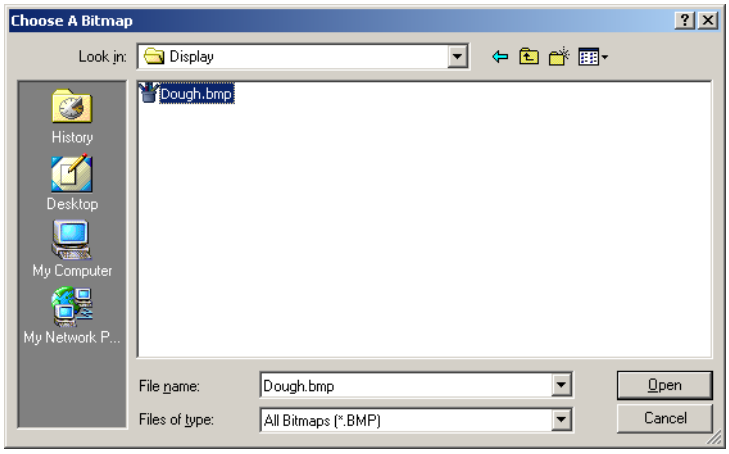
7. Click OK to complete adding the dynamic attribute and close the dialog box.

Adding a Graphic

Now let's add a portion of cookie dough underneath the first tank. Rather than drawing our cookie, we're going to use a drawing of a cookie saved as a bitmap graphics file (or bmp). This file is located in the Display directory.

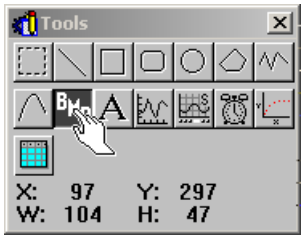
1. Select File→Choose Bitmap.

   The Choose A Bitmap dialog box appears:



2. Double-click the Dough.bmp file to select the cookie bitmap.

3. Now choose the Bitmap tool in the ioDisplay Configurator toolbox as shown below.



4. Click the cursor right above the conveyor belt and underneath the first vessel.

   If the graphic is a little out of place, it can easily be moved to the correct position using the Select tool.

5. Choose the Select tool in the toolbox, and then click the bitmap graphic to select it.

   Nine square sizing handles appear around the graphic.

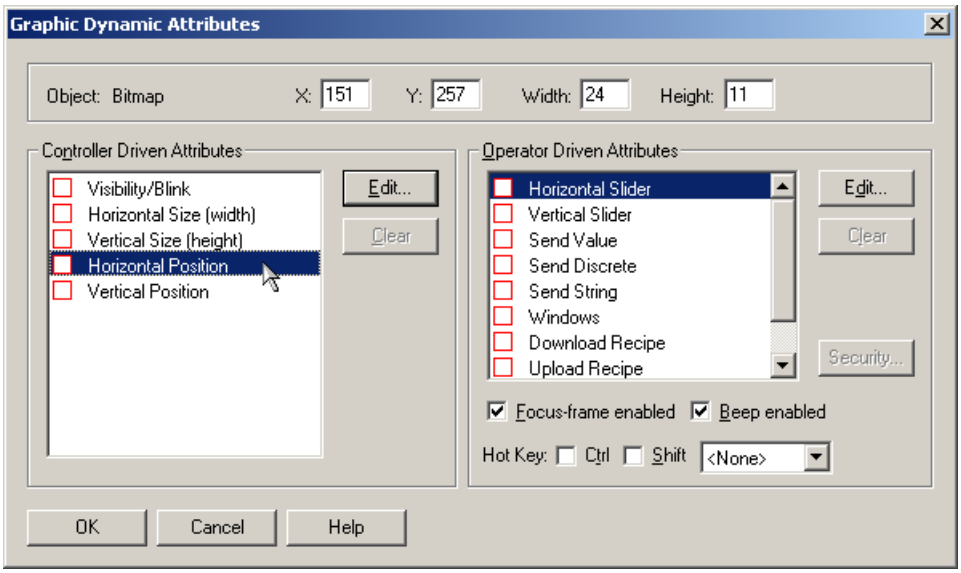
6. Put your cursor within the sizing handles, click your mouse, and drag the cookie to the correct position above the conveyor belt and underneath the first vessel.

   Besides visually placing the graphic, you can also use the arrow keys on the keyboard or use the X: and Y: coordinates displayed in the toolbox to help you place the cookie. We suggest coordinate locations X:151 and Y:257, but your coordinates may differ.

Now let's give the bitmap graphic some attributes to animate it. To make the cookie look like it's moving across the conveyor belt, we'll configure an attribute to affect its horizontal position.

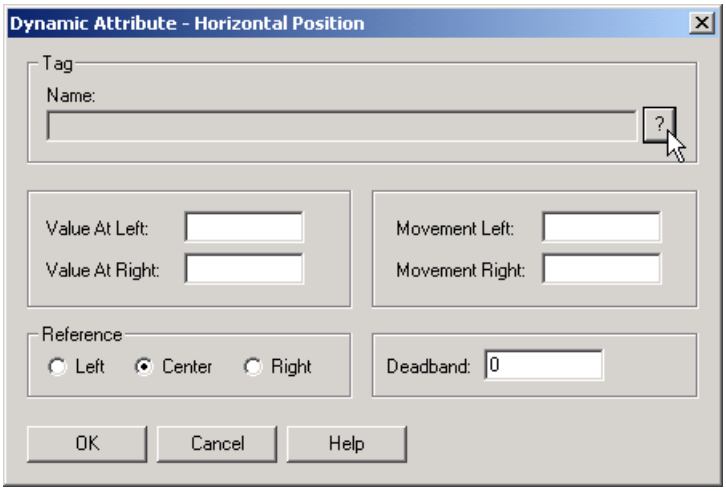
7. Double-click the cookie bitmap graphic.

The Graphic Dynamic Attributes dialog box opens:



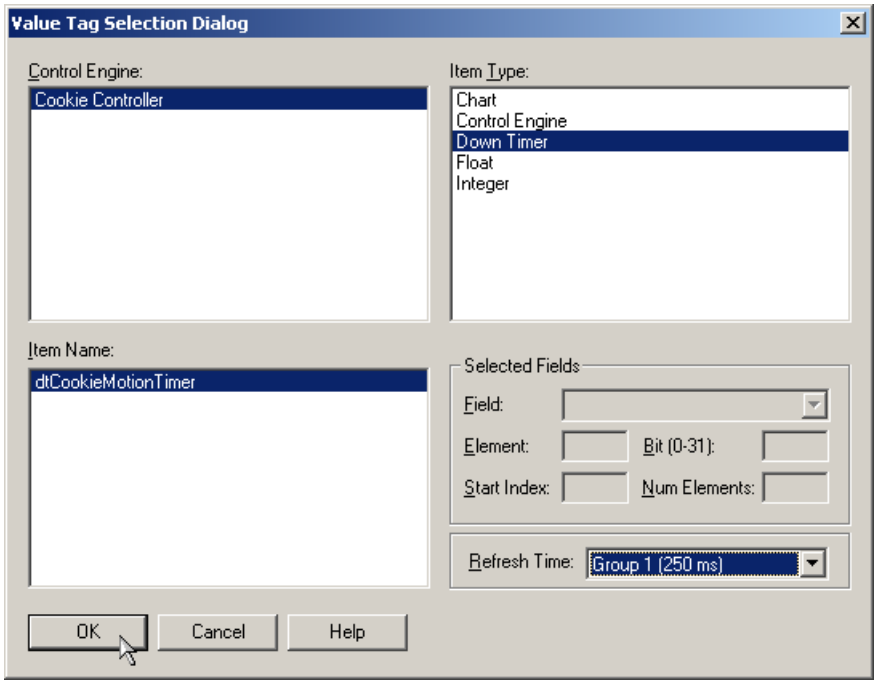
8. In the Control Engine Driven Attributes list, double-click Horizontal Position.

The Dynamic Attribute - Horizontal Position dialog box appears. We need to configure a tag to which we will connect the cookie bitmap graphic. This time we'll connect it to a value ioDisplay reads from the control engine.



9. Click the Tag Selection button  .

The Tag Selection dialog box appears:



10. Select the following:

- Cookie Controller from the Control Engine group
- Down Timer from the Item Type group
- dtCookieMotionTimer from the Item Name group

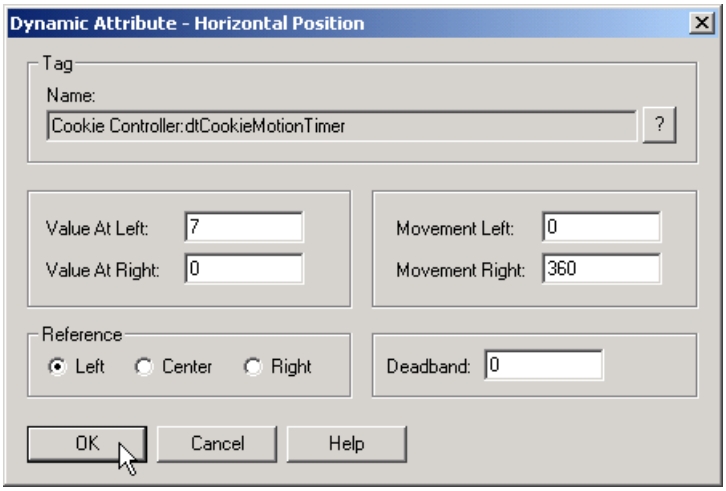
11. In the Refresh Time drop-down list, select Group 1 (250 ms).

12. Click OK.

13. In the Dynamic Attribute - Horizontal Position dialog box, enter the following values, using the TAB key to move from field to field:

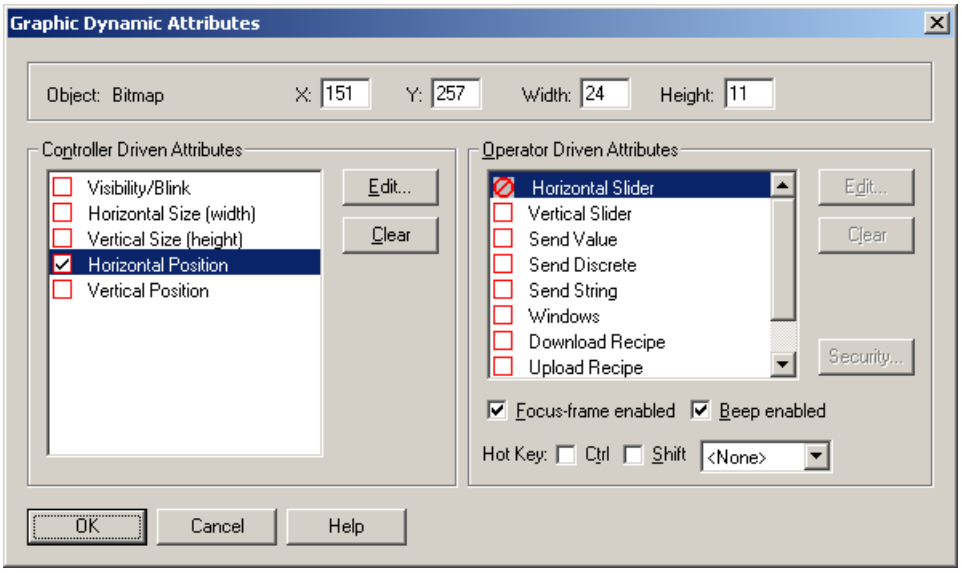
- Value At Left: 7
- Value At Right: 0
- Movement Left: 0
- Movement Right: 360
- Reference: Left
- Deadband: 0

When complete, the dialog box should look like the example below.



14. Click OK to close the dialog box.

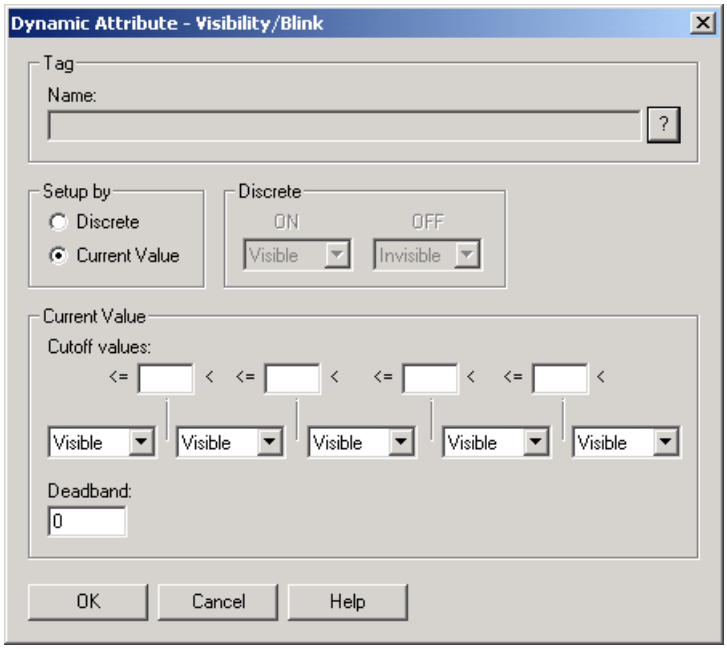
The Graphic Dynamic Attributes dialog box appears:



Now we need to configure an attribute that will make the cookie dough appear to drop out of the first vessel.

15. Double-click Visibility/Blink in the Control Engine Driven Attribute list.

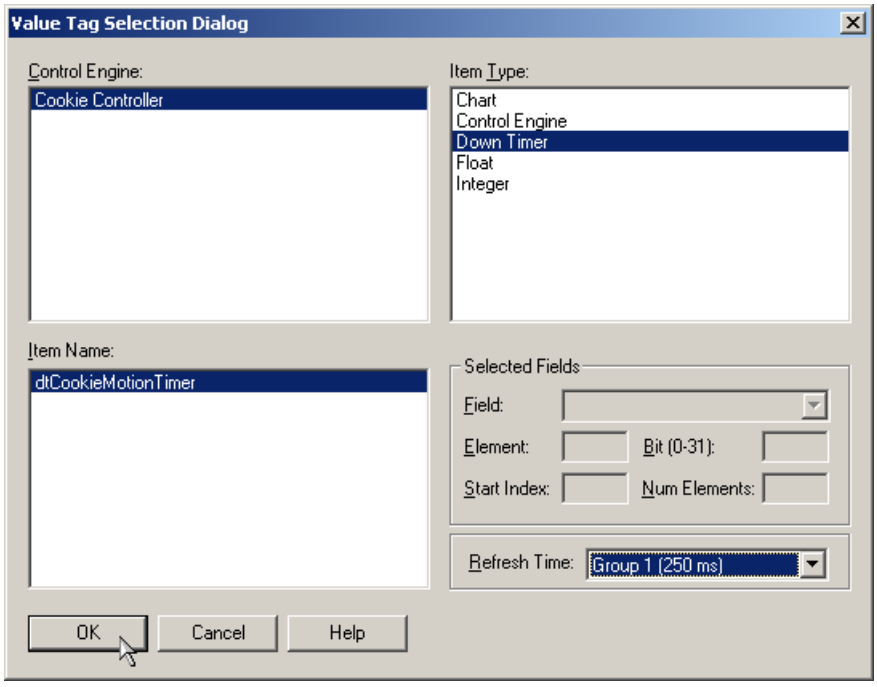
The Dynamic Attribute - Visibility/Blink dialog box opens:



16. In the group Setup by, select Current Value.

17. Click the Tag Selection button  .

The Value Tag Selection dialog box appears:

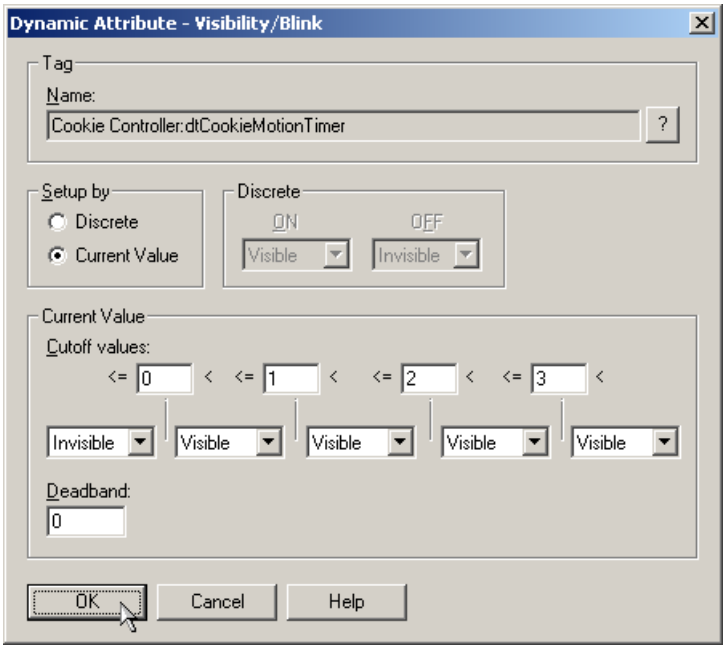


18. Select Down Timer as the Item Type and dtCookieMotionTimer as the Item Name.

19. In the Refresh Time drop-down list, select Group 1 (250 ms).

20. Click OK.

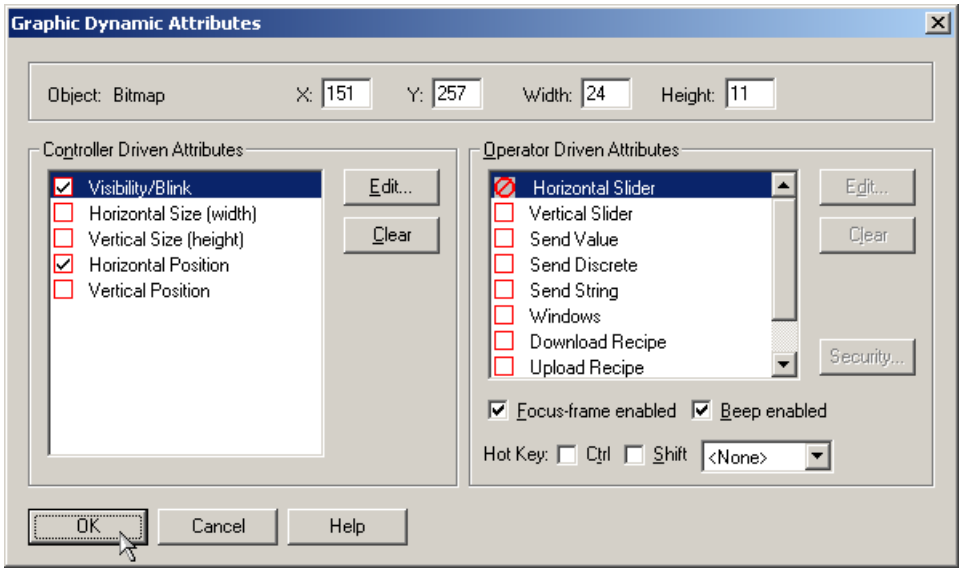
21. Fill in the remaining fields in the Dynamic Attribute - Visibility/Blink dialog box so that it looks like the example below:



22. Click OK.

In the Graphic Dynamic Attributes dialog box that appears, notice that the Visibility/Blink control engine-driven attribute is checked. In the Operator-Driven Attributes list, a Not Available button ⊘ appears next to the Horizontal Slider attribute. This means that this

attribute cannot be configured because other dynamic attributes that have already been configured will conflict with the attribute.



23. Click OK to close the dialog box.

24. Save the project by selecting File→Save Project.

25. Close ioDisplay Configurator by clicking the Close Window button .

Downloading to the Control Engine

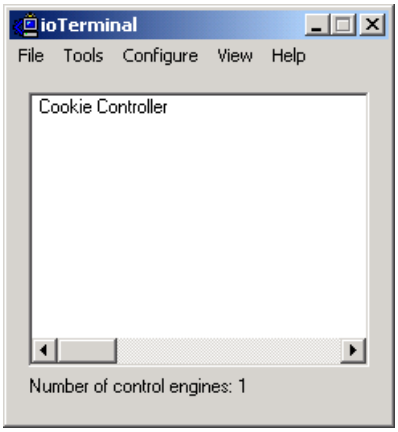
Let's try running our project and see if there's anything we need to change.

In order to see our animated display, we have to run the ioDisplay Runtime program. But before we do that, we need to download our ioControl strategy to the control engine.

NOTE: ioControl strategies are usually downloaded to a control engine for convenience using the ioControl application. However, we'll download our ioControl strategy using an ioDisplay utility called ioTerminal. To learn more about downloading strategies to a control engine, see the ioControl User's Guide.

1. Click the Windows Start button and select Programs→Opto 22→ioProject Software→Tools→ioTerminal.

The ioTerminal window appears, displaying the name of our control engine.



2. Select Cookie Controller, and then choose File→Download Control Engine Forth File to download the run file for our ioControl strategy.

   The Download File dialog box opens:

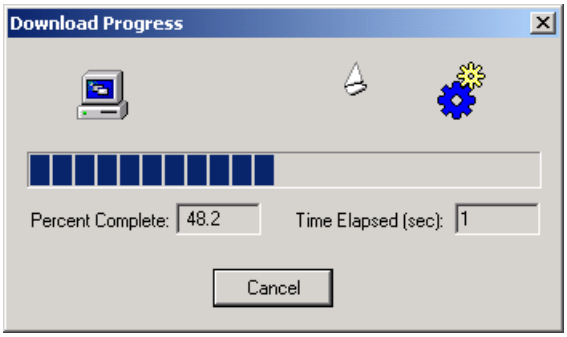
3. Click Browse and change directories to Program Files\Opto22\ioProject Software\ioDisplayExamples\ioCookies\Control.

4. Select "cookies.crn" and click Open.

   The file should be listed in the Download File dialog box.

5. Click OK to continue the download.

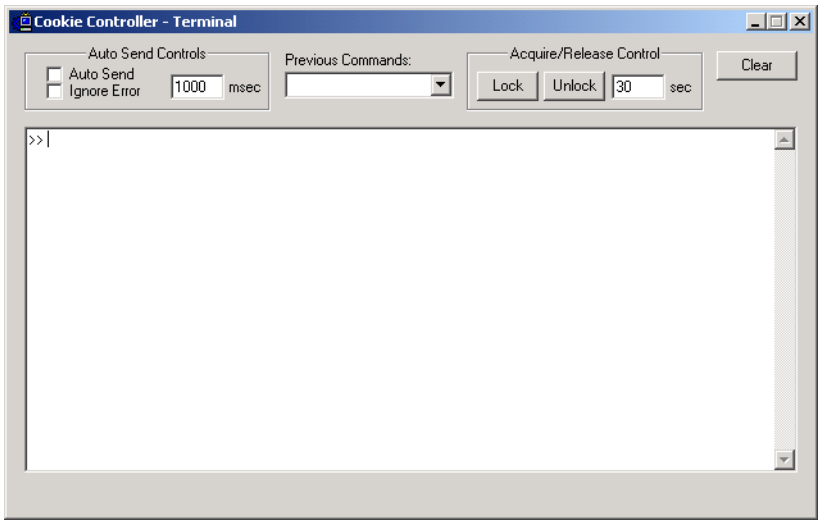
   A Download Progress screen appears briefly to display how the download of the strategy to the control engine is proceeding.



After the strategy has downloaded to the control engine, we'll start the strategy running on the control engine. For convenience, we'll start the strategy by entering commands in ioTerminal's terminal window. A more common method is to start the strategy while running ioControl.

6. When the download is finished, select Tools→Start Terminal.

   The terminal window appears. Notice that the window now has the same title as our control engine, and is ready to accept commands we enter in the list area.



The control engine accepts commands in the Forth programming language. Let's type in a Forth command to verify that the control engine is communicating properly. Commands are case-sensitive, so make sure you enter the commands exactly as shown below.

7. Click your cursor right next to the right-arrows >> and then type the following command:

   ```
   PTIME
   ```

8. Press ENTER.

   The time to which the control engine's internal clock is set is displayed.

9. Next, type the following command (including the underscore):

   ```
   _RUN
   ```

10. Press ENTER.

    This time we'll only see the double-arrow prompt >> on the next line. The strategy in our control engine is now running.

11. Close the ioTerminal window by clicking the Close Window button .

12. Close ioTerminal by clicking the Close Window button .

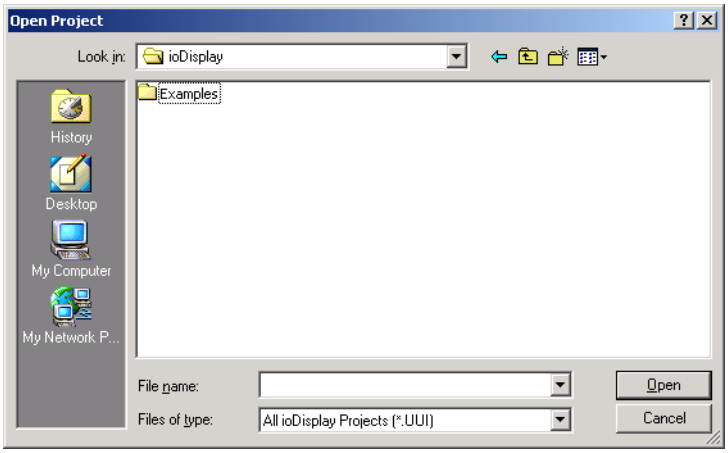
Running the Project

It's time to run the project and see what our display can do.

1. Click the Windows Start button and choose Programs→Opto 22→ioProject Software→ioDisplay Runtime.

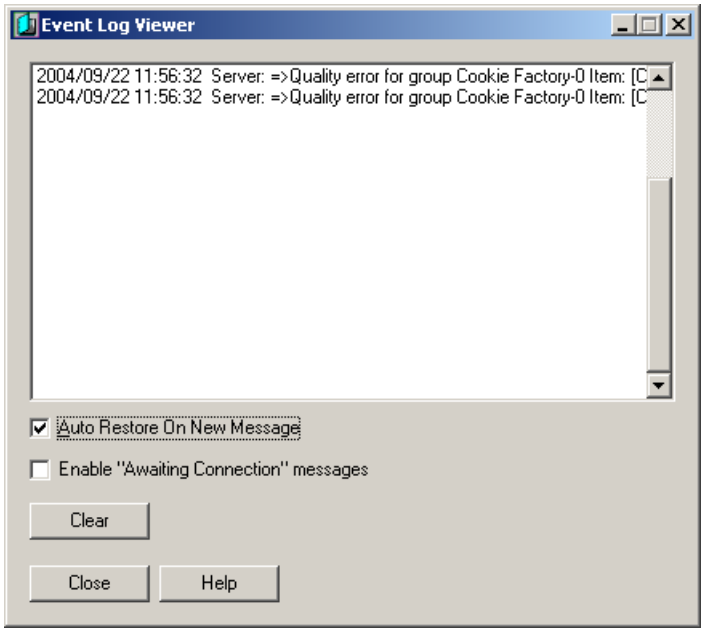
   The main window for ioDisplay Runtime appears. The window is empty since we haven't loaded our project yet.

2. In the Runtime window, choose File→Open Project.



3. In the Open Project dialog box that opens, navigate to the Program Files\Opto22\ioProject Software\ioDisplayExamples\ioCookies\Display directory, and then double-click the cfactory.uui project.

The Event Log Viewer window should appear, displaying messages about the ioDisplay Runtime session. You'll see a message showing that the control engine is connecting to ioDisplay's scanner.



At the bottom of the window, you'll see the Auto Restore on New Message option selected. This means anytime ioDisplay Runtime issues a message (that is, an error or status message), the Event Log Viewer will become the active, or topmost, window on the Windows desktop. You'll also see the Enable "Awaiting Connection" Messages option, which lets you choose whether to display some startup messages.

4. Click the Close button to close the Event Log Viewer window.

5. Start the cookie factory by clicking the on-screen Start button.

   Watch the cookie dough drop out of the first tank and move down the conveyor. Notice that the cookie appears to move outside of the oven and inspection stations.

6. Close ioDisplay Runtime by clicking the Close Window button .

Fine-Tuning the Visuals

Remember how the cookie looked traveling outside the oven and inspection stations? We'll quickly fix that by making the cookie bitmap graphic the rearmost graphic on the screen. This way, when it travels by the stations, it will appear to go through them.

1. Open ioDisplay Configurator.

   The Cookie Factory project should open automatically. If it doesn't, open the project file using the File menu.

2. With the Select tool, click the portion of dough on the conveyor belt.

3. Right-click the mouse, and from the pop-up menu that appears choose Z-Order→Send to Back.

   The graphic will now appear to travel through the oven and inspection stations, not in front of them.zz

4. To save the project and start ioDisplay Runtime to see the display working properly, select File→Save Project and Load Runtime.

There are a few more things you can explore on your own. Notice that when you pass your cursor over the setpoint boxes (SP), a black outline appears. Click on one of the boxes and an attribute dialog box appears. You can go back to the Configurator and see how this attribute was set up. You might also want to look at the tank flap attributes.

What's Next?

After stepping through this chapter, you should have a handle on how simple it is to use ioDisplay. By taking the time up front to step through our short demo, you're now ready to explore the many possibilities available.

Proceed to Chapter 3, "What Is ioDisplay?" to learn more about planning and designing an operator interface. You'll also find out more about the windows and menus that make up ioDisplay Configurator and ioDisplay Runtime.

At some point you may also want to take a look at the sample projects in the "Examples" folder under ioDisplay. You'll get ideas for how to use different ioDisplay onscreen objects and learn about various graphics you can use within your own ioDisplay projects.

CHAPTER 3

What Is ioDisplay?

Introduction

This chapter provides a general overview of using ioDisplay, including information on what it's used for, project structure and design, and general terms you'll encounter.

In This Chapter

About ioDisplay

ioDisplay is a software package used to create human-machine interfaces (HMIs), or operator interfaces, for monitoring control systems. You can use ioDisplay to create an HMI that will monitor an ioControl strategy running on an Opto 22 control engine, providing real-time and historical information to the operator about the performance of different parts of a control system.

ioDisplay works with ioControl, a flowchart-based control language for writing control applications. See the *ioControl User's Guide* for more information on using ioControl to create strategies.

Configurator and Runtime Applications

Two primary software applications make up ioDisplay: ioDisplay Configurator and ioDisplay Runtime.

ioDisplay Configurator—Use ioDisplay Configurator to create a project that contains graphics that appear on the computer monitor to represent your control process. The project also contains information on how these graphics are connected to data in an ioControl strategy running on a control engine, and defines how the graphics' attributes change as this data changes.

ioDisplay Runtime—Use ioDisplay Runtime to run the project created in ioDisplay Configurator. Running the project means that the attributes of the graphics on the computer monitor (such as size, position, or color) are continuously updated based on data provided by the control engine. If controls such as buttons and sliders are part of the ioDisplay project, the operator can use on-screen controls to change values that appear. This is how ioDisplay is used to control processes as well as monitor them.

ioDisplay also includes a separate "monitor-only" version of the ioDisplay Runtime application. This version of ioDisplay Runtime is functionally identical to the regular Runtime application, except that it cannot be used to send values to a control engine. This can be useful for industrial projects where no operator intervention is required.

The **Event Log Viewer**, which is part of ioDisplay Runtime, starts automatically when ioDisplay Runtime is started. The Event Log Viewer displays a window that posts messages about ioDisplay communication activity. Typically, it pops up above all other windows when a message is posted, but this feature can be disabled.

ioDisplay Terminology

Project—A collection of draw windows, historic logs, sounds, recipes, graphics, and all their attributes that has been developed with the ioDisplay Configurator. When the project is saved, several files are created.

- The *main project file,* which has an .uui extension.
- *Draw window files* are created automatically for each draw window used to display graphics in a project. These files have sequentially numbered file extensions starting with an uppercase .W (for example, .W01, .W02, and so on).

These project files, together with ioDisplay Runtime, present an animated graphics interface for a control system. See Appendix C, "ioDisplay Files" for a complete list of the files that make up an ioDisplay project.

Windows—ioDisplay projects have one or more draw windows inside the ioDisplay main window. A *draw window* is essentially a blank page on which you place, draw, or edit graphics and other elements that will make up your operator interface. A draw window has static attributes of position, size, and color. It also has visual states of open, closed, or iconified.Your project design determines the number of draw windows and their contents.

A *main window* is the area of the display where you can view your application. Following the conventions used in most Microsoft Windows applications, a *main window* contains a menu bar that allows you to select various command options, and a title bar that displays the full project path.

Objects—Objects include draw windows, graphics, alarm triggers, and trends (or graphs). There are two types of objects: *static* and *dynamic*. Static objects do not change while ioDisplay Runtime is running. Dynamic objects change appearance, or cause the appearance of other ioDisplay objects to change while the project is running.

Tags—A *tag* refers to data items, such as variables or I/O points loops, from an ioControl strategy. To access tags in a project, select the ioControl strategies for the project. All tags in the selected strategies are then available to ioDisplay. Tags are used to animate your operator interface through connections to graphic objects and their dynamic attributes. As the values of tags change through control engine- or operator-driven attributes, the appearance of the graphics change. Tags are also used as triggers to initiate system events such as sounds, historic logging, and window configurations.

Connections—A connection is made in ioDisplay when an ioControl tag is selected as either the source that will change a graphic, or as the tag destination for any data changes entered by the operator.

Planning a Project

An ioDisplay project is made up of a collection of windows and other elements you create and configure in ioDisplay Configurator. You add graphics to the windows to create an operator interface, and then connect to I/O data and variables in the tagname database of the associated ioControl strategy (the control program running on a Opto 22 control engine).

Once the windows, associated ioControl strategy and Opto 22 control engine, and other attributes of the ioDisplay project have been set up using ioDisplay Configurator, you can run the project in ioDisplay Runtime. When ioDisplay Runtime is started, the project communicates with one or more control engines. As the strategy runs on the control engine, values and states of tagnames in the ioControl strategy database are continuously updated. This changing data in turn modifies the attributes (such as size and position) of the graphics that are connected to the tagnames. The end result is an animated, continually updated display that shows the status of a control process.

Project Design

The usefulness of your ioDisplay project ultimately depends on how effective the display, or operator interface, is. To create an effective operator interface, you may want to consider these tips when you're designing your project:

- Know your control process, including both the theoretical operation of the process and the "hands-on" tasks required of the operator.
- Identify the information the end user of your project, the operator, needs to know at different points in time. Use this information to determine what will appear on the display.
- Consider the following ways to use and organize windows in a display:
  - A *single window* can display an overall picture of the control process, and is helpful in quickly assessing the general state of all operations.
  - *Individual windows* can display a closer look at the operations associated with different stations. The individual windows can contain detailed information and provide controls that would be difficult to present in a single window.

Project and Operator Interface Security

ioDisplay supports several important security features, including operator authentication, encrypted logging of operator actions, and password protection for the project files. You can configure your ioDisplay project to do the following:

- Allow or deny operator access to the HMI, as well as the use of individual on-screen controls, based on users and groups defined in a Microsoft Windows network. (See "Security Settings for Graphics and Dynamic Attributes" on page 7-4.)
- Log all HMI use and operator actions to an encrypted archive. (See "Security Settings" on page 10-7.)
- Assign a password to the ioDisplay project to prevent unauthorized users from opening it in the ioDisplay Configurator authoring application. (See "Protecting a Project with a Password" on page 4-3.)
- Assign a password to individual windows in an ioDisplay project to prevent unauthorized users from opening them. (See "Modifying Draw Windows" on page 6-2.)

Window Design

When you create a new project, a project window and one or more draw windows will be available in the ioDisplay main window. After deciding which windows you will use for your project, consider the design of individual windows and how they interact with other windows. This is where the organization of your windows comes into play. For example, you could have the operator use the Runtime menu commands to view different windows, or you could design buttons to let an operator jump directly to related windows.

Keep in mind that the visual state of a window can affect the performance of ioDisplay and the control engine. As more windows are opened or iconified, ioDisplay will gradually start to update graphics more slowly. Window states, listed below, also affect how the ioDisplay Runtime software application scans the control engine and updates graphics.

- An *open* window causes Runtime to update graphics with data from the built-in scanner.
- An *iconified* window causes Runtime to continue requesting data from the scanner, but graphics in the iconified window are not updated.
- A *closed* window removes the associated tags from the scanner.
- *SuperTrends* within a window can be configured to have Runtime request data from the scanner or not.

Other choices you will have to make are whether a window should be pop-up or full-screen, and whether a window's visual state is affected by another window. These and other aspects of configuring windows are covered in Chapter 6, "Working with Graphics," and Chapter 10, "Using ioDisplay Runtime."

Using Multiple Monitors

If your operator interface will run on Windows 2000, WIndows XP, or another version of Microsoft Windows that supports multiple monitors connected to one computer, an additional project planning decision is whether to design your ioDisplay project to use more than one monitor. The additional display space gained from using multiple monitors offers advantages such as being able to keep numerous windows open permanently. However, you should consider the additional hardware cost and extra desktop space a multiple monitor setup requires.

An important factor to also consider is that each window in an ioDisplay project requires computer memory (RAM). If you plan to display several windows on multiple monitors, the computer running the ioDisplay project may need to have additional memory installed.

Hardware and software requirements for using multiple monitors are described in "System Requirements" on page 5. For steps to set up an ioDisplay project to use multiple monitors, see "Extending a Project Across Multiple Monitors" on page 4-3.

ioDisplay Windows

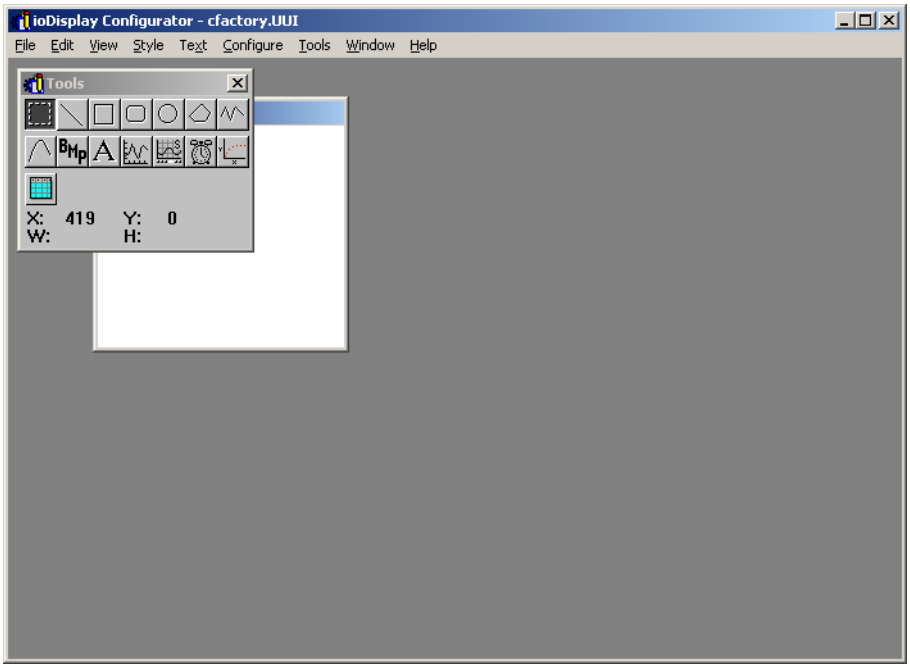
As mentioned previously in "About ioDisplay," ioDisplay is composed of two software applications, Configurator and Runtime. This means there are two environments in which you will use ioDisplay. In learning about the windows that make up ioDisplay, we will first discuss the main components of the Configurator environment, and then explore the Runtime environment.

ioDisplay uses standard Microsoft Windows conventions, so as you use Configurator and Runtime, you'll recognize familiar window elements such as title bars and the menu bar, as well as controls such as the minimize, maximize, and close buttons.

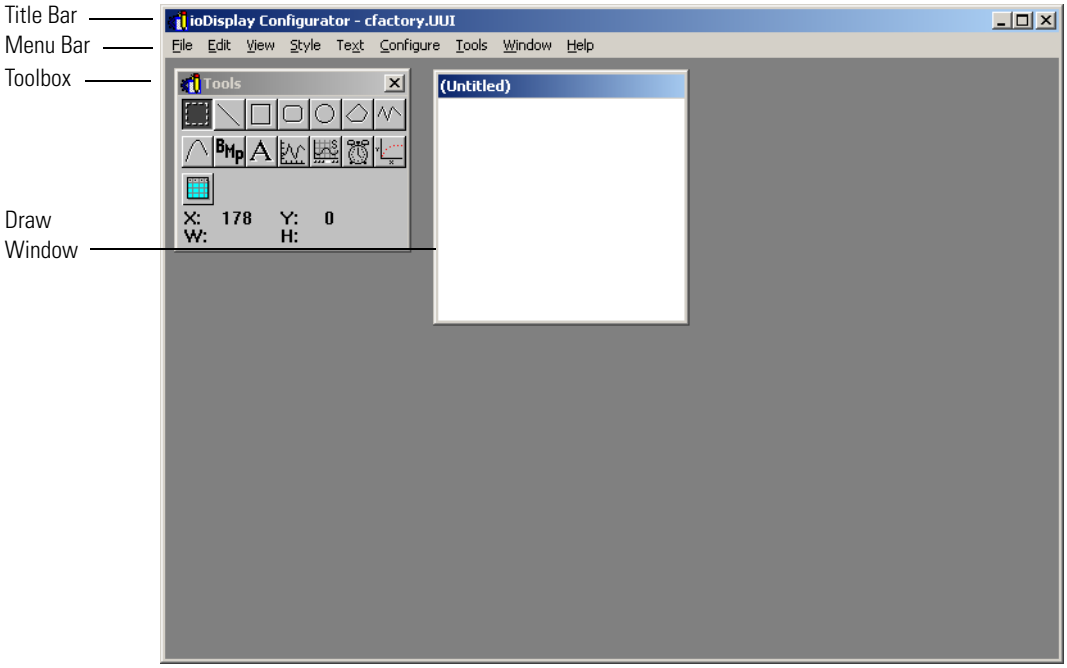
NOTE: If you need more information on working with Microsoft Windows, refer to the documentation from Microsoft and your computer manufacturer.

ioDisplay Configurator Main Window

When you first start ioDisplay Configurator and create a new project, your screen should look similar to this:



The Configurator main window consists of a title bar and a menu bar, along with other standard Windows elements, and contains the toolbox and one or more draw windows.



Hiding the Menu Bar

If you need additional space to position draw windows, you can hide the menu bar to use the space it occupies. If you do this, note that you won't be able to access commands on the menu bar.

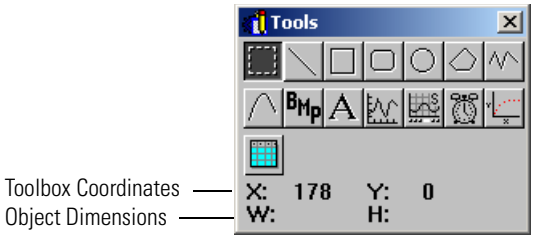
To hide the menu bar, do one of the following:

- Select View→Hide Menu Bar
- Press ESC on the keyboard.

To view the menu bar again, press ESC on the keyboard.

Toolbox

The toolbox contains a set of graphical icons that represent tools you can use in the Configurator. Click any tool to select it, and then use it in the draw window. Also, below the graphical icons, the toolbox displays coordinates and object dimensions to aid you in your drawing tasks.



If you want to use the space the toolbox occupies, you can hide the toolbox by selecting View→Hide Toolbox. To open the toolbox again, select View→Show Toolbox.

Tool Definitions

Each tool in the toolbox, arranged from left to right across the top of the toolbox, and then across the bottom, is described below. For more details about any tools available in the toolbox, see Chapter 6, "Working with Graphics."

Use the following tools as described below:

 **Select tool**—select, move, and size text and graphics.

 **Line tool**—draw lines.

 **Rectangle tool**—draw rectangles and squares.

 **Round rectangle tool**—draw rectangles and squares with rounded corners.

 **Ellipse tool**—draw ellipses and circles.

 **Polygon tool**—draw multi-sided objects.

 **Polyline tool**—draw multiple-line-segment objects.

 **Curve tool**—draw single curved lines or multiple-line-segment curved lines.

 **BMP tool**—place a bitmap.

 **Text tool**—add text to draw windows.

 **Trend tool**—place trend charts in draw windows.

 **SuperTrend tool**—place real-time and historic trends in draw windows.

 **Alarm tool**—place alarm graphics in draw windows.

 **XY Plot tool**—place graphs that display data plotted on x and y axes.

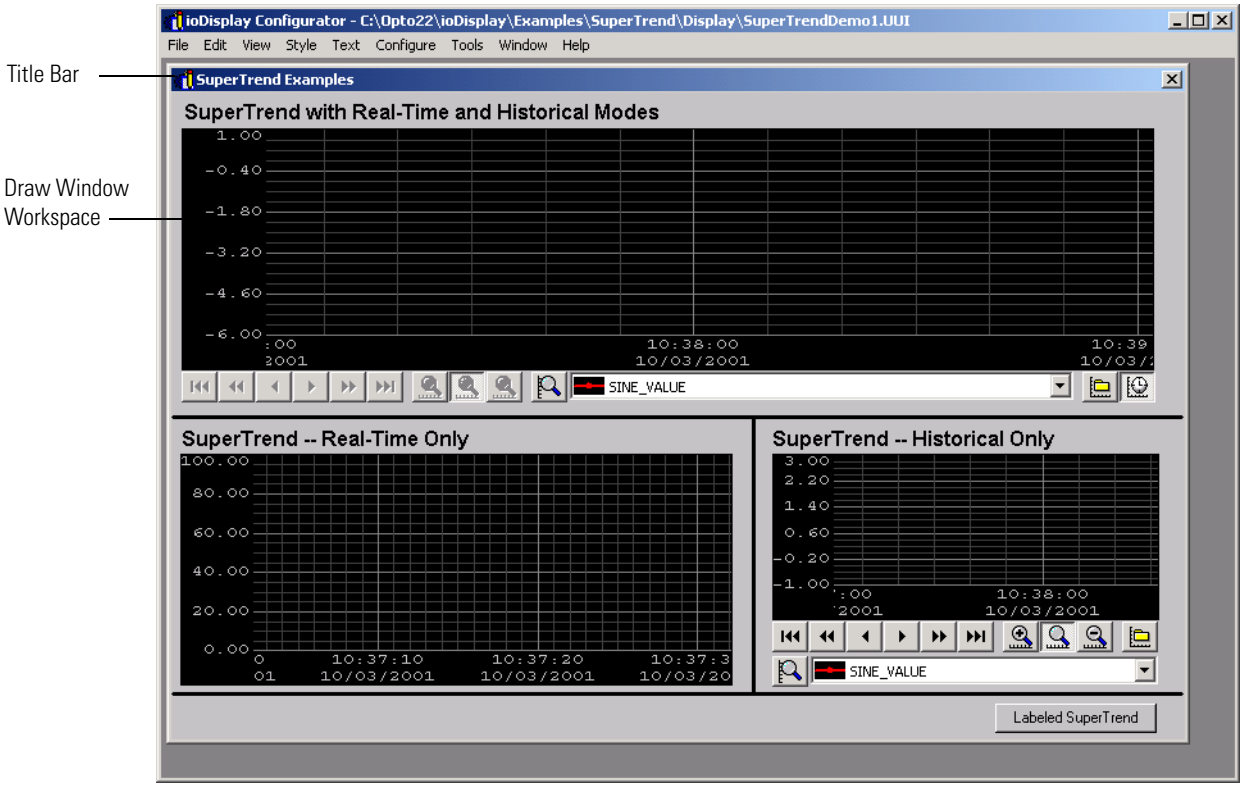
 **Numeric Table tool**—place numeric tables in draw windows.

Toolbox Coordinates and Object Dimensions

Just underneath the tools, you can see the toolbox coordinates and object dimensions. The coordinates show the cursor's position (in pixels) when it is over a draw window. The X: and Y: coordinates are read with the axis zero-points at the top-left corner of the draw window. If you create an object, the object's dimensions (width and height) are shown next to W: and H:, respectively.

Configurator Draw Windows

Configurator draw windows are where all graphics for your ioDisplay project are drawn and edited. They contain the graphics and other elements you work with to create your display.

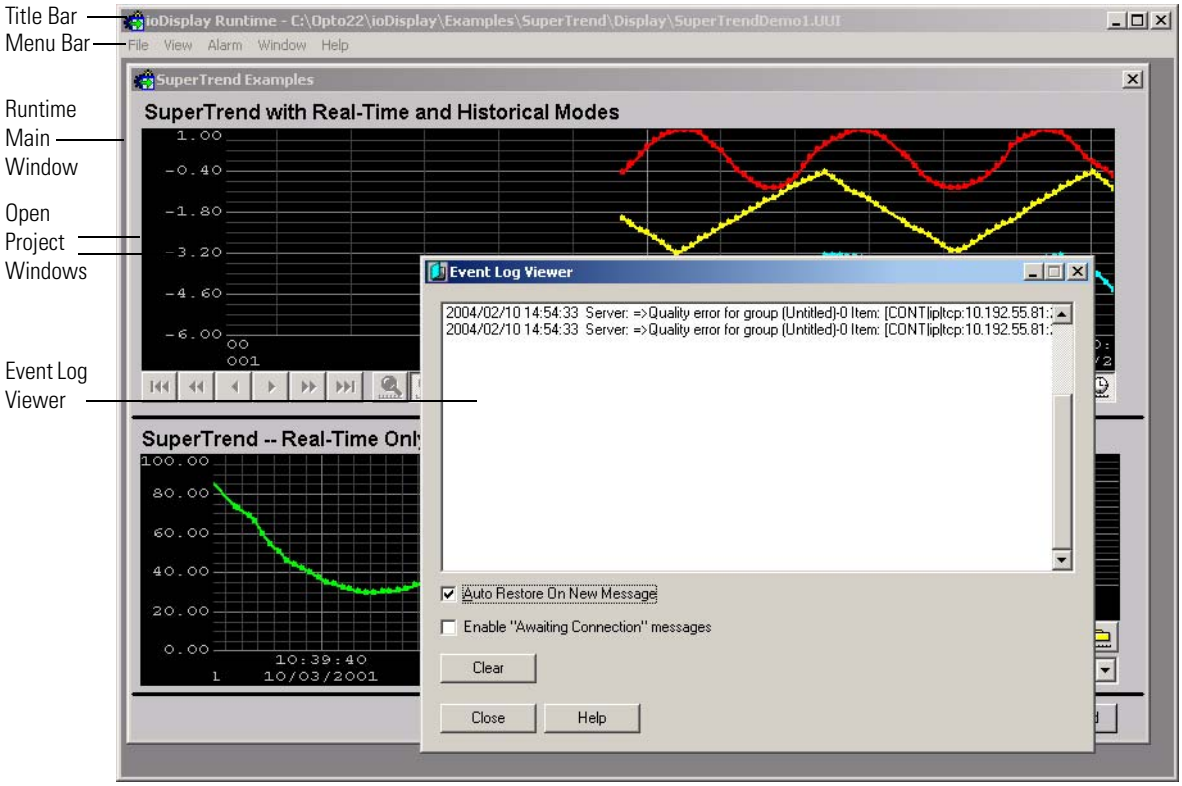


Redrawing an Active Draw Window

You can redraw an active draw window in ioDisplay Configurator by selecting View→Redraw. Incomplete graphics (such as an incomplete polygon) in the draw window are removed when you select this command.

ioDisplay Runtime Main Window

When you start ioDisplay Runtime, what appears on screen should look similar to this:



The Runtime main window consists of a title bar, a menu bar, other standard Windows elements, project windows, and the Event Log Viewer. Much like a frame, Runtime's main window contains all the elements and actions that occur during Runtime use.

If you need additional space to position project windows and other windows in Runtime, you can hide the menu bar to use the space it occupies. If you do this, note that you won't be able to access commands on the menu bar.

Changing How the Main Window Appears in Runtime

You can configure the main window so that it appears without a title bar or menu bar, and also change several other settings. See "Setting up Runtime" on page 10-2 to learn how to configure elements of the main window.

Hiding the Menu Bar

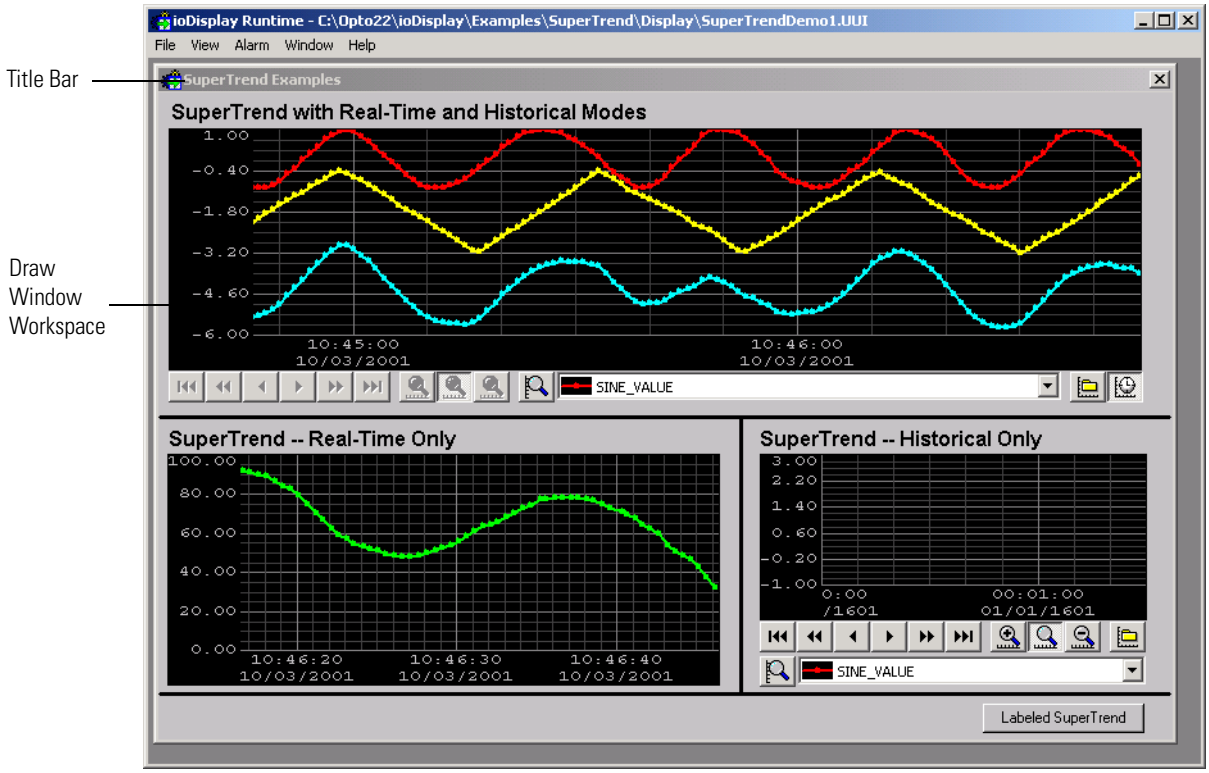
While working with an ioDisplay project in Runtime, you can gain additional space on your screen by hiding the menu bar. To hide the menu bar, do one of the following:

- Select View→Hide Menu Bar
- Press ESC on the keyboard.

To view the menu bar again, press ESC on the keyboard.

Runtime Project Windows

After you've opened a project in Runtime, you see the project windows.



Project windows are the Configurator draw windows that you created for the project. Notice that when you launch your project in Runtime, these windows are the same size and in the same relative position as when you closed the Configurator project. (Depending on certain configuration options, the relative positions of the windows may differ slightly from those in the Configurator project.)

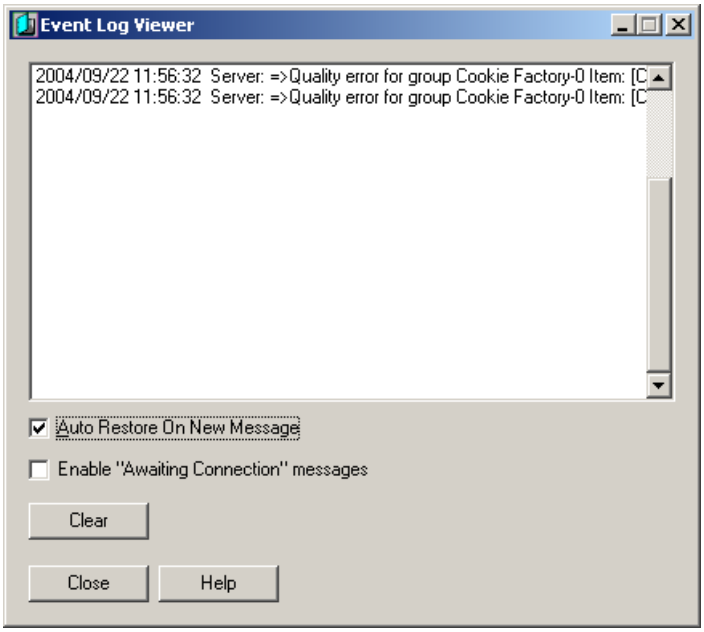
Project windows are composed of a title bar and a workspace. The title bar displays the name of the window, and the workspace contains all of the graphics work you did in the Configurator.

Configuring How Draw Windows Appear in Runtime

You can define how a draw window appears in ioDisplay Runtime, including its visual state (closed, iconified, or open), relative position, and other settings. See "Using Draw Windows" on page 6-1 for instructions and more information.

Runtime Event Log Viewer

The Event Log Viewer contains a list of system errors and messages that occur during Runtime. The most recent messages appear in the list, but when there are more messages than can reasonably fit, scroll bars appear so that you can view older messages.



You can also double-click any message to view its entire contents, if they are not already completely visible. The Event Log Viewer can be manipulated like any other standard window.

The Auto Restore on New Message option sets whether the Event Log Viewer dialog box automatically jumps to the foreground when a new event message is received. The Enable "Awaiting Connection" Messages option is used to hide or display common startup messages.

CHAPTER 4

Working with Projects

Introduction

This chapter explains how to work with projects. You'll find out how project files are organized, and then learn how to create, open, and save a project. An optional, advanced procedure for customizing how ioDisplay Configurator starts is also presented.

In This Chapter

How Projects Are Organized

An ioDisplay *project* is a collection of all the files created in ioDisplay Configurator that define one operator interface. The project includes the windows you create, bitmaps that appear in them and their attributes, and any other elements you set up. See Chapter 3, "What Is ioDisplay?" to learn more about the various components of an ioDisplay project. Also see Appendix C, "ioDisplay Files" for a complete list of files associated with ioDisplay.

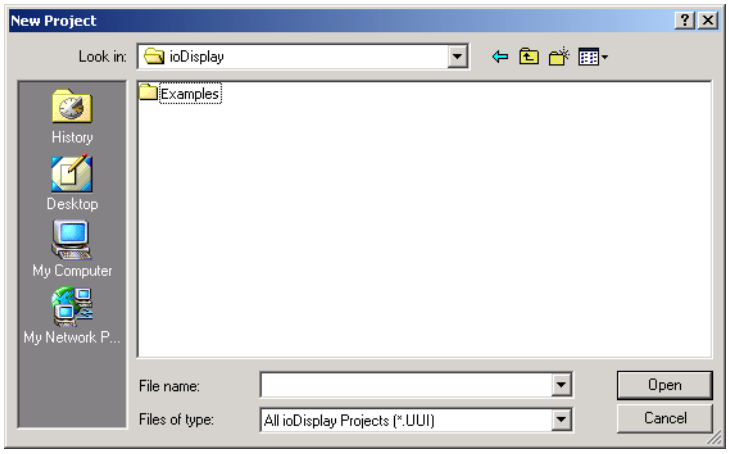
Each ioDisplay project should be located in its own directory. You'll find that separate directories make keeping track of any one project's files easier, especially when you back up the project or copy the files to disk. Though not recommended, multiple projects can be stored in the same directory if necessary.

Creating a Project

To create a new project in ioDisplay Configurator, follow these steps:

1. Select File→New Project.

   The New Project dialog box that appears should look similar to the example below.



   Dialog boxes in ioDisplay follow common Microsoft Windows conventions, so you'll recognize familiar items such as the file list, the Up One Level button, and the File name field.

   NOTE: If you need more information on how to use dialog boxes or other common parts of Microsoft Windows, refer to the documentation from Microsoft and your computer manufacturer.

2. Type a project name in the File name field.

   When you're done creating the new project file, this name is automatically appended with the suffix .uui, indicating an ioDisplay project file.

3. If you want to save your project in a new directory, follow the sub-steps below, and then continue with step 4.

   a. Click the Create New Folder button.

      The new folder you created appears in the list of files and folders. The name of the new folder should be highlighted, meaning you can enter a new name for the folder.

   b. Type a new name for the folder, preferably one that includes the project name.

   c. Double-click the new folder to open it.

4. Click Open to create the project.

   If you have selected a directory that already includes a project, a warning message appears, and you'll be allowed to try again.

When you click Open, the project is created, and a draw window and toolbox appear within the Configurator's main window. The untitled draw window is where you will begin drawing your

operator interface. The toolbox contains several icons representing graphic drawing tools. We will discuss the toolbox more in Chapter 6, "Working with Graphics."

Extending a Project Across Multiple Monitors

If you are designing your ioDisplay project to use multiple monitors connected to the same computer, after creating the project simply extend ioDisplay Configurator's main window across the monitors you want to use. For hardware and software requirements for using multiple monitors, see "System Requirements" on page 1-5.

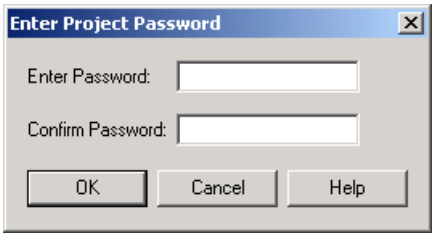
Protecting a Project with a Password

You can protect your ioDisplay project with a password to prevent others from opening and modifying the project using ioDisplay Configurator. The project can still be opened and run in ioDisplay Runtime.

To protect your project with a password, do the following:

1. Select File→Password Protect Project.

   The Enter Project Password dialog box opens.



2. Type a password in the Enter Password field.
3. Type the same password a second time in the Confirm Password field.
4. Click OK.

Your ioDisplay project now cannot be opened in ioDisplay Configurator without entering the password.

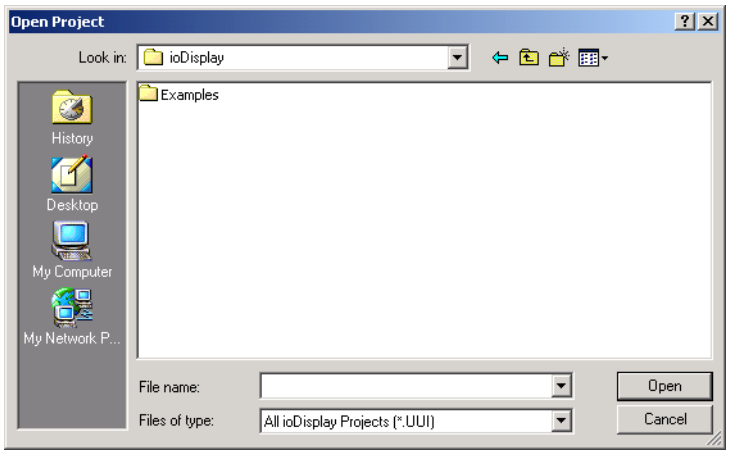
Opening a Project

To open an existing project in ioDisplay Configurator or ioDisplay Runtime, follow these steps:

1. Select File→Open Project.

   (Configurator only) If you already have an open project, you will be asked if you want to save it. Click Yes or No, or click Cancel to close the Save Project dialog box.

   The Open Project dialog box appears.



2. Navigate to the folder where your project is located.

3. Double-click the file in the list to open the project. (You can also select the file, and then click Open.)

   The project opens with any windows positioned just as you last left them.

   NOTE: If the project has been protected with a password, the Enter Password dialog box appears. Type the password for this project and click OK to open the project.

Converting an OptoDisplay Project to ioDisplay

A project created in Opto 22's earlier OptoDisplay software can be converted into an ioDisplay project using conversion utility software included with ioDisplay. Converting an OptoDisplay into ioDisplay can save much time and effort that would otherwise be needed to recreate your older project in ioDisplay. See the *FactoryFloor to ioProject Migration Guide* (Opto 22 form 1596) for instructions and other information on converting an OptoDisplay project to ioDisplay.

Saving a Project

There are three options for saving a project in ioDisplay Configurator: Save Project, Save Project As, and Save Project and Load Runtime.

Save Project

To save your project to the same file name you opened or created, Select File→Save Project. If there have been no changes to the project since you last saved it, no messages appear when this save occurs.

Save Project As

1. To save the project with a new name, select File→Save Project As.
2. In the Save As dialog box, type a project name in the File name field.

   When you're done saving the project file with a new name, this name is automatically appended with the suffix .uui, indicating an ioDisplay project file.
3. If you want to save your project in a new directory, follow the sub-steps below, and then continue with step 4.
   a. Click the Create New Folder button.

      The new folder you created appears in the list of files and folders. The name of the new folder should be highlighted, meaning you can enter a new name for the folder.
   b. Type a new name for the folder, preferably one that includes the project name.
   c. Double-click the new folder to open it.
4. Click Save to save the project with a new file name.

Save Project and Load Runtime

To save your project to the same file name you opened or created, and then start the project in ioDisplay Runtime, select File→Save Project and Load Runtime. This option is particularly useful when you are testing a project and switch often between the Configurator and Runtime components.

Saving Versions of a Project

When developing an ioDisplay project, you can save progressively numbered versions of the project files (for example, MyProject_01, MyProject_02, etc.). Having these "snapshots" of your project as you develop it can be valuable if you need to return to an earlier version, or need to trace the steps you took while building the ioDisplay project.

To automatically create a numbered version of modified ioDisplay project files each time you save the project, select File→Auto Increment Version. Project (.UUI) and window (.WXX) files will be copied, renamed with a version number, and placed in the same directory as the current project. Other project files such as background bitmap images and similar graphics are not copied.

Closing a Project

To close the current project you have open, select File→Close Project. If you've modified the project since it was last saved, you will be asked if you want to save those changes before closing the project.

Only one project can be open at any one time in ioDisplay. Creating or opening a project automatically closes any currently open project first. When this happens, you will be asked if you want to save changes if they haven't been already saved.

Customizing a Project

NOTE: The following section presents advanced procedures for customizing how ioDisplay starts up and opens a project. ***These procedures are not required to run ioDisplay Configurator or Runtime, or to open projects.***

Modifying Default Project Properties

After you've loaded a project for the first time, you'll notice that every time you start ioDisplay Configurator, the application knows which project to load, what sizes the windows are, and several other startup conditions. At some point, you may want to change these initial ioDisplay conditions. To do so, you can run the Windows system Registry Editor utility to modify these conditions.

WARNING: *Use the Windows Registry Editor carefully. It is strongly recommended that you make a backup copy of your Windows Registry before continuing with this procedure. Without a backup copy, if you delete the wrong properties and cannot return the Registry to its original state, application and system files can become unusable and will have to be reinstalled.*

1. From the Windows Start menu, select Run.

   The Run dialog box appears.

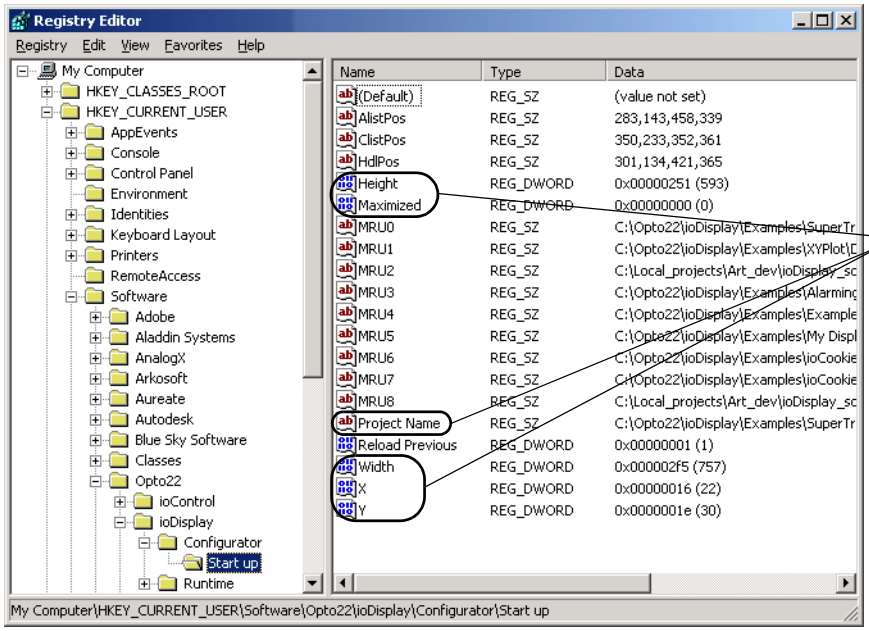
2. Enter the following command in the Open field and press ENTER:

   ```
   regedit
   ```

   The Registry Editor window appears. You should see several folders listed under My Computer.

3. Open the HKEY_CURRENT_USER folder, and then continue to open each of the following sub-folders as they appear:
   - Software
   - Opto22
   - ioDisplay
   - Configurator or Runtime
   - Start up

   In the Start up folder you'll see several properties defined, as shown below.



4. Select the following properties:
   - Height
   - Maximized
   - Project Name
   - Width
   - X
   - Y

   Do not select the (Default) property.

5. Select Edit→Delete, or press DEL on the keyboard.

6. Select Registry→Exit to close the Registry Editor.

ioDisplay has now been initialized to its original startup conditions, and will open as if no project had ever been loaded.

Creating an MS-DOS Batch File

You can use an MS-DOS batch file to have ioDisplay Runtime open and run a project. The batch file you create can be associated with an icon on the Windows desktop. This is a convenient way for an operator to quickly start an ioDisplay project without having to search for the project file in a dialog box.

To create a batch file to open a project in ioDisplay Runtime, do the following:

1. Open an empty text file using Windows Notepad or another text editor.
2. Enter specific commands to perform the following tasks:
   - Change drives to the drive containing the project.
   - Change directories to the directory containing the project.
   - Start ioDisplay Runtime.
3. Now save the file using the .bat file extension.

Once you have created the batch file, you can make a Windows shortcut of this file and place it on the Windows desktop for easy access. Alternately, you could place the shortcut in the Windows Start Up folder so the project starts to run when the computer starts up.

Batch File Example

Here's an example of a batch file that opens a project in ioDisplay Runtime:

```
E:
cd "\strategies\first strategy"
Program Files\Opto22\ioProject Software\ioDisR "first strategy.uui"
```

Here's what each line of the batch file does:

- The first line changes drives to the E: drive.
- The second line changes directories to the directory "\strategies\first strategy."
- The last line starts ioDisplay Runtime, located in \Opto22\ioDisp, using the project called "first strategy.uui."

CHAPTER 5

Configuring Control Engines and Tags

Introduction

This chapter shows how to define the connections to Opto 22 control engine and I/O points, or "tags," that ioDisplay requires.

In This Chapter

Configuring Control Engines

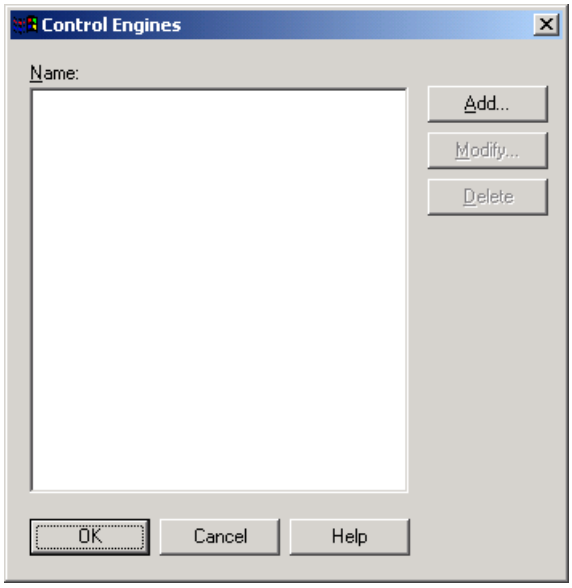
ioDisplay uses the data values a control engine receives from I/O points and integer, float, and string variables to change the attributes of on-screen graphics. (I/O points are defined when you create an ioControl strategy; for more information, see the *ioControl User's Guide*.) For an ioDisplay project to receive this information on individual I/O points (called *tags*), you must first define a connection to an Opto 22 control engine.

ioDisplay can also connect to an Opto 22 controller running an OptoControl strategy. Tags in the OptoControl strategy will be available in your ioDisplay project the same way ioControl tags are.

Follow these steps to select and configure a control engine:

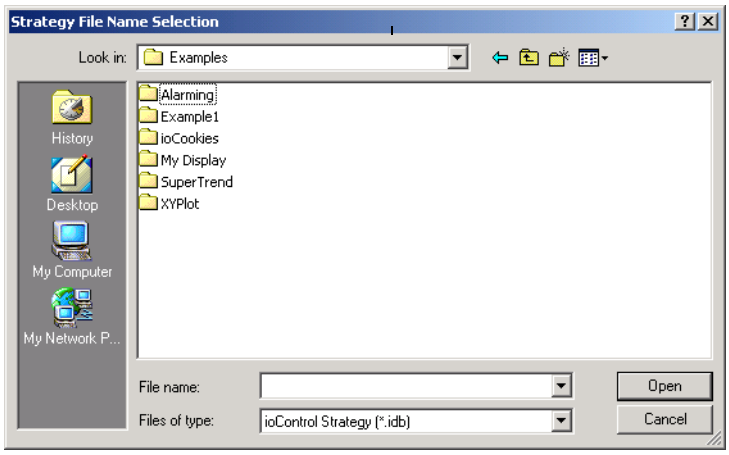
1. Start ioDisplay Configurator and open a project that will be associated with the control engine.
2. Select Configure→Control Engines.

The Control Engines dialog box opens:



If you have not previously configured a control engine for the ioDisplay project you opened, the Name list is empty and only the Add button is available.

3. To locate an ioControl strategy running on the control engine you want to connect to, click Add.



4. In the Strategy File Name Selection dialog box that opens, navigate to the ioControl strategy that is running on the control engine you plan to select.

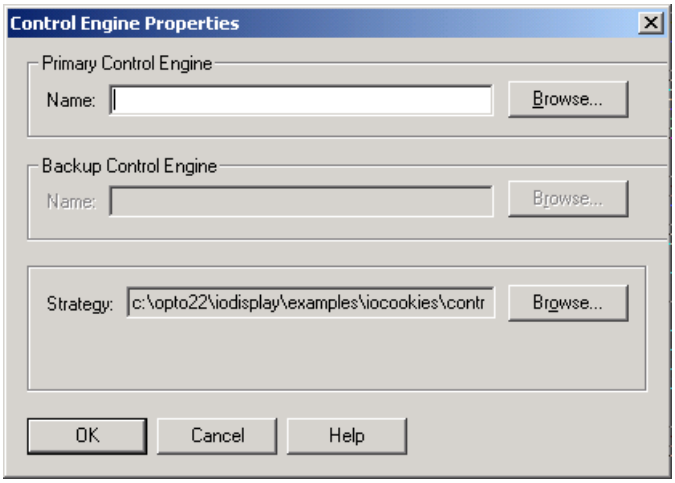
   To locate an OptoControl strategy, click the drop-down menu in the Files of Type dialog box and select "OptoControl Strategy *.cdb".

NOTE: The next few steps differ between ioDisplay Basic and ioDisplay Professional. Follow the set of instructions for your version of ioDisplay.

Final Controller Configuration with ioDisplay Basic

1. Select the strategy file and click Open.

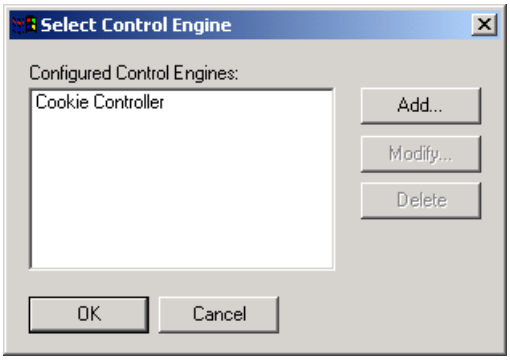
   The strategy you selected appears in the Strategy field of the Control Engine Properties dialog box:



   Now you need to select the primary control engine from which ioDisplay will receive I/O point information. *Remember that this control engine must be running the strategy you selected.*

2. Click the Browse button in the Primary Control Engine group.

   The Select Control Engine dialog box appears:

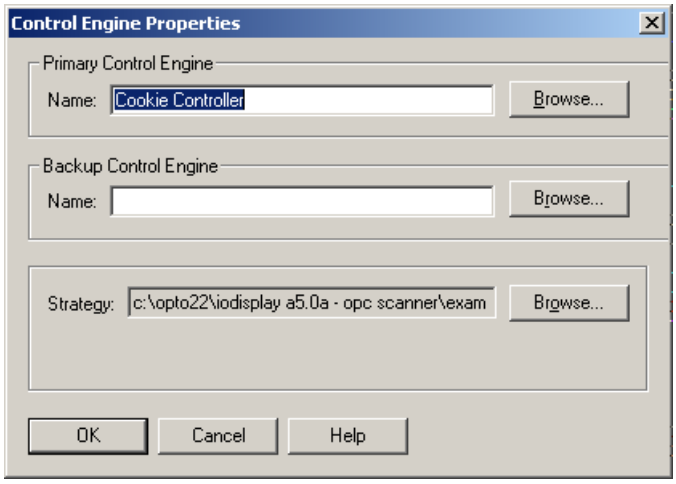


   All control engines that have been configured to connect to your PC are listed, whether or not they are associated with your strategy. If you previously configured a control engine for use with ioControl, for example, it would appear here, even if it didn't appear earlier when you opened the Control Engines dialog box.

   NOTE: If the control engine you want to use doesn't appear in the Select Control Engine dialog box, you must connect and configure this control engine to make it available. Instructions for adding, modifying, and deleting control engines appear in Chapter 4, "Working with Control Engines," in the ioControl User's Guide.

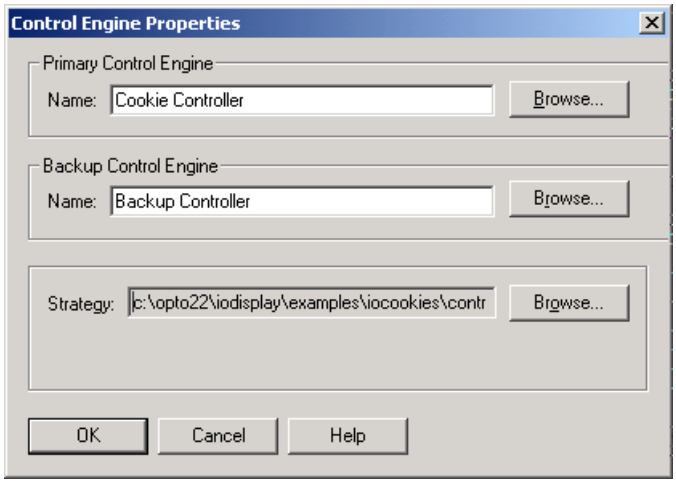
3. To choose a control engine that connects with ioDisplay, select its name and click OK.

   The primary control engine you have added appears in the Control Engine Properties dialog box:



4. To designate a backup control engine, click Browse in the Backup Control Engine group, choose a control engine from the list and click OK.

   The backup control engine you have added appears in the dialog box.



   A backup control engine is used automatically in case the primary control engine fails or becomes unavailable. Control is returned to the primary control engine when it becomes available again.

5. When all the parameters in the Control Engine Properties dialog box are correct, click OK to save your settings and close the dialog box.

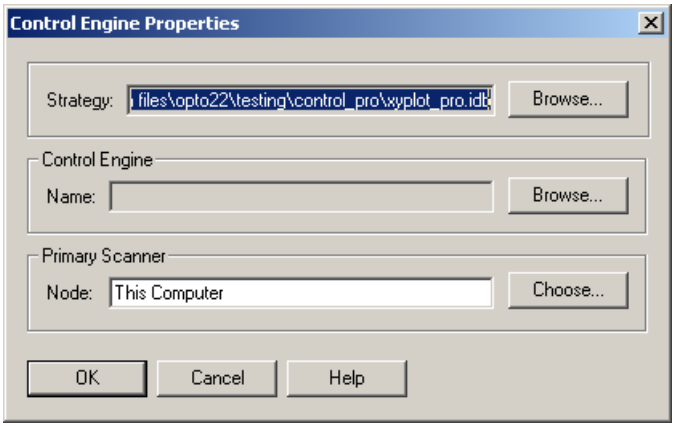
Local or Remote Scanner

When you configure the control engine in ioDisplay, you should also designate a local or remote scanner. In ioDisplay, a *scanner* is a software component that manages all communications with one or more control engines. By default, the scanner runs on the same PC running the ioDisplay project, but your application may benefit from configuring a *remote scanner.* See "Configuring the Scanner" on page 5-6 for more information.

Final Controller Configuration with ioDisplay Professional

1. Select the strategy file and click Open.

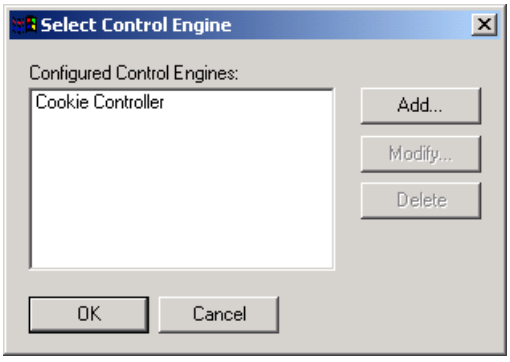
   The strategy you selected appears in the Strategy field of the Control Engine Properties dialog box:



   Now you need to select the primary control engine from which ioDisplay will receive I/O point information. *Remember that this control engine must be running the strategy you selected.*

2. Click the Browse button in the Primary Control Engine group.

   The Select Control Engine dialog box appears:

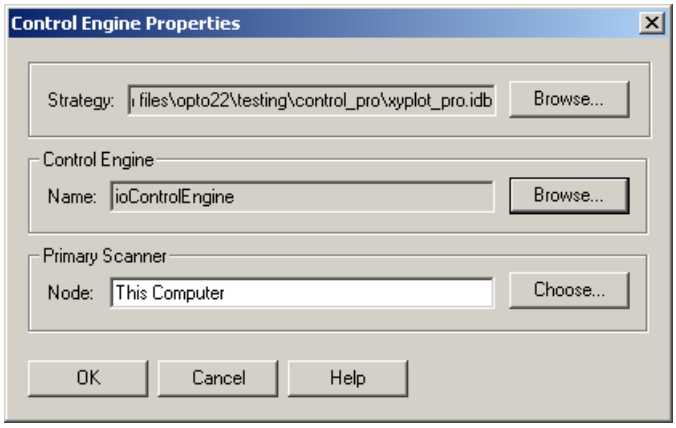


All control engines that have been configured to connect to your PC are listed, whether or not they are associated with your strategy. If you previously configured a control engine for use with ioControl, for example, it would appear here, even if it didn't appear earlier when you opened the Control Engines dialog box.

NOTE: If the control engine you want to use doesn't appear in the Select Control Engine dialog box, you must connect and configure this control engine to make it available. Instructions for adding, modifying, and deleting control engines appear in Chapter 4, "Working with Control Engines," in the ioControl User's Guide. *To add, modify, and delete FactoryFloor controllers, see the same chapter in the* OptoControl User's Guide *(Opto 22 form 724).*

3. To choose a control engine that connects with ioDisplay, select its name and click OK.

   The primary control engine you have added appears in the Control Engine Properties dialog box:



To complete setting up the control engine, select primary and backup scanners. Scanners are described in the next section, "Configuring the Scanner."

Configuring the Scanner

ioDisplay exchanges information with one or more Opto 22 controllers using a software component called a *scanner*. This scanner is automatically installed on the same computer as ioDisplay software, but can also run on a different computer. This is described in "Using OptoOPCServer as a Remote Scanner" on page 5-7.

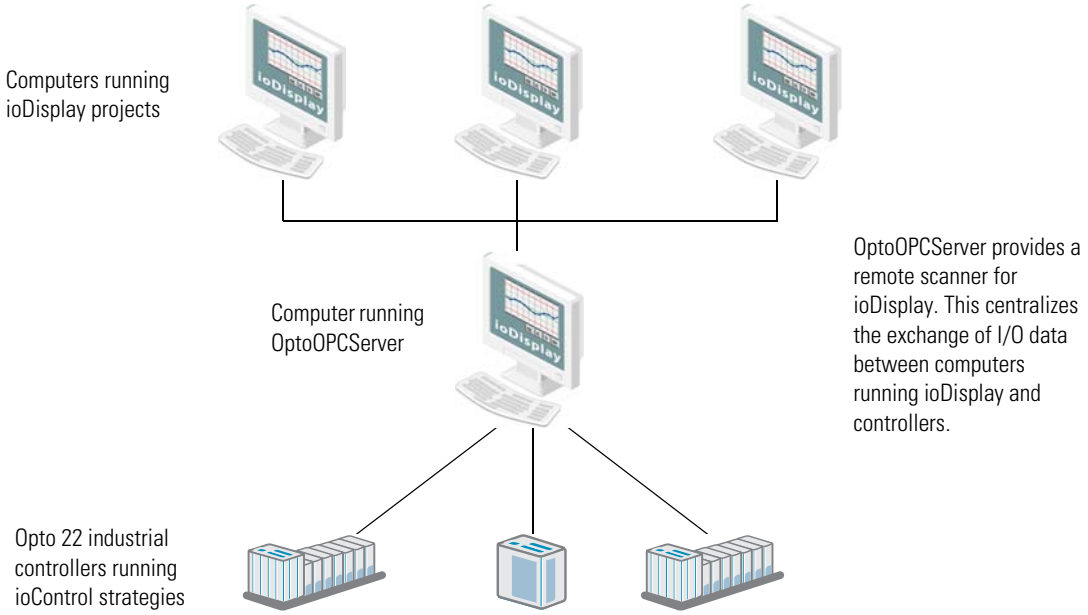
NOTE: No configuration is needed to run the scanner on the same computer as ioDisplay.

When an ioDisplay project is running, ioDisplay periodically checks to see if the scanner is operating. If the scanner is not operating, an error message is displayed in the Runtime Error Log. Configuring how often the scanner is checked is described in "Setting Scanner Heartbeat Interval" on page 5-9.

Using OptoOPCServer as a Remote Scanner

Different ioDisplay projects—or copies of the same project—can run on multiple computers which are connected on a network. For example, for supervisory reasons you may need to run several monitor-only versions of ioDisplay projects on office computers in addition to using regular monitor-and-control versions of ioDisplay at operator workstations.

If you need to run multiple instances of ioDisplay on separate computers, use Opto 22's OptoOPCServer software as a centralized scanner. Running on a PC or network server, OptoOPCServer centralizes communications with Opto 22 control engines. This greatly increases the efficiency with which computers running ioDisplay can exchange information with control engines. OptoOPCServer software is sold separately from ioDisplay, and is available from Opto 22 and local Opto 22 distributors.



NOTE: ioDisplay software can communicate only with its built-in scanner or with OptoOPCServer software. ioDisplay is not a generic OPC client, and cannot be used with third-party OPC servers.

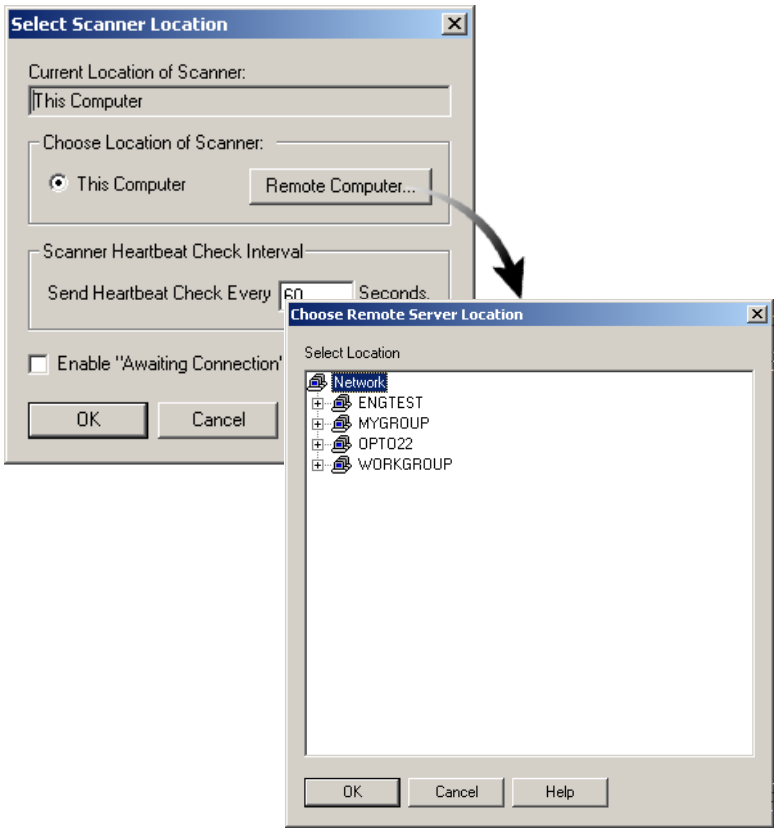
In addition to serving as a remote scanner for ioDisplay, OptoOPCServer supports third-party SCADA, HMI, data collection, and other OPC-client software. For these third-party OPC clients, OptoOPCServer can supply I/O point data from SNAP Simple I/O and SNAP Ethernet I/O systems that is not supported by ioDisplay.

OptoOPCServer must be correctly installed and configured on a network server or workstation before an ioDisplay project can use it as an scanner. See the *OptoOPCServer User's Guide* (Opto 22 form 1439) for detailed information on installing OptoOPCServer and configuring it for network operation. (This form is provided with the purchase of OptoOPCServer.)

Configuring Remote Scanner Location in ioDisplay

To use a remote computer running OptoOPCServer as a scanner, do the following:

1. In ioDisplay Configurator, select Configure→Scanner Location.
2. In the Select Scanner Location dialog box that opens, click Remote Computer.



3. In the list of computers that appears, navigate to the PC that has OptoOPCServer software installed and configured. Select the computer and click OK.

   IMPORTANT: *To successfully use OptoOPCServer on a remote computer with ioDisplay, Windows user and application permission settings on both local and remote computers must be correctly configured. See the* OptoOPCServer User's Guide (*Opto 22 form 1439) for detailed instructions for configuring OptoOPCServer and OPC clients for network operation.*

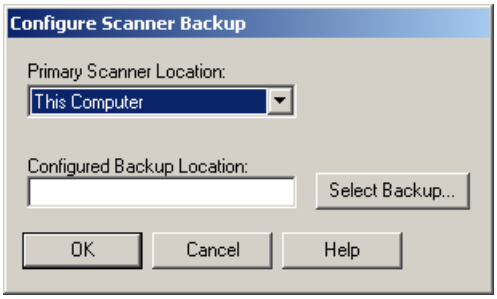
4. Click OK to close the dialog box and save your settings.

Configuring a Backup Scanner

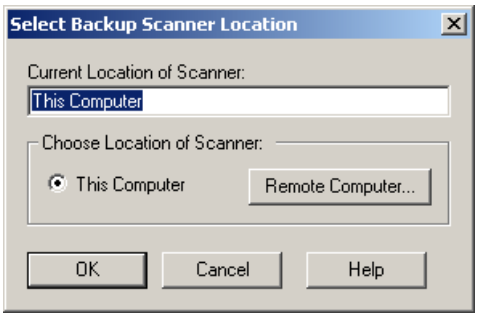
For each local or remote scanner the ioDisplay project uses, you can set up a backup scanner that will be used when the primary scanner is not available.

1. In ioDisplay Configurator, select Configure→Backup Scanner.

   The Configure Scanner Backup dialog box appears.



2. Select the primary scanner from the drop-down list in "Primary Scanner Location" and then click Select Backup.



3. In the Select Backup Scanner Location dialog box, click Remote Computer and select a computer from the list tree that you want to use for the backup scanner. Click OK when done.

4. Click OK to save your changes and close the Select Backup Scanner Location dialog box.

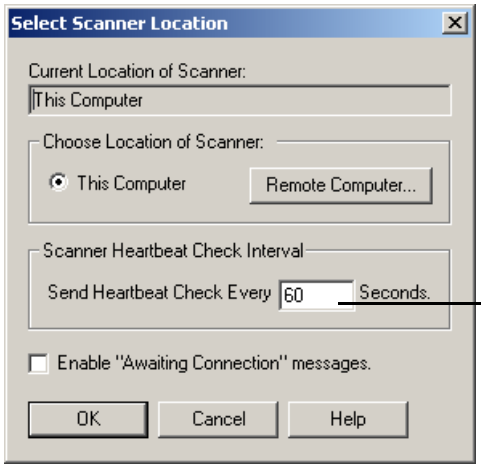
5. In the Configure Scanner Backup dialog box, click OK to save your changes.

Setting Scanner Heartbeat Interval

To set how often ioDisplay checks the operation of the scanner, do the following:

1. In ioDisplay Configurator, select Configure→Scanner Location.

The Select Scanner Location dialog box appears.



The scanner heartbeat check interval is the interval in seconds at which ioDisplay will check to see if the scanner is operating.

2. In "Scanner Heartbeat Check Interval," enter the interval in seconds at which ioDisplay will check to see if the scanner is operating.

   You can enter a value between 1 and 3600 seconds (integer values only). The default value of 10 seconds is suitable for most applications. If there is a slow communications link between the computer running the ioDisplay project and the computer running the scanner, use a longer heartbeat interval to reduce network traffic.

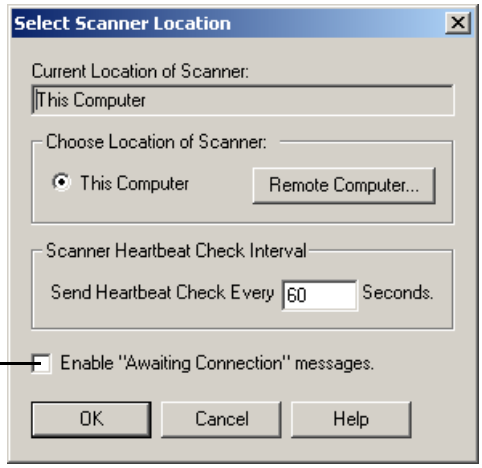
3. Click OK to close the dialog box and save your settings.

Hiding or Displaying Runtime Startup Messages

When an ioDisplay project starts in Runtime, multiple "Bad Quality" or "Not Connected" messages often appear in the Event Log Viewer. These startup messages appear when the ioDisplay scanner has not yet connected to a control engine specified in the project. To hide or display these Runtime startup messages, do the following:

1. In ioDisplay Configurator, select Configure→Scanner Location.

The Select Scanner Location dialog box appears.



Select the Enable "Awaiting Connection" Messages checkbox to display all startup messages. Deselect this checkbox to hide Bad Quality and Not Connected startup

2. Select or deselect the Enable "Awaiting Connection" Messages checkbox to display or hide any "Bad Quality" or "Not Connected" messages that may occur when the ioDisplay project starts.

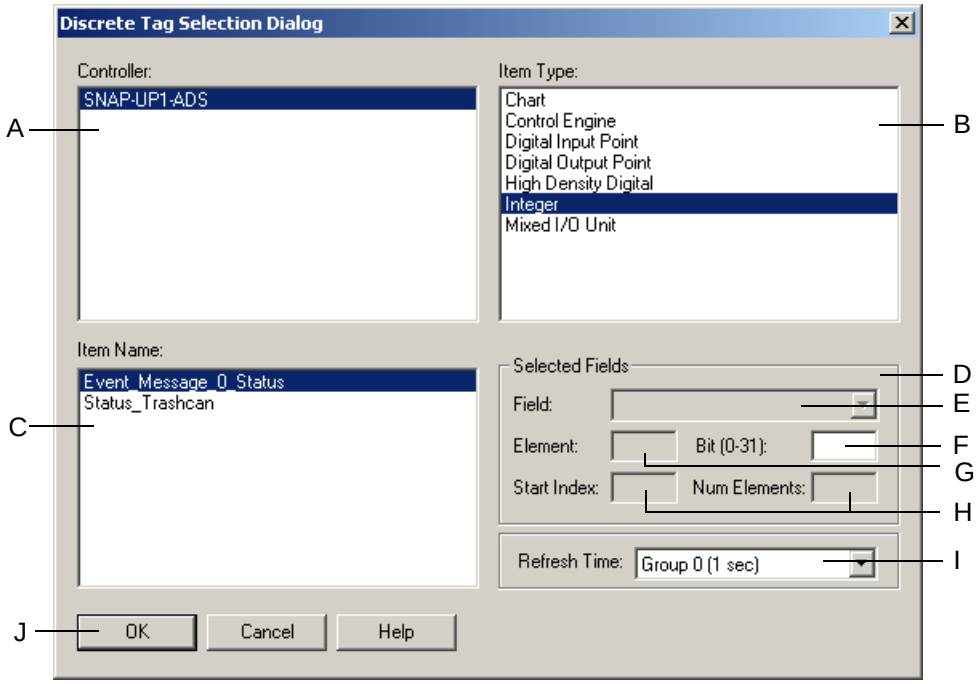
3. Click OK to close the dialog box and save your settings.

Configuring Tags

Graphics in ioDisplay can be linked directly to the values of ioControl tags, so you will configure tags quite often as you develop ioDisplay projects. Tags are configured using the Tag Selection dialog box, which you can access from many dialog boxes in ioDisplay Configurator by clicking the Tag Selection button ?.

The tags that appear in the Tag Selection dialog box are actively filtered; rather than display all possible item types and item names, only the item types available for the selected control engine appear in the Item Type list, and only the item names available for a selected item type appear in the Item Name list. For detailed information on item types and names in ioControl strategies, see the *ioControl User's Guide*.

To select a tag, complete the fields as follows:



A Select the control engine that contains the tag you wish to use. If only one control engine is available, it is automatically selected. Choosing a control engine updates the Item Type list (B) so that it displays only the ioControl data types available in that control engine's ioControl strategy.

B Select the type of data you wish to use. The list contains the data types available in the selected control engine's ioControl strategy. (The data types that appear depend on which data type was selected in the "Setup by" field in the preceding dialog box.) When you select a specific item type, a list of all the tags of that selection type appears in the Item Name list box (C). Your Item Type selection also determines the options available in the Selected Fields group (D).

ioDisplay can access tags for most types of variables and other information in an ioControl or OptoControl strategy. See the *ioControl User's Guide* or *OptoControl User's Guide* for information on a specific type of variable.

Pro

ioDisplay can also access memory-map-based Scratch Pad variables in SNAP Ultimate controller/brains, the SNAP-LCE controller, and SNAP PAC controllers; see *"Selecting Tags for I/O Unit Scratch Pad Variables" on page 5-15* for information on configuring Scratch Pad variables in the Tag Selection dialog box.

C This is an alphabetical list of the available ioControl tags of the type specified in the Item Type list. Select the tag you want to use from this list.

D The item type of the tag you select determines which of these fields, if any, need to have contents specified. If an entry is not needed, then the option is not available.

IMPORTANT: *There are differences in how some tags are configured in this dialog box.*

- *If SNAP high-density digital modules are used in your I/O system, see "Selecting Tags for SNAP High-Density Digital Modules" on page 5-13 for more information.*
- *If the Scratch Pad feature available in some SNAP brains is used to store bit, integer, floating point, or string values, see "Selecting Tags for I/O Unit Scratch Pad Variables" on page 5-15 for more information.*

E Specifies the data that is associated with the selected tag. For example, if the selected tag is of Item Type Digital Input Point, the available field is State. If the tag Item Type is Float, the Field list box is disabled.

F If the base type is Integer, a specific bit in the range specified may be selected from the integer.

G If the selected Item Type is one of the Table types and only a single element of the table is desired, enter the index of that element in this field.

H To select multiple elements from Item Type Table, use the Start Index to specify the first element and Num Elements to specify how many.

I If a control engine-driven attribute is being edited, select the refresh time group to be used for scanning. All tags that are defined as part of the same refresh time group are scanned at the same time. A time group with a refresh time of 0 seconds is scanned as quickly as the speed of the communications link permits.

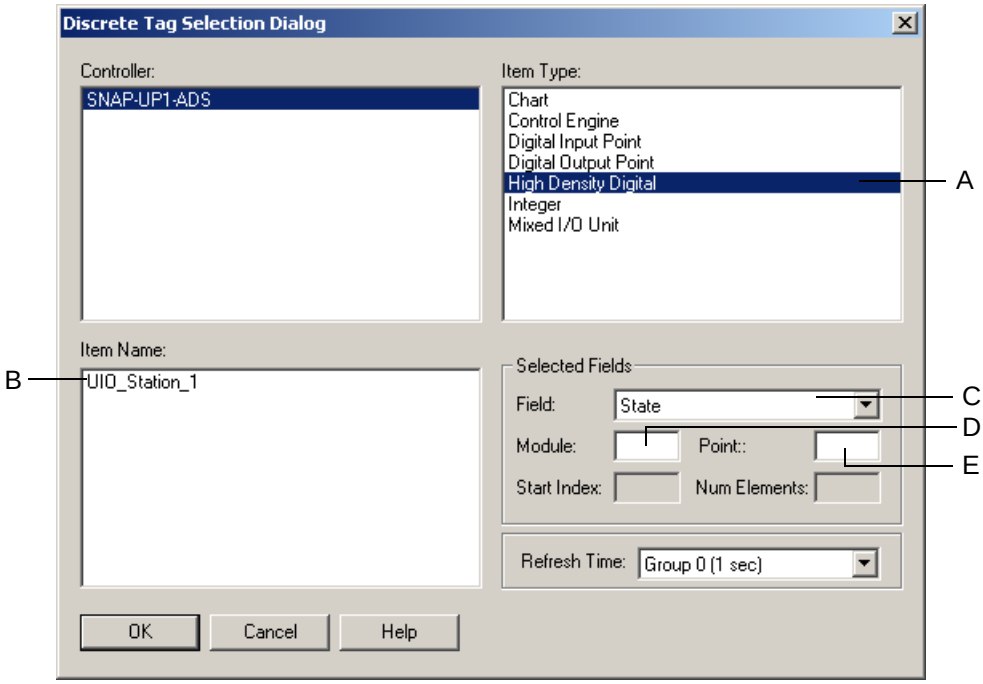
J Click OK to save your settings. (Click Cancel to close the dialog box without making any changes.)

Selecting Tags for SNAP High-Density Digital Modules

Tags for Opto 22's SNAP high-density digital modules are handled differently than other tags in an ioControl strategy. Tag selection dialog box fields for working with high-density digital module(s) in an ioDisplay project are described below.

High-density digital modules provide 32 digital input or output points in one SNAP module. See the *ioControl User's Guide* for information on using these modules in an ioControl strategy. See Opto 22 form 1547, the *SNAP High-Density Digital Module User's Guide* for more information.

To configure tags for a high-density digital module, do the following:



A Select the item type "High Density Digital" to see all I/O units configured in the ioControl strategy for your ioDisplay project.

If the ioControl strategy has at least one configured I/O unit, "High Density Digital" always appears in the Item Type list, even if a SNAP high-density digital module is not used in your I/O system.

B Select the name of the I/O unit where the SNAP high-density digital module you want to use is installed.

C Choose a field based on the type of dynamic attribute (controller-driven or operator-driven), and whether you want to work with a single point on the high-density module or all 32 points. One or more of the following fields will be available:

Field	Dynamic Attribute	Tag	Description	Must specify:	
				Module	Point
State	Controller-driven	Discrete	Returns on/off status of specified point.	●	●
		Value	Returns a 32-bit integer indicating status of all 32 points.	●	n/a
	Operator-driven	Discrete	Sends a true/set to set state and a false/clear to clear state.	●	●
		Value	Sends a 32-bit integer to set states of all 32 points.	●	n/a

Field	Dynamic Attribute	Tag	Description	Must specify:	
				Module	Point
Counts	Controller-driven	Discrete	n/a	n/a	n/a
		Value	Returns a 32-bit integer for a single point.	●	●
	Operator-driven	Discrete	Refer to the "Read and Clear" dynamic attribute on page 7-17.		
		Value			
Latch (ON)	Controller-driven	Discrete	Returns Latch (ON) status of specified point.	●	●
		Value	Returns a 32-bit integer indicating status of all 32 points.	●	n/a
	Operator-driven	Discrete	Clears ON latch. Choose "Set" to clear latch if "prompt for value" dialog box is used.	●	●
		Value	Sends a 32-bit integer to clear ON latches of all 32 points.	●	n/a
Latch (OFF)	Controller-driven	Discrete	Returns Latch (OFF) status of specified point.	●	●
		Value	Returns a 32-bit integer indicating status of all 32 points.	●	n/a
	Operator-driven	Discrete	Clears OFF latch. Choose "Set" to clear latch if "prompt for value" dialog box is used.	●	●
		Value	Sends a 32-bit integer to clear OFF latches of all 32 points.	●	n/a

D Enter the position (0–16) where the high-density digital module is installed on the mounting rack.

E Enter the number (0–31) of the module's digital input or output point.

Selecting Tags for I/O Unit Scratch Pad Variables

Opto 22 SNAP PAC, SNAP-LCE, and SNAP Ultimate I/O controllers have a section of memory called the Scratch Pad where bit, integer, floating point, and string values can be stored and accessed. The Scratch Pad is convenient for making information available to other Ethernet-enabled Opto 22 brains and controllers, as well as Opto 22 software and custom software applications written using Opto 22's OptoMMP Communication Toolkit. See the *ioManager User's Guide* (Opto 22 form 1440) and the *ioControl User's Guide* (Opto 22 form 1300) for information on using bit, integer, floating point, and string Scratch Pads.

NOTE: The SNAP controller or brain using the Scratch Pad must be defined as an I/O unit in the ioControl strategy that the ioDisplay project is using.

In ioDisplay, tags for Scratch Pad *integer* and *floating point* values are configured the same way as tags for an ioControl numeric table variable. Like a table variable, for these Scratch Pads you must specify an index number and the number of elements to be read. Scratch Pads for integer and floating point values can have up to 10,240 elements, but unlike ioControl table variables, the elements in these Scratch Pads are non-contiguous. This means that when you specify a range that exceeds 1024 elements (indices 0–1023), the range must be divided between two tags as shown below.

Item Name in Tag Selection Dialog Box	Description	First Element	Last Element
*IO_Unit_Name:*SP_INT	Scratch Pad integer value	0	1023
*IO_Unit_Name:*SP_INT_EXT	Scratch Pad integer value, extended	1024	10239
*IO_Unit_Name:*SP_FLOAT	Scratch Pad floating point value	0	1023
*IO_Unit_Name:*SP_FLOAT_EXT	Scratch Pad floating point value, extended	1024	10239

Example: The I/O unit labeled "Station_01" has floating point values stored in elements 0 to 2047. To display all these values using, for instance, ioDisplay's Numeric Table object you would need to do the following:

1. Create two separate Numeric Table objects.
2. In the Tag Selection dialog box for the first table object, select the Item Name "Station_01:SP_FLOAT."
3. For the Start Index value, enter 0, and for Number of Elements, enter 1024.
4. In the Tag Selection dialog box for the second table object, select the Item Name "Station_01:SP_FLOAT_EXT."
5. For the Start Index value, enter 1024, and for Number of Elements, enter 1024.

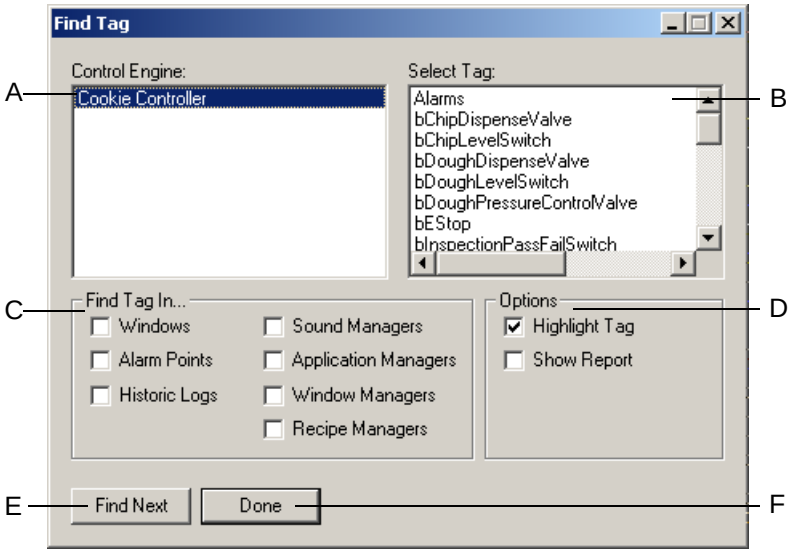
When the project is run in ioDisplay Runtime, the first table will show floating point values in indices 0 through 1023, while the second table will show values in indices 1024 through 2047.

Searching for Tags in an ioDisplay Project

You can search an ioDisplay project to find where an ioControl tag has been used.

1. Select View→Find Tag

The Find Tag dialog box appears.



A Select the control engine that contains the tag you wish to find. Choosing a control engine updates the Select Tag list (B) so that it displays only the tags available in that control engine's ioControl strategy.

B Select the tag you wish to find. The list contains only the tags defined in the selected control engine's ioControl strategy.

C Select one or more ioDisplay objects to search for the tag selected in (B). For example, if Windows is selected, then any graphic in a project window will be searched for the selected tag.

D Format and display your search results using the following options:

- Select Highlight Tag to automatically select the object containing the tag being searched for.
- Select Show Report to automatically open in Windows Notepad a report listing all tags found. The report is displayed after the search is complete.

E Click Find Next to start searching for a tag. After a tag is located, click Find Next again to continue searching. When no more tags are found, the Find Tag dialog box will close. If Show Report in (D) is selected, the report of tags found will be displayed in Windows Notepad.

F Click Done to close the Find Tag dialog box.

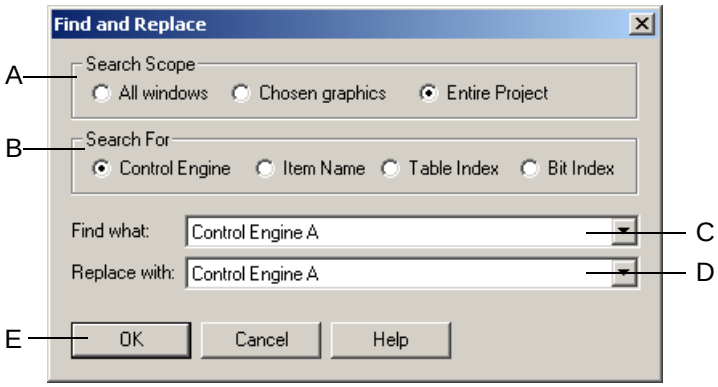
Finding and Replacing Tags in an ioDisplay Project

You can find and replace tags in an entire ioDisplay project, or just in one or more selected graphics. This find and replace feature works with tags for a control engine, item name, table index, or bit index.

1. In a draw window in ioDisplay Configurator, select the graphics you want to search and replace tags in. To search graphics in the entire ioDisplay project, select at least one graphic on the screen.

2. Select Edit→Replace. (You can also right-click on a graphic and select Replace from the pop-up menu.)

   The Find and Replace dialog box appears.



A Select "All windows" to search and replace in all windows in the ioDisplay project, "Chosen graphics" to search and replace tags in just the graphics you've selected, or "Entire Project" to search and replace tags in all parts of the ioDisplay project.

B Select the type of tag to search and replace.

C Enter the name of the tag to find. If "Control Engine" is selected in (B), you can choose the name of the control engine from a drop-down menu. For the other tag types, enter the name of the tag exactly as it appears in the ioControl strategy.

D Enter the name of the tag to replace the tag found in (C). The replacement item must be the same kind of item as the item being searched for. If "Control Engine" is selected in (B), you can choose the name of the control engine from a drop-down menu. For the other tag types, enter the name of the tag exactly as it appears in the ioControl strategy.

E Click OK to start the search, or click Cancel to close the dialog box without any changes.

Correcting Tags from a Strategy

When you select an ioControl strategy to use with your ioDisplay project, ioDisplay automatically imports that strategy's tagname database. This is unlike most other HMI software applications, which require you to create a separate SCADA database in addition to the strategy or control program itself.

This tight connection with the strategy's tagname database, however, can sometimes cause problems when the current ioControl strategy used by an ioDisplay project is modified. ioDisplay may incorrectly read the tags associated with the resulting strategy. ioDisplay Configurator includes a feature called AutoCorrect Tags that fixes most tag errors that may occur this way.

The AutoCorrect Tags feature works by comparing all tagnames, IDs, and table index references that are used with dynamic attributes in the ioDisplay project. If discrepancies are found between the items in the tagname database and the ioDisplay project, the errors that can be corrected are fixed. Both corrected tags as well as those that could not be corrected are listed in the results report AutoCorrect Tags generates.

When To Use AutoCorrect Tags

It's generally advisable to use the AutoCorrect Tags option after making any changes to the ioControl strategy associated with your ioDisplay project. Specifically, you should run AutoCorrect Tags in the following situations:

- If you use an ioDisplay project that contains objects you imported or copied from an OptoDisplay project. (OptoDisplay is the HMI authoring tool for Opto 22's FactoryFloor Software Suite.)
- If you import a window created in a different ioDisplay project. (Window files can be exported from an ioDisplay project, saved as files, and imported into another project. See "Importing, Exporting, and Saving Draw Windows" on page 6-5 for information.)

There are some tag errors in an ioDisplay project that AutoCorrect Tags cannot fix. These errors include if you do either of the following:

- Delete a tag from a strategy
- Shorten the length of a table in a strategy

You may also get unreliable results if you delete a tag from a strategy, and then create a new tag with the same name.

Using AutoCorrect Tags

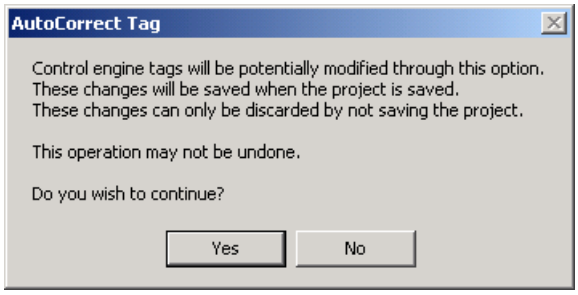
IMPORTANT: Always save changes to your ioDisplay project before using the AutoCorrect Tags option.

If you don't want to use the corrections made by AutoCorrect Tags, simply close the project without saving. Remember that not saving the project means you will lose any other changes you have made to the project.

Follow these steps to correct tags in your ioDisplay project:

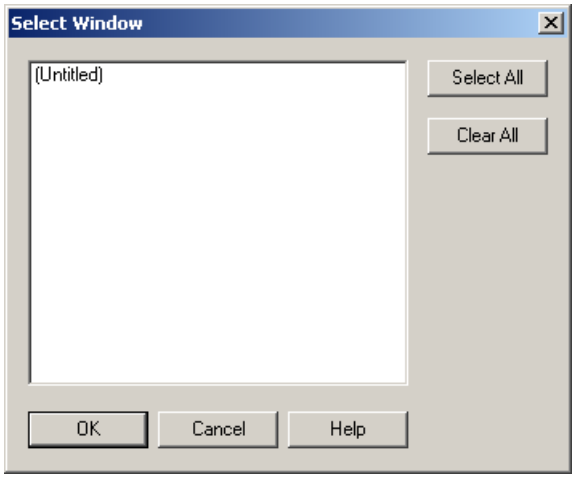
1. Select Tools→AutoCorrect Tags.

The following warning appears:



2. Click Yes.

   The Select Window dialog box appears.



3. Select one or more windows to run AutoCorrect Tags on.

   This is useful for large ioDisplay projects that may contain hundreds of windows, but only have a few windows that have been modified or imported.

4. If ioDisplay finds any problems with the tags and can fix them, it will do so. The changes, however, won't become a permanent part of your project until you actually save the project.

5. To correct tags, click Yes. (Click No to close the window and not make any changes.)

   The Windows application WordPad launches, displaying a results file that describes any problems AutoCorrect Tags may have found with the tags from the strategy. The results file has a name of the form Opto**n**.$$$, where **n** is an arbitrary number.

The following illustration shows a sample results file created by AutoCorrect Tags:

```
    Title:   ioDisplay AutoCorrect Tags Results File
    Project: C:\Opto22\ioDisplay\Examples\Factory.uui
A — File:    C:\Temp\Opto2.$$$
B — Date:    2001/10/03
    Time:    13:05:58
    Comment: This file is not deleted automatically.
             Summary information is provided at the end of this file.

C — Table length changed in strategy (tag corrected):
             Tag name: Marvin138:Float Table.RecipeFloatTbl(1)
             Old Length: 5
             New Length: 3
             Location: Window - master, Ellipse at 38,159, DynAttrColor

D — Tag not found in strategy (cannot correct):
             Tag name: Marvin138:Integer.HistoricLogTrigger2
             Location: Historic Log - Mass Storage, Start Trigger

E — Name changed in strategy (tag corrected):
             Old Tag Name: Marvin138:Integer.DOWNLOAD_TRIGGER
             New Tag Name: Marvin138:Integer.ACTIVATE_DOWNLOAD
             Location: Recipes - Peanut Butter Cookies, Trigger

F — Name changed in strategy (tag corrected):
    Table length changed in strategy (tag corrected):
    Index into table out of bounds (cannot correct):
             Old Tag Name: Marvin138:Integer Table.RecipeIntegerTbl(4)
             New Tag Name: Marvin138:Integer Table.RecipeIndex(4)
             Old Length: 5
             New Length: 3
             Location: Recipes - Peanut Butter Cookies, Notification
    -------------------------------------------------
             Number of tags changed: 4
G ———————— Number of tags not found: 1
             Number of tags index into tables out of bounds: 1
```

Here's an explanation of what the results file contains:

- A Name and location of the results file created by AutoCorrect Tags.
- B Date and time the file was created.
- C Warning message that reports that the table length of RecipeFloatTbl changed from an old length of five elements to a new length of three elements. "Location" shows where the tag was used in the ioDisplay project. In this example, it was found in a window called "master," attached to an ellipse at x: and y: coordinates of 38 and 159, with the color dynamic attribute.
- D Error message that the tag "HistoricLogTrigger2" is no longer part of the ioControl strategy, and this could not be corrected by the AutoCorrect Tags tool. The tag was used as the start trigger for a historic log called "Mass Storage."

  To fix this error, you'll have to assign another tag in your ioDisplay project to use as the start trigger for this historic log. (Note that ioDisplay won't recreate the connection to the tag if you open your ioControl strategy and add the old tagname again. Internally, ioDisplay can't correlate the old tagname and the new, similarly named tag.)

E Warning message that the tagname "DOWNLOAD_TRIGGER" changed to "ACTIVATE_DOWNLOAD" in the strategy. The old and new tagnames are reported, and the ioDisplay project is corrected to use the new tagname. The tag was used in a recipe called "Peanut Butter Cookies" as its trigger.

F Warning message that multiple errors have been found for one tagname:

- The table name "RecipeIntegerTbl" changed to "RecipeIndex," and its length changed from five elements to three elements. The ioDisplay project is updated with this change.
- An "index into table out of bounds" error was detected and couldn't be corrected. Specifically, the project tried to use the fifth element of the table (RecipeIntegerTbl[4]), but the fifth element no longer exists. To correct this problem you must specify a valid index. The tag was used in a recipe called "Peanut Butter Cookies." When the recipe is successfully downloaded, ioDisplay writes a value to this notification tag.

G The final tally of all the warnings and errors found by AutoCorrect Tags is reported here.

Note that if you run AutoCorrect Tags again, you will see only those errors that were reported as "cannot correct." The other reported errors have been corrected.

When you no longer need the Opto**n**.$$$ file, you can delete the file from your hard drive.

CHAPTER 6

Working with Graphics

Introduction

This chapter describes how to use ioDisplay Configurator to create and edit graphic objects, and how to import images. It also describes how to configure the windows in which these items appear.

In This Chapter

Using Draw Windows

Draw windows in ioDisplay Configurator are “blank pages” where you place, create or edit all the graphics for your ioDisplay project. These windows also contain trend, SuperTrend, and alarm objects. When you create a draw window, you can define its size, position, and color, as well as its state (open, closed, or iconified) in which it appears when the project opens in ioDisplay Runtime.

Draw windows can be protected with a password, and be set up to open when an alarm is triggered. You can also configure whether menus, borders, and other standard window elements appear.

Creating and Deleting Draw Windows

When you create a new ioDisplay project, one draw window appears in the main project window. To create additional draw windows, you can create a new window, or copy an existing draw window and its attributes.

Making a New Draw Window

1. Select Windows→New.

   The Window Properties dialog box opens.

2. Enter a name for the window, and configure other settings as necessary.

   See "Modifying Draw Windows" below for instructions on configuring draw windows.

3. Click OK when done.

   The new draw window appears in the project window.

Copying an Existing Draw Window

1. Select Windows→Copy.

   The Window Properties dialog box opens.

2. Enter a new name for the window, and configure the existing settings if necessary.

   See "Modifying Draw Windows" below for instructions on configuring draw windows.

3. Click OK when done.

   The new draw window appears in the project window.

Deleting an Existing Draw Window

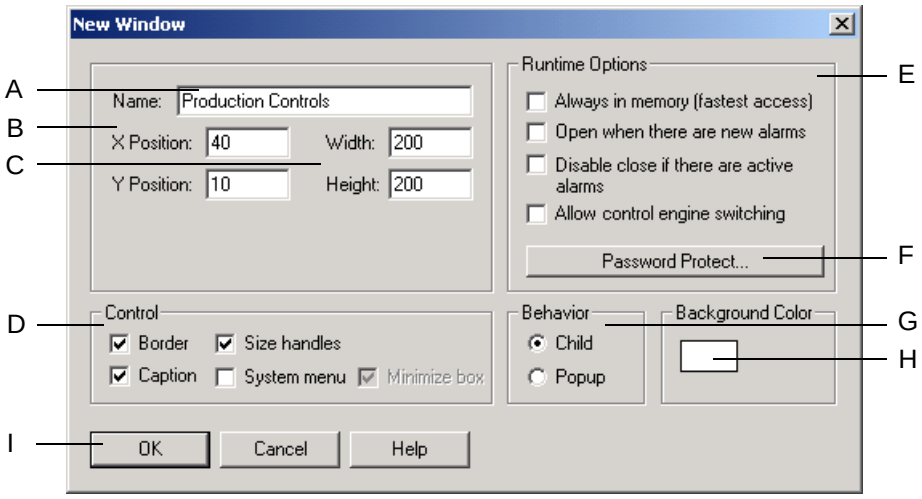
1. Using the Select tool, click on a draw window to select it.
2. Select Windows→Delete.
3. Click Yes in the message box that appears to confirm the deletion.

Modifying Draw Windows

You can change many properties of a new or existing draw window, such as the window's position, color, and behavior, among other properties. These properties are set in the Windows Properties dialog box. To open this dialog box, do one of the following:

- Create a new draw window or copy an existing one.
- Click on a draw window to select it, and then choose Windows→Properties.

The Window Properties dialog box appears.



A Enter the name of the draw window. This name appears in the window's title bar unless the title bar is hidden by deselecting the "Caption" option in the Control group (D).

B Enter the X and Y coordinates where the window appears in the project window. The X and Y coordinates indicate the location, in pixels, of the draw window's upper-left corner; the upper-left corner of the project window has X and Y coordinates of 0.

C Enter the width and height of the draw window. Width and height are measured in pixels.

D Use the options in the Control group to configure the appearance of the draw window. Select or deselect the following options:

Border—Hides or displays the narrow edge of the window. This option must be selected for the other options in the Control group to be available.

Caption—Hides or displays the bar at the top of the window where the window name appears. This option must be selected to move the window within the larger project window.

Size handles—If selected, lets you resize the window by clicking and dragging an edge or corner of the window.

System menu—Hides or displays the small system menu icon and the Close Window button located at the top of the window. This option is unavailable if the Caption option has not been selected.

Minimize box—Hides or displays the standard Windows close box in the upper-right corner of the window. This option is unavailable if the System menu option has not been selected.

E Use the items in the Runtime Options group to configure how the window opens when the project is run in ioDisplay Runtime. Select or deselect the following options:

Always in memory (fastest access)—If selected, the window's information is loaded and saved in the computer's memory when the project runs. Use this option for a window that you know will be opened and closed often. This option is selected automatically if you place an alarm graphic in a window.

Normally, only windows that have been opened or iconified are saved in memory. If you use this option with many windows, more computer memory is required, and your ioDisplay project will require more time to start up. Using this option with fewer windows uses less memory, and your project will start up more quickly.

Open when there are new alarms—If selected, a closed or iconified window that contains an alarm graphic will open when the alarm is triggered. This option is only available if a window contains an alarm graphic that is set to summary or detailed view. For more information on configuring alarms, see "Adding Alarm Graphics" on page 9-37.

Disable close if there are active alarms—If selected, prevents an open window that contains one or more active alarms from being closed. All active alarms in a window must be acknowledged before the window can be closed. For more information on working with alarms, see "Working with Alarms" on page 10-19.

Allow control engine switching—If selected, all graphics in the window can use data from the same ioControl strategy running on a different control engine. The operator switches between control engines in Runtime. For more information on using data from multiple control engines, see "Switching a Window between Control Engines" on page 10-18.

F To assign a password to a window, click here and then enter a password in the dialog box that appears. When a password is assigned to a draw window, a closed window cannot be opened without first entering the password. (Open windows that are iconified or hidden are not affected.)

You cannot assign a password to a window that both contains an alarm and has the Runtime option "Open when there are new alarms" selected.

G Use the options in the Behavior group to set how the window appears on-screen when the project is run in ioDisplay Runtime. Select one of the following options:

Child—prevents the window from being moved or minimized outside the boundaries of the main project window.

Popup—lets the window be moved or minimized outside of the main project window.

H To set the background color of a window, click the color square and then select a color in the dialog box that appears.

I Click OK to save your settings.

Opening and Closing Draw Windows

To open or close a draw window in your ioDisplay project, do the following:

1. From the Windows menu, select Open or Close.

   The Open Windows or Close Windows dialog box appears.

2. Click the name of the window you want to open or close, and then click OK.

   See "Working with Multiple Windows" on page 6-5 to learn how to select or deselect multiple window names.

Other ways of opening and closing windows include the following:

- To open a window that has been opened previously, select its name from the bottom of the Windows menu.
- To close a window in which the system menu appears, click the Close Window button in the upper-right corner.

Working with Multiple Windows

There are several ways to select or deselect multiple window names in the Open Windows or Close Windows dialog box.

- To select individual window names, hold down the SHIFT key and click each window name.
- To highlight all names in the list, click Select All.
- To not select any name in the list, click Deselect All.
- To easily close or open all but one window, click on a single name and then click Inverse.

Importing, Exporting, and Saving Draw Windows

An ioDisplay window can be exported from one project, saved as a file, and then imported into another ioDisplay project. The exported window file contains all the objects and tags that were in the original window. Exporting and importing draw windows is a convenient way to reuse the same window in different ioDisplay projects.

Exporting Windows

To export a window in an ioDisplay project, do the following:

1. Select the window in your ioDisplay project that you want to export.

   The window must be open within the Configurator window.

2. Select Window→Export Window.

   The Export Window As dialog box opens.

3. Navigate to the location where you want to save the exported window file and then click Save.

Importing Windows

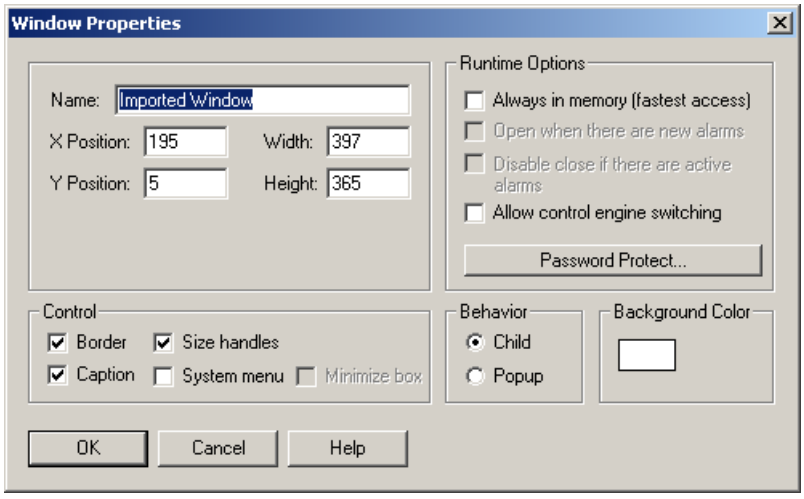
To import a window in an ioDisplay project, do the following:

1. Select Window→Import Window.

   The Import Window dialog box opens.

2. Navigate to the location where the exported window file is saved, select the file, and then click Open.

   The Windows Properties dialog box appears.



3. In the Name field, change the default name "Imported Window" to a suitable name for your project.

4. Change other properties of the window if needed, and then click OK. See "Modifying Draw Windows" on page 6-2 for more information on changing window properties.

   The window is added to the ioDisplay project.

5. If the new window is not visible in the project window, select Window→Open, choose the window in the list that appears, and then click OK.

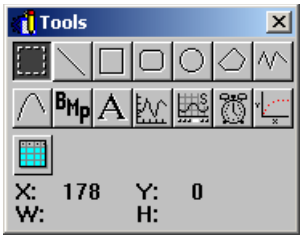
NOTE: When a window is imported into an ioDisplay project, the tags used by objects in that window are not verified for accuracy. Run AutoCorrect tags on the window after importing it. See "When To Use AutoCorrect Tags" on page 5-19 for instructions.

Drawing Graphic Objects

Once you've opened a project in ioDisplay Configurator, you can use the drawing tools in the ioDisplay toolbox to create graphic objects in your active window.

Follow these general steps to draw a graphic object:

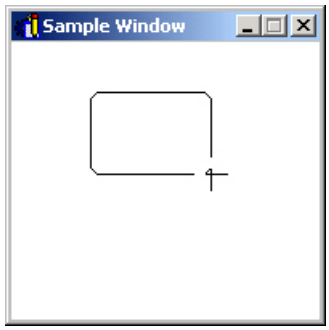
1. If your toolbox is not visible, select View→Toolbox to see the toolbox as it appears below. (You can close the toolbox by selecting View→Hide Toolbox.)



2. Select a drawing tool from the toolbox by clicking on it.

   The cursor will turn into a crosshair.

3. Click the crosshair in a window, then drag it in any direction to create a graphic.



In the example above, the Round Rectangle tool  was selected.

Tool Shortcut Keys and Key Combinations

There are two ways to use keyboard keys with tools in the toolbox:

- **Shortcut Keys**—Each drawing tool has a shortcut key associated with it to make selecting and switching between tools easier and faster. To choose a tool, just press the corresponding key on the keyboard.
- **Key Combinations**—By pressing various key combinations when using a drawing tool, you can alter the appearance of the graphic object you draw.

For each tool in the toolbox, the table below lists the icon, description, shortcut key, use, and key combination(s) available for that tool.

Tool	Description	Use
	Select tool (ENTER OR SPACEBAR)	Used to select, move, and resize graphic objects.
	Line tool (\ )	Draws straight lines.
	CTRL+Line tool	Draws constrained straight lines at angles of 90 degrees.

Tool	Description	Use
	Rectangle tool ([ )	Draws squares and rectangles.
	CTRL+Rectangle tool	Draws squares with the reference point in the top left corner.
	SHIFT+Rectangle tool	Draws rectangles with the reference point in the center.
	SHIFT+CTRL+Rectangle tool	Draws squares with the reference point in the center.
	Round Rectangle tool (R)	Draws squares and rectangles with rounded corners.
	CTRL+Round Rectangle tool	Draws squares with rounded corners with the reference point in the top left corner.
	SHIFT+Round Rectangle tool	Draws rectangles with rounded corners with the reference point in the center.
	SHIFT+CTRL+Round Rectangle tool	Draws squares with rounded corners with the reference point in the center.
	Ellipse tool (E)	Used to draw circles and ellipses.
	CTRL+Ellipse tool	Draws circles with the reference point in the top left corner.
	SHIFT+Ellipse tool	Draws ellipses with the reference point in the center.
	SHIFT+CTRL+Ellipse tool	Draws circles with the reference point in the center.
	Polygon tool (P)	Used to draw polygons as follows: • Drag and click to form vertex points. • Double-click the last vertex to close the polygon. Sizing handles appear around the object. If you can't select the polygon later with the Select tool, the polygon is not complete. Refresh the window (View→Redraw) to remove incomplete graphics.
	Polyline tool (L)	Used to draw connected lines as follows: • Drag and click to draw connected lines. • Double-click on the last line drawn to finish the polyline. Sizing handles appear around the object. If you can't select the polyline later with the Select tool, the polyline is not complete. Refresh the window (View→Redraw) to remove incomplete graphics.

Tool	Description	Use
	Bezier Curve tool (C)	Used to draw curves as follows: • Click at least four points in the draw window to draw a curve. • Thereafter, click points one at a time in groups of three to continue drawing the curve. • Double-click the last point to finish the curve. Sizing handles appear around the object. If you can't select the curve later with the Select tool, the curve is not complete. Refresh the window (View→Redraw) to remove incomplete graphics.
	Bitmap tool (B)	Used to place bitmap graphic files selected with File→Choose Bitmap. Once the bitmap has been chosen, just click to place the bitmap in the window.
	Text tool (T)	Used to write text for labels and titles. Put your cursor where you want the text to start. Type your text. When finished, click the mouse away from the text you just typed or press ENTER. Modify text by selecting it with the Select tool and by using Edit→Edit Text. The text tool is also used to create text objects which, when configured with the Text-In (from Control Engine) dynamic attribute, are used for viewing strings and other values from the control engine. See "Text In (from control engine)" on page 7-24 for more information.
	Trend tool (G)	Used to draw graphs that display real-time data against time. See Chapter 8, "Working with Trends" for more information.
	SuperTrend tool (Y)	Used to draw graphs that display both real-time and historical data against time. See Chapter 8, "Working with Trends" for more information.
	Alarm tool (A)	Used to draw objects that display alarms and other status information. See Chapter 9, "Configuring Trigger-Based Events" for more information.

Tool	Description	Use
	XY Plot tool (X)	Used to draw graphs that display data plotted on x and y axes. See Chapter 8, "Working with Trends" for more information.
	Numeric Table tool (Q)	Used to display data from up to four numeric tables. See "Working with Numeric Tables" on page 6-24 for more information.

Selecting Graphic Objects

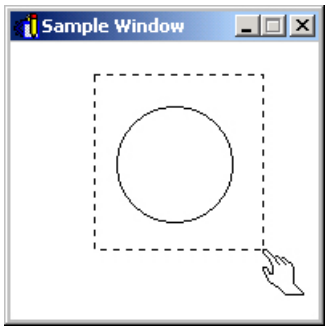
You can use the Select tool to choose one or more graphic objects in a draw window.

Selecting One Object

The simplest way to select a graphic object in a window is to just click on it with the Select tool. You can also select a graphic by clicking and dragging.

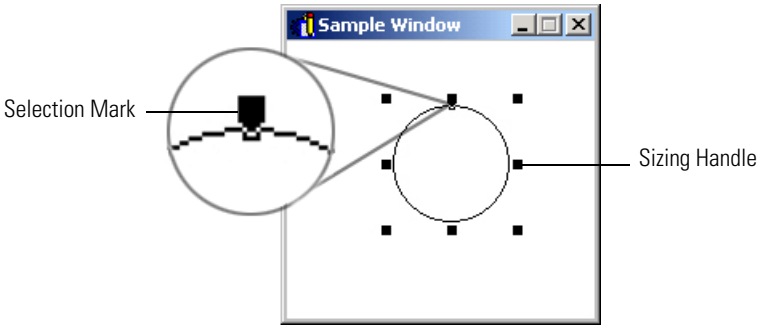
1. Choose the Select tool  from the toolbox.
2. Click the cursor just outside the graphic object you want to select and drag the cursor completely over the object.

   A selection box should appear as shown below:



NOTE: If you drag the cursor from left to right, you must completely enclose the object in the selection box to select it. However, you can also select an object by dragging the cursor from right to left over only part of the object.

3. After you release the mouse button, several sizing handles and one selection mark appear around the selected object.



Handles and Selection Marks

When a graphic object is selected, several solid black sizing handles appear, along with one transparent selection mark. The selection mark shows you whether you've selected a single graphic object or several objects in a group. Each graphic object in a group has a selection mark. Sizing handles can be used to resize the graphic, which we'll talk about a little later in this chapter.

Selecting Several Objects

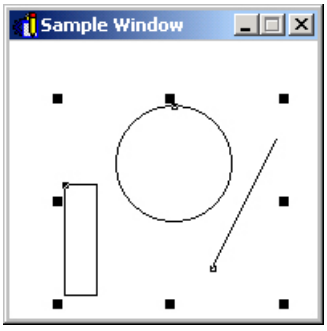
There are a few ways to select multiple graphic objects. The simplest way is to click and drag.

1. Choose the Select tool from the toolbox.

2. Click the pointer just outside the objects you want to select and drag the pointer across the group of objects you want to select.

   Make sure that you include all the objects within the selection box that appears.

   After you release the mouse button, several sizing handles and one selection mark will appear around the selected objects.



Another way to select several objects is to choose the first graphic using the Select tool, then hold down the SHIFT key and click on each additional object you want included in the selection group. Notice that a selection handle appears on each object you add to your group of objects.

Selecting All Objects

To select all the graphic objects that appear in your active window, choose Edit→Select All (ALT+E+S). You can also right-click and choose Select All from the pop-up menu.

Deselecting One or More Objects

The easiest way to deselect one or all graphics is to click anywhere outside the sizing handles. All of the handles disappear and no graphics are selected.

From a selected group of objects, you may need to pick some graphics you actually want as part of a final selection group. You can do this using key combinations.

- To deselect an object within a group of selected objects, hold down the SHIFT key and click on the object you don't want to include.
- To select only one object within a group of selected objects, hold down the CTRL key and click on the object. This deselects all other objects.

Grouping and Locking Graphics

You can combine two or more graphic objects into a group so that they are handled as one object. You can then manipulate the graphic as one unit. As a unit, the grouped graphics can be selected, moved, resized, and have dynamic attributes assigned. You can also lock the position of a graphic in a draw window so it can't be moved.

CAUTION: If you group objects, ioDisplay Runtime processes only the dynamic attributes assigned to the group. Dynamic attributes assigned to individual members of a group are ignored. If a group is later ungrouped, any previously configured dynamic attributes of the individual graphics will be recognized and processed in Runtime.

Grouping Objects

1. Select two or more graphics.
2. Choose Edit→Group. (You can also right-click and choose Group from the pop-up menu.)

   There will be no visible change, but the objects are collected into one group.

Ungrouping Objects

1. Select a set of graphics that were previously grouped.
2. Choose Edit→Ungroup, or right-click and choose Ungroup from the pop-up menu.

   You will see the sizing handles still appear around the former group. Click off the graphics and then click on an individual graphic. You will see it's not part of the group anymore.

Locking Objects in a Draw Window

After you've arranged several objects in a draw window, sometimes it can be useful to lock the position of one or more items so they aren't accidentally moved. To lock one or more objects, select the item(s) and choose Edit→Lock Position. To unlock objects, select the item(s) and choose Edit→Unlock Position.

Changing Lines and Fills

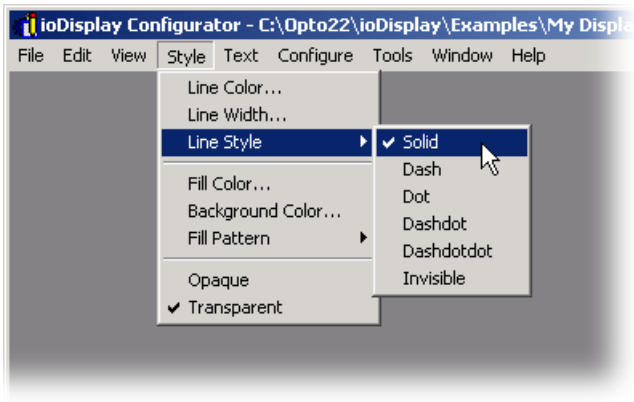
You can change the line and fill style of objects you've already drawn, or set the default for objects you're about to draw.

NOTE: If you select more than one object and the graphics have different line or fill attributes, no attributes will appear in the menus and dialog boxes for each attribute. However, you can still select new attributes for all selected lines.

Applying or Changing Line Attributes

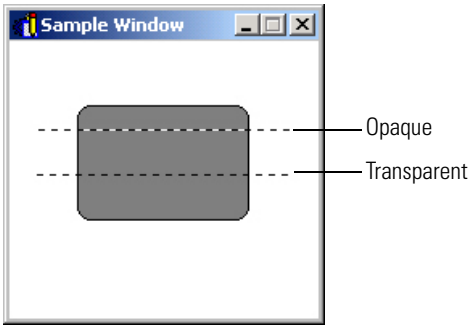
1. Do one of the following:
   - To change attributes for one or more existing objects, select the object(s) using the Select tool.
   - To set attributes for subsequent graphics that you create, do not select any object.
2. Choose the line attributes you want to apply to the selected object(s):
   - To change the line color, select Style→Line Color, then choose a color from the Color dialog box and click OK. (You can also right-click the object, then choose Line→Color from the pop-up menu.)
   - To change the line width, select Style→Line Width, then enter a pen width in the Pen Width dialog box and click OK. Pen widths are measured in pixels. (You can also right-click the object, then choose Line→Width from the pop-up menu.)

- To change the line style, select Style→Line Style, then choose a line style from the list that appears (shown below). (You can also right-click the object, choose Line→Style from the pop-up menu, then select a line style from the list that appears.)



- To change the opaque or transparent attributes of a line, select the appropriate attribute from the Style menu. Note that these attributes can be applied only to non-solid lines with a line width of one pixel.

  The transparent attribute lets an object's color show between the dashes in a line, and the opaque attribute lets the background color of the window show between the dashes in a line. Samples of opaque and transparent line attributes are shown in the example below.



Applying or Changing Fill Attributes

1. Do one of the following:
   - To change attributes for one or more existing objects, select the object(s) using the Select tool.
   - To set attributes for subsequent graphics that you create, do not select any object.
2. Choose the fill attributes you want to apply to the selected object(s):

- To change the fill color, select Style→Fill Color, then choose a color from the Color dialog box and click OK. (You can also right-click, choose Fill→Color from the pop-up menu, choose a color from the Color dialog box, and click OK.)
- To change the fill pattern, select Style→Fill Pattern and choose a fill attribute. A fill color other than white needs to be in effect in order to see the new fill pattern. (You can also right-click, choose Fill→Pattern from the pop-up menu, and choose a pattern.)
- To change the background color used behind a fill pattern, select Style→Background Color, then choose a color from the Color dialog box and click OK. (You can also right-click, choose Background→Color from the pop-up menu, choose a color from the Color dialog box, and click OK.)

Importing Graphics

You can easily import bitmap graphics, Windows metafile graphics, and JPEG images into an ioDisplay window to enhance or add detail to your operator interface. For your convenience, ioDisplay also includes the Symbol Factory, a large library of graphics designed especially for industrial applications.

A bitmap graphic is a picture, drawing, or other image saved in Microsoft Windows BMP file format. Bitmap graphic files have the file extension .bmp. If you'd like to use bitmap graphics in your project which have been saved in another graphic file format, such as TIFF (file extension .tif), you must first convert the file to BMP format. Commercial and shareware applications that can do this are widely available; two popular commercial image editing and conversion applications for Microsoft Windows are Paint Shop Pro from Jasc Software, and Photoshop from Adobe Systems Incorporated.

A Windows metafile is a drawing saved in the Microsoft Windows metafile (WMF) format. Metafile graphics have the file extensions .wmf or .emf. Unlike a bitmap graphic, a metafile maintains its resolution when resized and will not appear jagged or blurred. There is a variety of clip art and other graphics available in WMF format; for example, Microsoft Office often includes an assortment of clip art in WMF format.

The Joint Photographic Experts Group (JPEG) file format is a highly compressed format commonly used for photographs. This format is often used for displaying images in a Web browser.

Importing a Bitmap Graphic

To import a bitmap graphic into your ioDisplay project, first choose the bitmap image you want to use, and then use the Bitmap tool to place the bitmap in the window.

1. Select File→Choose Bitmap.
2. In the Choose A Bitmap dialog box that appears, navigate to the folder that contains the bitmap graphic you want to use, and then select the file name and click Open.

3. Select the Bitmap tool  in the toolbox and click the pointer in the desired location in the window.

   The bitmap is centered at the location you have clicked.

Importing a Metafile or JPEG Graphic

To import a Windows metafile or JPEG graphic into your ioDisplay project, do the following:

1. Select Edit→Paste from File→Import (Metafile or JPEG). (You can also right-click and select Import Metafile or Import JPEG from the pop-up menu.)

2. In the Import dialog box that appears, navigate to the folder that contains the graphic you want to use, and then select the file name and click Open.

   The selected file is placed in the active draw window.

You can also import a metafile or JPEG graphic used in another program by copying or cutting the graphic to the Windows clipboard, and then pasting it in the project draw window.

Importing a Graphic from the Symbol Factory

To import a graphic as a metafile graphic from the Symbol Factory into your ioDisplay project, do the following:

1. Select Edit→Paste from File→Symbol Factory. (You can also right-click and select Symbol Factory from the pop-up menu.)

   The Symbol Factory window opens.

2. Browse through the categories and thumbnails of graphics until you find the graphic that you want to use.

3. Click the graphic and drag it into the ioDisplay draw window.

   You can also select Edit→Paste as Picture (.wmf) Only.

The selected graphic is now available as a metafile graphic in the active draw window. The Symbol Factory window will remain open until you close it, or until you exit ioDisplay Configurator.

Bitmap Graphics in Symbol Factory

Although Symbol Factory graphics are provided as metafiles, you can also import these graphics as bitmaps. To import a graphic as a bitmap graphic from the Symbol Factory into your ioDisplay project, do the following:

1. Select Edit→Paste from File→Symbol Factory. (You can also right-click and select Symbol Factory from the pop-up menu.)

The Symbol Factory window opens.

2. Browse through the categories and thumbnails of graphics until you find the graphic that you want to use.
3. Click the graphic and then select Edit→Copy.
4. Switch to the ioDisplay draw window and select Edit→Paste.

The selected graphic is now available as a bitmap graphic in the active draw window. The Symbol Factory window will remain open until you close it, or until you exit ioDisplay Configurator.

Saving Objects as Bitmaps

After creating one or more objects in a draw window, you may want to save the object as a bitmap graphic file. This is useful, for example, if you want to document your operator interface.

NOTE: Saving a graphic object as a bitmap graphic file is not the same as copying an object, then pasting it into another window. When an object is saved as a bitmap, it loses all dynamic attributes and other properties it has been configured with.

1. Use the Select tool to select the graphic you want to save as a bitmap.

   If you don't select a graphic, the entire active window will be saved as a bitmap file.
2. Select File→Save as Bitmap.
3. In the Save As Bitmap dialog box that appears, navigate to the desired folder and enter a file name. (You can enter a three-letter extension other than .bmp, but the file will still be saved as a bitmap image.)
4. Click Save to save the image.

Copying, Duplicating, and Pasting

There are two ways that you can make copies of graphics you've created or added to a window. You can copy one or more graphics to the Windows clipboard, or duplicate the selected graphic(s) in the same window without affecting the contents of the clipboard.

You can also copy graphics into your ioDisplay project from another project, but both projects must be created with the same version of ioDisplay. Graphics created in an OptoDisplay project cannot be used in an ioDisplay project. (OptoDisplay is an HMI application similar to ioDisplay that is part of Opto 22's FactoryFloor software suite.)

Copying and Pasting an Object

1. Select one or more graphic objects.
2. Choose Edit→Copy. (The keyboard shortcut for this command is CTRL+C or CTRL+INS. You can also right-click and choose Copy from the pop-up menu.)

   The selected objects are copied to the Windows clipboard.
3. Click on the window where you want to paste the object(s).
4. Choose Edit→Paste. (The keyboard shortcut for this command is CTRL+V or CTRL+INS. You can also right-click and choose Paste from the pop-up menu.)

   The clipboard contents are pasted in the center of the active window.

Duplicating an Object

1. Select one or more objects.
2. Choose Edit→Duplicate. (The keyboard shortcut for this command is CTRL+D. You can also right-click and choose Duplicate from the pop-up menu.)

   A copy of the graphic is placed immediately below the selected graphic. Note that the contents of the Windows clipboard are not affected by duplicating an object.

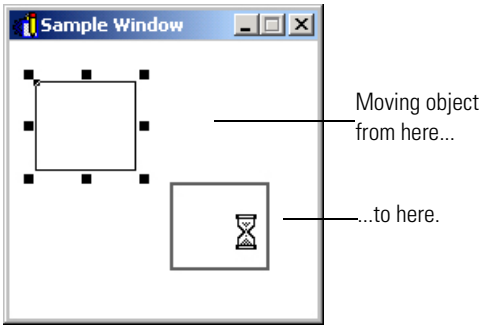
Moving and Resizing Graphics

To build your operator interface, you will need to be able to position, resize, and reshape graphic objects. This is done using the Select tool with different options. (Objects can also be positioned and resized by entering values in the Graphic Dynamic Attributes dialog box; see Chapter 7, "Using Animated Graphics," for more information.)

Moving Graphics

1. Choose one or more objects to move using the Select tool  .
2. To move an object, click the object (but not on a sizing handle) and drag it to the new position.

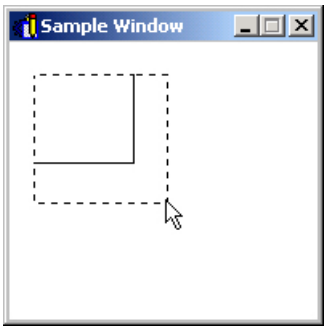
You can't drag objects from one window to another. You must copy or cut objects to move them between windows.



Resizing Graphics

1. Select one or more objects to resize using the Select tool .
2. To resize an object, position the pointer over a square handle, click, and drag.

   As illustrated below, the pointer turns into an arrow, and the size of the object changes relative to the sizing handle you're dragging.



3. When the object(s) are the size you want, release the mouse button.

Resizing Multiple Graphics to Equal Dimensions

When multiple graphics are selected, they can be resized equally so that all the graphics have the same height and width.

1. Using the Select tool , choose two or more objects to resize.
2. Select Edit→Size, then one of the following options:
   - **Grow to Largest Height**—All selected objects are resized to the height of the tallest object selected.
   - **Grow to Largest Width**—All selected objects are resized to the width of the widest object selected.

- **Shrink to Smallest Height**—All selected objects are resized to the height of the smallest object selected.
- **Shrink to Smallest Width**—All selected objects are resized to the width of the least wide object selected.

You can also select these options by right-clicking on the selected objects, selecting Size from the pop-up menu, then choosing a Grow or Shrink option.

Reshaping Graphics

You can adjust the individual points that make up a polyline, polygon, or Bezier curve object.

1. Using the Select tool  , select a polyline, polygon, or Bezier curve object.
2. Select Edit→Edit Points, or right-click and choose "Edit Points" from the pop-up menu.
3. Move the cursor over a point on the selected object.

   When the cursor is over a point on the object, the point is highlighted with a small black square.
4. Click the highlighted point, drag it to the desired location, and then release the mouse button.

   The highlighted point will appear in the new location.
5. When you are done adjusting points on the object, click outside of the object to deselect it.

Changing Stacking Order

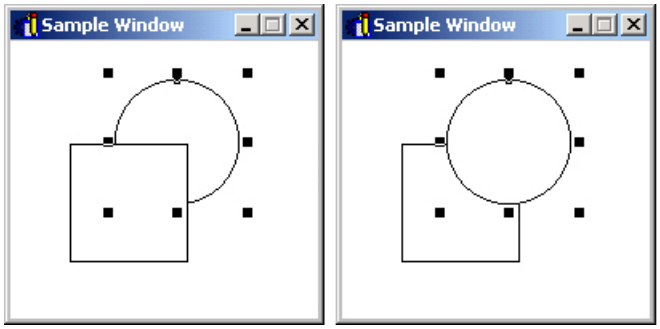
As objects are placed in draw windows, they're assigned a stacking order (or "Z-order") to define which object appears in front of or in back of another object. The position of objects in this stacking order can be modified as follows:

1. Select one object, or multiple objects that have been grouped together.
2. Choose Edit→Z-Order and select one of the following menu items:
   - Bring to Front
   - Move Forward
   - Send to Back
   - Move Backward

   You can also right-click on an object, choose Z-Order from the pop-up menu, and select a command.

   As shown below, if you select Bring to Front, the graphic is moved to the front of other graphics. Likewise, if you select Send to Back, the graphic is moved behind other graphics.

The commands Move Forward and Move Backward work similarly, except that the object you select is moved ahead or behind one graphic, not all graphics in the window.



Before Bring to Front After Bring to Front

Deleting Objects

There are several ways to delete an object in a window. Depending on the commands you use, the method you choose may affect the contents of the Windows clipboard.

1. Select one or more objects. (Use Edit→Select All to select all objects.)
2. Delete the graphics using one of the following methods:
   - To cut an object and save it in the Windows clipboard so you can use it elsewhere, choose Edit→Cut, or press SHIFT+DEL, or press CTRL+X, or right-click and choose Cut from the pop-up menu.

     You can paste the object into another window or elsewhere in the same window.
   - To permanently delete an object, choose Edit→Delete, or press the DEL key, or right-click and choose Delete from the pop-up menu.

     The objects are *not* copied to the clipboard and cannot be retrieved.

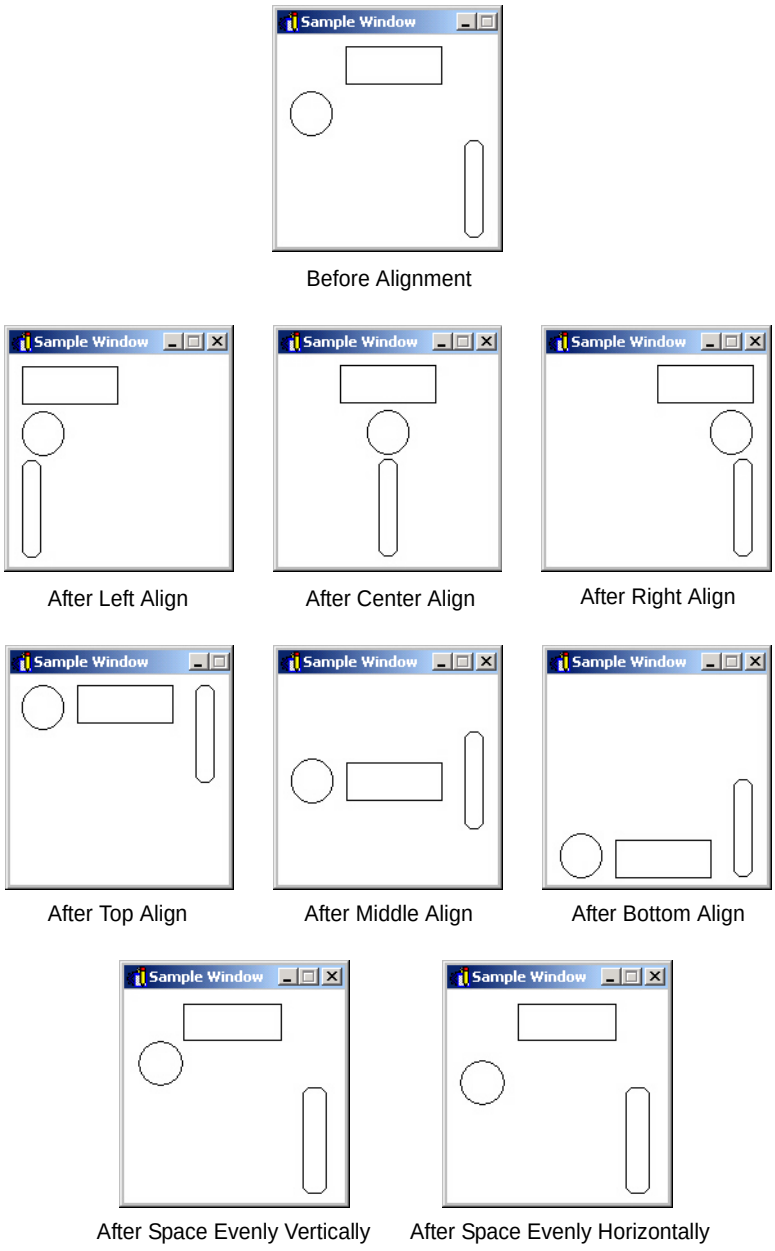
Aligning Graphics

You can align selected objects based on common edges, or based on common centers through objects. You can also adjust the space between objects.

1. Select the objects you want to align.

   You have to select at least two objects to enable this command.
2. Select the Edit→Align command and choose an option to base the alignment on. (You can also right-click and choose Align from the pop-up menu.)

The following illustration shows the results of applying the different alignment options:



Rotating and Flipping Graphics

You can rotate objects in 90° increments, or flip graphics around a central horizontal or vertical axis.

1. Select one or more objects.

2. Choose Edit→Flip/Rotate and select an option, or right-click and choose Flip/Rotate from the pop-up menu.
   - To flip a graphic from right to left (or vice versa), choose Flip Horizontal.
   - To flip a graphic from top to bottom (or vice versa), choose Flip Vertical.
   - To rotate a graphic clockwise 90°, select Rotate Clockwise.
   - To rotate a graphic counterclockwise 90°, select Rotate CounterClockwise.

Working with Text

The Text tool is a convenient way to label, title, and add impact to your graphics. Text that you add to a graphic can be changed at any time, and formatted using different fonts, font sizes, and colors.

IMPORTANT: *The text tool is also used to create objects that display values and string data from a control engine. Configure the text object you create using the Text In (from control engine) dynamic attribute; see "Text In (from control engine)" on page 7-24 for specific instructions.*

Adding Text

1. Choose the Text tool from the toolbox, or right-click and select Text from the pop-up menu.
2. Click the cursor where you want to place your text.

   You can also choose to place the text somewhere other than its final location, work on the text until it's ready to use, and then move it to the desired location.
3. Type the text.
4. When you're done with the text you want to type, click outside the text area.

   The text you've just typed is now an object, and you can select it and manipulate it like other objects.

Editing Text

1. With the Select tool, choose the text object you want to modify.
2. Choose Edit→Edit Text, or right-click and choose Edit Text from the pop-up menu.
3. Enter the new text in the Edit Text dialog box that appears, then click OK.

   The text object is now modified with the new text.

Formatting Text

1. With the Select tool, choose the text object you want to modify.
2. Choose a formatting option from the Text menu, or right-click the text object and select an option from the pop-up menu. The following formatting options are available:
   - **Font**—Changes the font family used for the text in a text object. You can use any fixed or TrueType font family installed on the computer.
   - **Size**—Changes the size of the characters in a text object.
   - **Color**—Defines the color in which the text appears.
   - **Background**—Defines the color of the area directly behind a text object. This formatting option is only visible when the Opaque attribute is selected.
   - **Style**—Changes the weight, italicization, and other characteristics of the characters in a text object. Styles available are Normal, **Bold**, *Italic,* Underline, and ~~Strikeout~~.

   The text object is now modified with the format changes.

See"Text Menu" on page D-12 of Appendix D, "ioDisplay Menu Reference" for additional information on creating and formatting a text object.

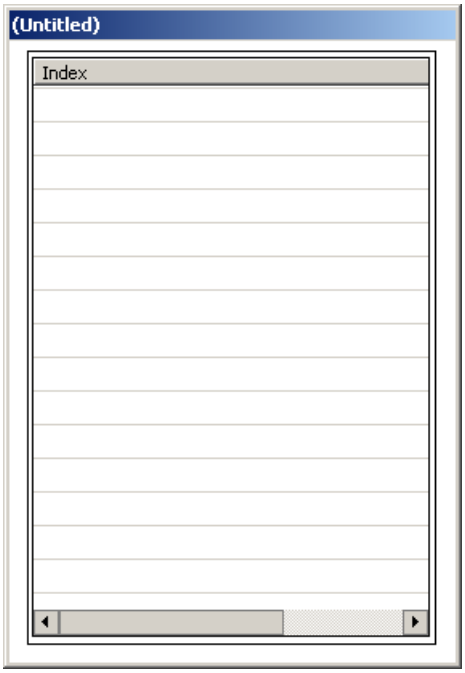
Working with Numeric Tables

Using ioDisplay's Numeric Table tool  , you can add an object to display the contents of numeric tables used in an ioControl project. In a single on-screen object, you can display the contents of up to four separate numeric tables. The tables can contain either 32-bit integers or floating point values.

Creating a Numeric Table

1. Select the Numeric Table tool in the toolbox.
2. Click the mouse button, drag the mouse to the desired size, and then release the mouse button.

The numeric table object that appears should resemble the example below:

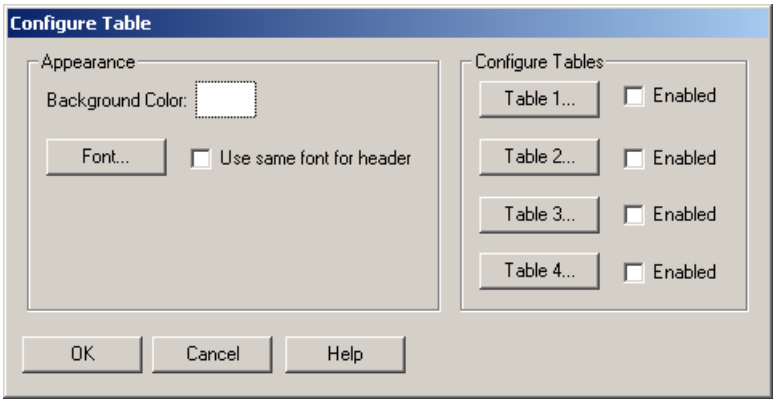


Configuring a Numeric Table

After creating a numeric table object in a draw window, you must specify which table(s) will be displayed. For each table, you can optionally specify which table elements will be displayed.

1. Double-click the numeric table object with the Select tool.

   The Configure Table dialog box appears.



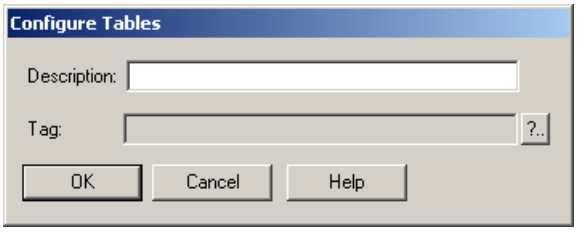
2. In the Appearance section, choose the background color and font used in the numeric table object.
   - To select background color, click the white square next to Background Color, select a color in the Windows color selector that appears, and then click OK.

– To select the font used, click Font, select a font in the Windows font selector that appears, and then click OK. If you want the header of each table column to use the font you selected, select "Use same font for header."

3. In the Configure Tables window, do the following for each table you want to display:

a. Click Table.

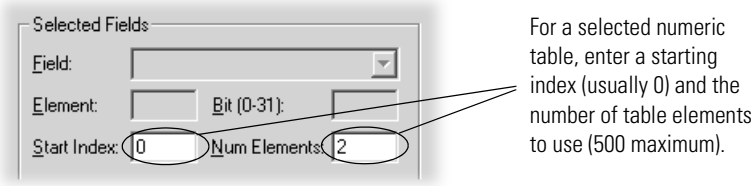
The Configure Tables dialog box appears.



b. Enter a description for the table.

c. Click the Tag Selection button and in the Tag Selection dialog box that appears, select a table to display. To learn more about configuring tags in your project, see "Configuring Tags" on page 5-11.

d. To display a range of elements in the table you select, enter a start index and the number of elements to be displayed.



e. Click OK to exit the Tag Selection dialog box.

f. Click OK to exit the Configure Tables dialog box.

g. If you want to display another table in the numeric table object, click another Table button, otherwise click OK to complete configuration.

Printing Graphics

To print the displayed windows, select File→Print to display the Print dialog box. If the settings are correct for your printer, click OK.

If you need to change printer settings, click the Properties button in the Print dialog box. You can also change the printer settings without printing the Configurator screens by selecting the File→Printer Setup command.

CHAPTER 7

Using Animated Graphics

Introduction

This chapter describes how to animate graphics to show how I/O data and other values change in real time. It also describes how ioDisplay scans data from control engines to update its graphics, and how you can adjust this scanning to optimize your ioDisplay project for best performance.

In This Chapter

About Animated Graphics

As your ioControl strategy runs on the control engine, values and states of tags in the ioControl strategy database are continuously updated. ioDisplay uses this changing data to modify attributes (such as size, position, and color) of the graphics that you have connected to the tags. The end result is an animated, continually updated display that shows status information about a control process.

To animate graphics, you must assign *dynamic attributes* to objects you've drawn. These are attributes that make the graphic object change based on the values read from or sent to control engines, based on events that happened, or based on how the operator interacts with the interface.

Two types of dynamic attributes can be assigned to a graphic object: control engine-driven attributes and operator-driven attributes.

Control engine-driven attributes are always assigned to a particular tag (I/O point, value, etc.) in an ioControl strategy running on an attached control engine. This type of attribute changes a graphic as tags are read from a control engine. For example, if a tag reflects a

lower-level alarm condition for a process, the attached graphic can change color to red to alert the operator.

Operator-driven attributes are assigned to a graphic object. These attributes change the graphic as an operator interacts with the interface. As a result, events may be triggered, or new tag values may be sent to an attached control engine. For example, if an operator clicks a graphic that looks like a button, a valve is closed.

Adding Dynamic Attributes to Graphics

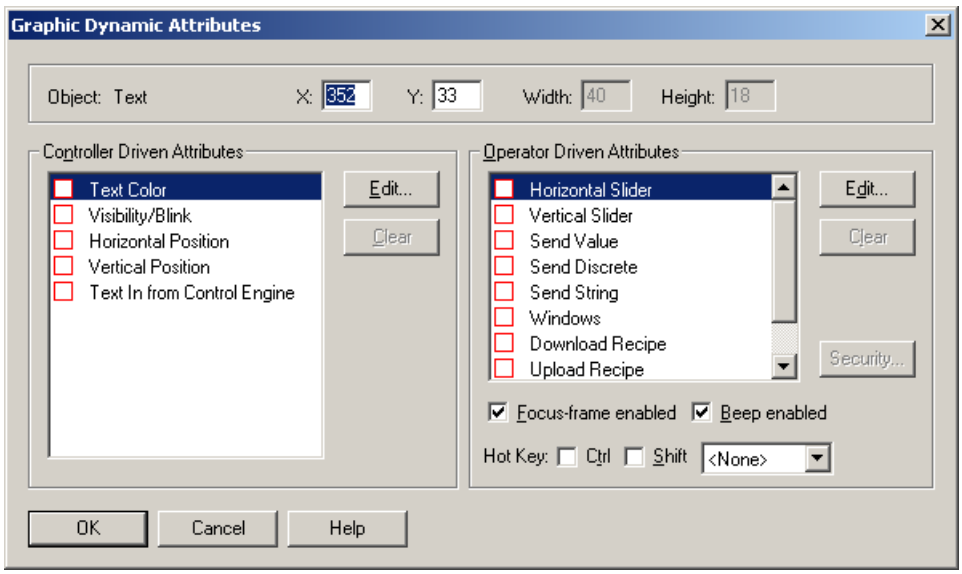
As you finish drawing your operator interface, you can start assigning dynamic attributes to some of the on-screen graphics. We will begin by explaining the general process you'll use to assign dynamic attributes to an object, and then give detailed explanations of each attribute that can be assigned.

Assigning a Dynamic Attribute

1. Choose the Select tool from the toolbox and double-click the graphic to which you want to assign a dynamic attribute. (You can also click the graphic once and Edit→Edit Dynamic Attributes.)

   NOTE: Dynamic attributes can only be assigned to one object at a time. If you want several objects to have the same attributes, select the objects, and then choose Edit→Group to make them one object. (You can also right-click on the selected graphics and choose Group from the pop-up menu.) Remember that if you ungroup the objects, the attributes you configured when they were a group aren't retained.

   The Graphic Dynamic Attributes dialog box appears:



At the top of the dialog box is a brief description of the item selected, including its location (x and y coordinates) and dimensions (width and height) in the draw window. You can change the location and dimensions of the graphic by entering new values in place of the current ones. (You can't change the dimensions of a text graphic.)

2. Choose an attribute you want to configure in either the Control Engine-Driven Attributes or Operator-Driven Attributes list, and then click the Edit button for that list.

   The dialog box that appears will differ depending on the attribute you selected. The options and features of each attribute are covered in detail in "Available Dynamic Attributes," which starts on page 7-6.

3. Configure the attribute as required, and then click OK to return to the Graphic Dynamic Attributes dialog box.

4. If you made any changes to operator-driven attributes, complete these options:

   - **Focus-frame enabled**—If this option is checked, a light border will appear around the graphic in Runtime when the operator moves the cursor over it. This border can be used as a visual aid to let the operator know that an event will occur when the graphic is clicked. (You must still configure additional dynamic attributes for the graphic so these events can occur.)
   - **Beep enabled**—If this option is checked, the operator will hear a beep when the graphic is clicked. Use this as an audio confirmation.
   - **Hot Key**—This feature associates a keystroke sequence with a graphic that has a dynamic attribute assigned and configured. A *hot key* is a key on the keyboard that, when pressed (sometimes in combination with an optional CTRL or SHIFT key), will activate the associated graphic's operator-driver attributes. This lets the operator use a keystroke sequence instead of using the mouse to click on the graphic, allowing ioDisplay Runtime to be operated without a mouse or similar device.

     NOTE: The hot key will work only for graphics that are in an opened or minimized window; closed windows will not be affected.

5. To configure operator security for this graphic object, click Security. See "Security Settings for Graphics and Dynamic Attributes" on page 7-4 for configuration information.

6. To clear an attribute you've configured, simply highlight the attribute and click Clear.

   You should see an empty checkbox beside the attribute. Sometimes after you configure an attribute and return to the Graphic Dynamic Attributes dialog box, the Not Available indicator ⊘ appears in the checkbox for another attribute. This symbol means that you cannot configure that type of dynamic attribute for this object as a result of the attribute you just configured.

7. When you're done configuring dynamic attributes for this object, click OK to save your settings and close the dialog box.

Assigning Multiple Dynamic Attributes to a Graphic

If you assign more than one dynamic attribute to a graphic object, the attributes will execute in the order in which they were assigned when you run the project in ioDisplay Runtime. For example, if you assign a button the Send Discrete dynamic attribute, and then assign the same button the Send Value dynamic attribute, that is the order in which the attributes will execute when the operator clicks the button.

Security Settings for Graphics and Dynamic Attributes

Your application requirements and the environment in which it is used may require strict control over accessing and using the operator interface you create in ioDisplay. You can configure your ioDisplay project to provide this level of security by defining user authentication permissions for individual onscreen graphic objects. This authentication is based on the users and groups defined in a Microsoft Windows network.

When the ioDisplay project is run in ioDisplay Runtime, an operator who clicks on an object with security permissions is prompted to enter a Windows network username and password. If the login information is incorrect, or if the operator is not permitted to use that object, an alert message is displayed. If runtime operator logging is active for the project, the login attempt—successful or not—will be added to the runtime operator log.

Important Considerations for User- and Group-Level Security Settings

There are several important considerations to keep in mind when configuring user- and group-level authentication for a graphic object.

- By default, *all* operators have permission to use an object. It isn't necessary to configure security if all operators will have permission to change the tag value for the object.
- Security permissions cannot be configured for the Send Discrete dynamic attribute when it is configured as "Direct" or "Reverse".
- When the project is running in ioDisplay Runtime and an object is clicked, "Deny Access" security permissions have priority over "Grant Access" permissions. This means that if a user has been granted access, but is a member of a group that has been denied access, the user will not be able to use the onscreen object.

Configuring Security Permissions for a Graphic Object

To assign user- and group-based security permissions to a graphic object, do the following:

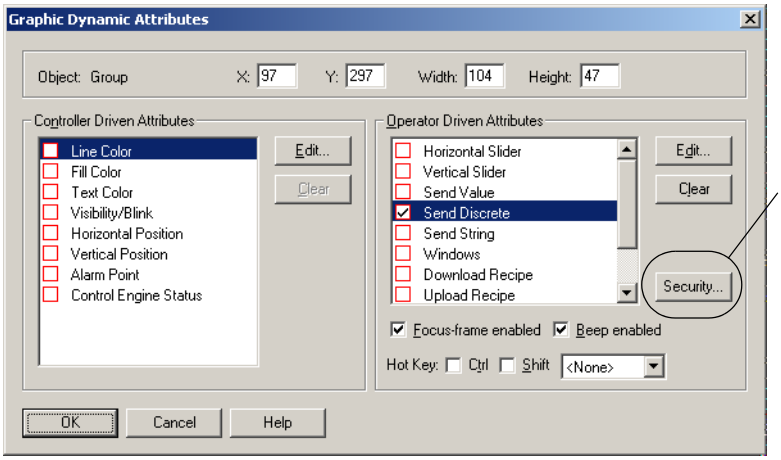
1. Double-click the graphic object to which you will assign one or more operator-driven dynamic attributes.

   The Graphic Dynamic Attributes dialog box opens. Note that the Security button on the right side is not active.

2. Double-click an operator-driven attribute that you want to use, configure it as needed, and then click OK.

*NOTE: Security permissions are not applied to individual dynamic attributes, but are applied to all dynamic attributes selected **before** you click the Security button. If a graphic object needs to have multiple dynamic attributes, make sure to select and configure all attributes before configuring security settings.*

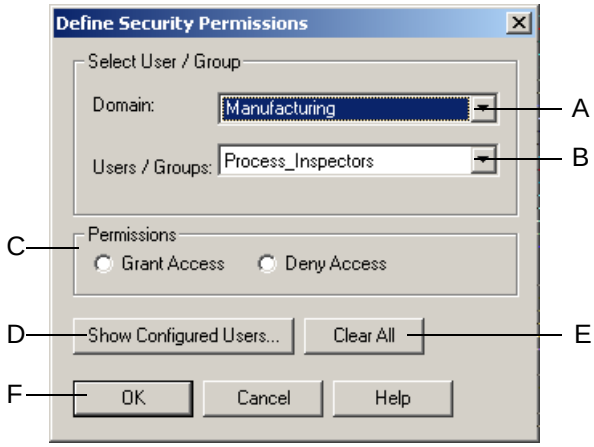
In the Graphic Dynamic Attributes dialog box that is again visible, the Security button is now active.



After configuring one or more operator-driven dynamic attributes, click Security to configure user- and group-level authentication for the graphic object.

3. Click Security.

The Define Security Permissions dialog box opens.



A Select the Windows network domain that contains the users and/or groups to whom you want to grant or deny access to the graphic object.

B Select the Windows user or group to whom you want to grant or deny access to the object.

C For the selected user or group, select Grant Access or Deny Access.

D Click Show Configured Users to view all permissions currently assigned for the graphic object.

E Click Clear All to erase all configured permissions for the graphic object.

F Click OK to save changes, or keep Cancel to close the dialog box without any changes being made.

Available Dynamic Attributes

You can add the dynamic attributes listed below to a graphic. Note that not all attribute types are available for all types of graphics; only attributes that can be used with a particular graphic type appear in the Dynamic Attributes dialog box.

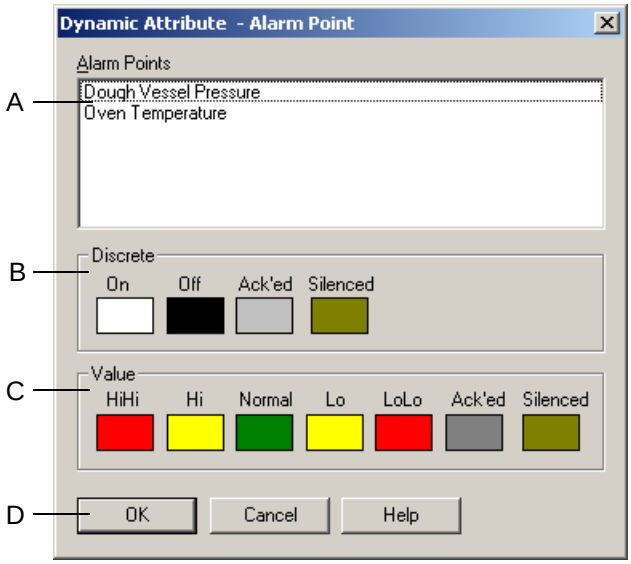
Dynamic Attribute	Type	See Page
Alarm Point	control engine-driven	7-7
Control Engine Status	control engine-driven	7-8
Download Recipe	operator-driven	7-9
Execute Menu Item	operator-driven	7-9
Fill Color	control engine-driven	7-10
Horizontal Position	control engine-driven	7-12
Horizontal Size (Width)	control engine-driven	7-13
Horizontal Slider	operator-driven	7-14
Launch Application	operator-driven	7-15
Line Color	control engine-driven	7-16
Read and Clear	operator-driven	7-17
Rotate	control engine-driven	7-18
Send Discrete	operator-driven	7-19
Send String	operator-driven	7-20
Send Value	operator-driven	7-21
Text Color	control engine-driven	7-22
Text In (from control engine)	control engine-driven	7-24
Upload Recipe	operator-driven	7-25

Dynamic Attribute	Type	See Page
Vertical Position	control engine-driven	7-26
Vertical Size (Height)	control engine-driven	7-27
Vertical Slider	operator-driven	7-28
Visibility/Blink	control engine-driven	7-29
Windows	operator-driven	7-31

Alarm Point

Use this attribute to change the color of a graphic object based on the state of an alarm point. You can use this attribute with circles, rectangles, and polygons. By default, the color of the graphic corresponds to the Normal, or unalarmed, state. For more information about configuring alarm points and using alarms in a project, see "Alarming" on page 9-28.

Double-click Alarm Point in the Graphic Dynamic Attributes dialog box to display the following:



A The names of the alarm points configured for the project appear here. Click an alarm point in the list, and then set the colors that will be used for the graphic object in the Discrete group (B) or the Value group (C).

Alarm points that monitor digital points are configured as *discrete* alarm points, and have only two states: On and Off. Alarm points that monitor analog points are configured as *value* alarm points have the states HiHi, Hi, Normal, Lo LoLo, Ack'ed, and Silenced. See "Configuring Alarm Points" on page 9-28 for more information about setting up alarm points.

B To select colors for a discrete alarm point's On and Off states, click the color box for each state and then choose a color in the dialog box that appears. If you have selected an

alarm point for a digital I/O point, only the Discrete group will appear in the Dynamic Attribute - Alarm Point dialog box.

C To select colors for a value alarm point's HiHi, Hi, Normal, Lo LoLo, Ack'ed, and Silenced states, click the color box for each state and then choose a color in the dialog box that appears. If you have selected an alarm point for an analog I/O point, only the Value group will appear in the Dynamic Attribute - Alarm Point dialog box.

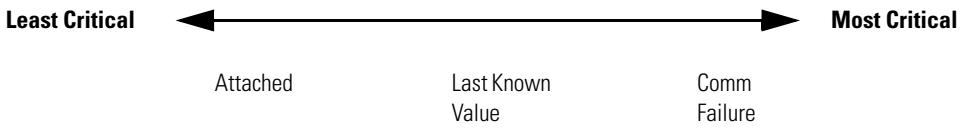
D Click OK to save your settings.

Control Engine Status

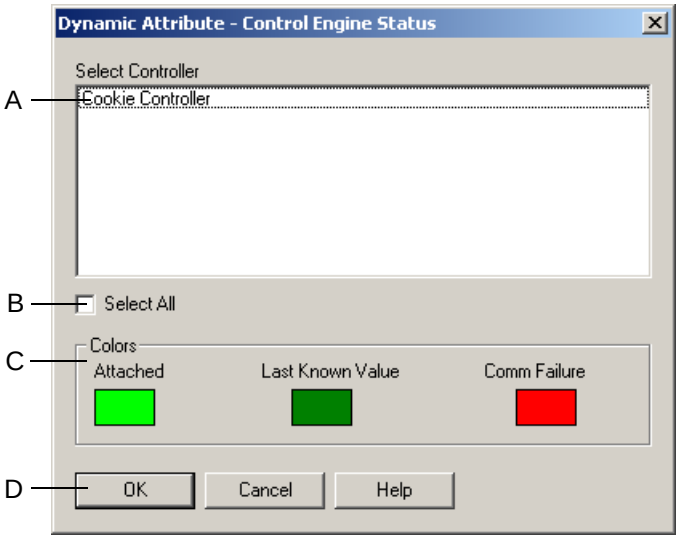
Use this attribute to change the color of a graphic object based on the status of one or more control engines. You can use this attribute with circles, rectangles, and polygons.

NOTE: When the Control Engine Status dynamic attribute is selected, all other input dynamic attributes are disabled.

If the Control Engine Status attribute is used to monitor multiple control engines, the graphic object will display the status color for the control engine that is in the most critical condition. Least critical to most critical status is shown below:



Double-click Control Engine Status in the Graphic Dynamic Attributes dialog box to display the following:



A The names of the control engines configured for the project appear here.

Select a control engine. To select multiple control engines, hold down the CTRL key and click each one you want to select.

B To select all control engines in the list, click Select All.

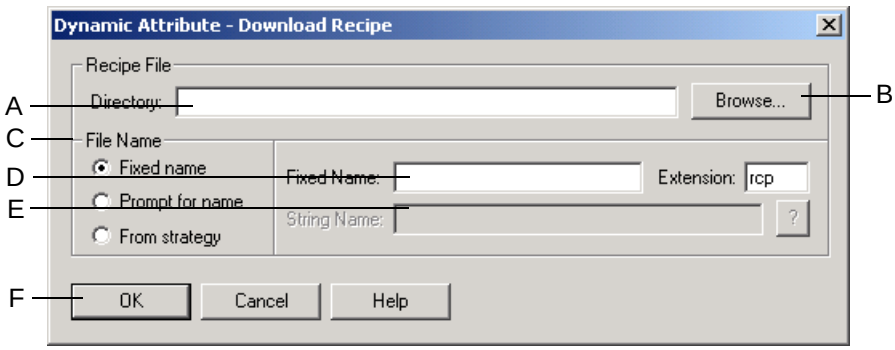
C To select colors for a control engine's state, click the color box for the state and then choose a color in the dialog box that appears.

D Click OK to save your settings.

Download Recipe

Use this attribute to download a recipe file to a control engine when a graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text. For more information about recipes, see "Recipes" on page 9-19.

Double-click Download Recipe in the Graphics Dynamic Attributes dialog box to display the following:



A Enter the directory location of the recipe file. Use the Browse button B to quickly enter a directory name.

B Click to quickly find the recipe file directory for A. Use the Select Download File Directory dialog box that appears to navigate to the desired directory, and then click OK.

C Choose the source of the recipe file name:

- If you choose Fixed Name, D is highlighted.
- If you choose Prompt for name, the operator will be prompted for the recipe file name.
- If you choose From strategy, E is highlighted.

D If Fixed Name was selected in C, enter the name of the recipe file located in Directory (A). Notice the file extension is .rcp.

E If From strategy was selected in D, use the Tag Selection button ? to enter a tagname of type string that contains the recipe file name. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

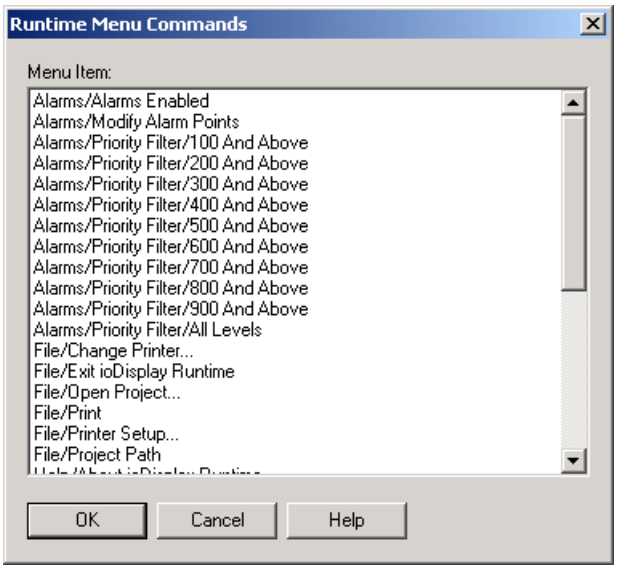
F Click OK to save your settings.

Execute Menu Item

Use this attribute to run a single Runtime command when an object is clicked. This is useful if you want to hide the menu bar and only allow limited access to certain items. This is an

operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Execute Menu Item in the Graphics Dynamic Attributes dialog box to display the following:

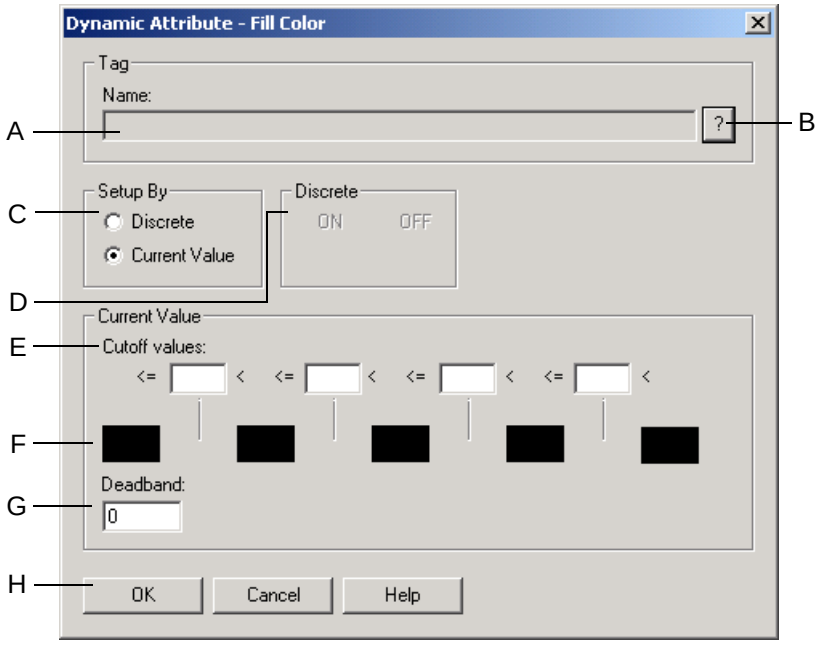


Select a command from the Menu Item list and click OK. (Click Cancel to close the dialog box without making any changes.)

Fill Color

Use this attribute to change a graphic's fill color based on a tag value from the control engine. This is a control engine-driven attribute and is available for rectangles, round rectangles, ellipses, and polygons.

Double-click Fill Color in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As the tag value changes in the ioControl strategy, the graphic's fill color will change.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Choose whether the tag you're configuring is a discrete value (with a value of ON or OFF), or whether it's a current value you're looking for. If you choose Discrete, the D group is highlighted; if you choose Current value, the E group is highlighted. The choice you make in this selection must match the tag choice you can made in A.

D Select the color you want for the ON state by clicking on the color field below ON. The Color dialog box appears; choose a color and then click OK. Repeat this step for the OFF state.

E Enter a value in each Cutoff Value field to specify the range of values for each color group configured in the color fields. Values entered must be in increasing order. Each Cutoff value field must have a numeric value entered or you will get a warning to enter one. After the warning, the cursor blinks in the first Cutoff value field requiring a value.

F To configure a color for the range set up in E, click on a color field. The Color dialog box appears; choose a color and then click OK. Repeat this step for each color field you want to change.

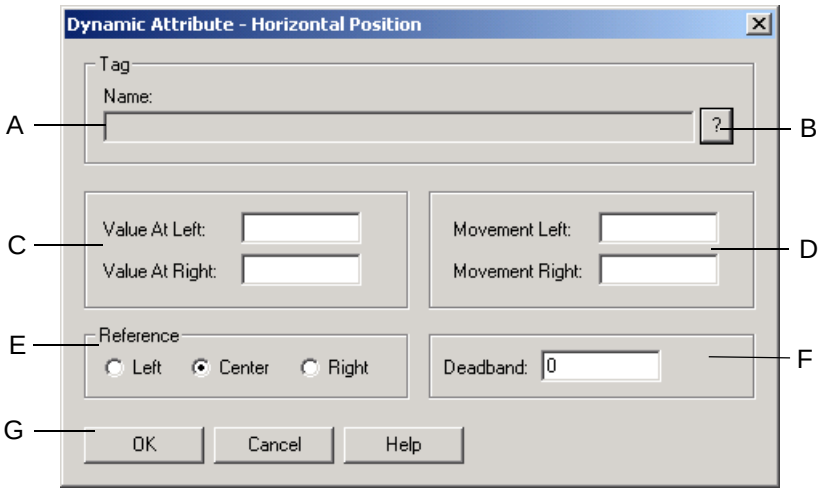
G Enter a value to be added and subtracted from each Cutoff value to determine the actual value at which the color will change to the next color field. For example, let's say the Cutoff values are 1, 10, 20, and 30, and the color fields are red, yellow, green, blue, and black. The Deadband is 3. A tag with a value of 9 is read. The next tag value read is 11. The graphic color remains yellow because the value read is within the deadband range, even though a value of 11 is in the green range.

H Click OK to save your settings.

Horizontal Position

Use this attribute to adjust the horizontal position of a graphic based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines, rectangles, round rectangles, ellipses, and polygons, polylines, curves, bitmaps, and text.

Double-click Horizontal Position in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As the tag value changes, the graphic's horizontal position is changed.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about configuring this dialog box.

C Enter the leftmost and rightmost value for the tagname in A. For example, if you know your tag values will be from 0 to 100, you may wish to enter a range of 0 to 100 or a subset of this range, such as 0 to 50.

D Enter the leftmost and rightmost movement the object can change. The movement units are in pixels. Suppose your left and right values are 0 and 100, and your left and right movements are 0 and 200. When the tag value is 50, the object will be moved 100 pixels to the right.

E Select the reference point for the object. The choices are left, center, and right. The Horizontal Size (Width) dynamic attribute must also be configured in order for this option to affect the graphic.

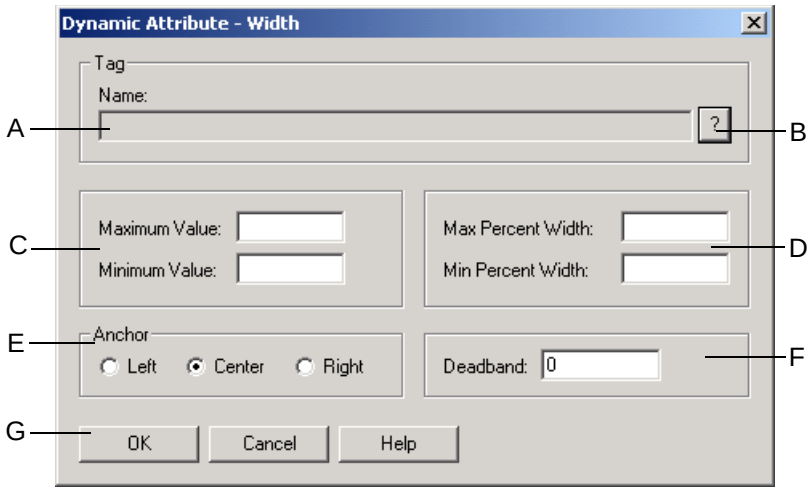
F Enter a value to be added and subtracted from the previously read tag value to determine if the graphic's movement will actually change. Using our previous example in D, let's say the deadband is 5. A tag is read and has a value of 50. The next tag reading must be greater than 55 in order for the graphic to move.

G Click OK to save your settings.

Horizontal Size (Width)

Use this attribute to adjust the width of a graphic based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines, rectangles, round rectangles, ellipses, polygons, polylines, and curves.

Double-click Horizontal Size (Width) in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As the tag value changes in the ioControl strategy, the graphic's width is changed.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Enter the maximum and minimum value for the tagname in A. For example, if you know your tag values will be from 0 to 100, you may wish to enter a range of 0 to 100 or a subset of this range, such as 0 to 50.

D Enter the maximum and minimum percentage the object can change. The percentage range is from 0 to 1,000 percent. Suppose your minimum value is 0 and your maximum value is 10, and the minimum and maximum percentages range is 0 to 200. When the tag value is 10, the object will be twice as big as you've originally drawn it. When the tag value is 5, the object will be the same size you've drawn it, which is 100 percent.

E Enter the anchor point for the object. This is the reference point on the object from which the graphic changes. The choices are left, center, and right.

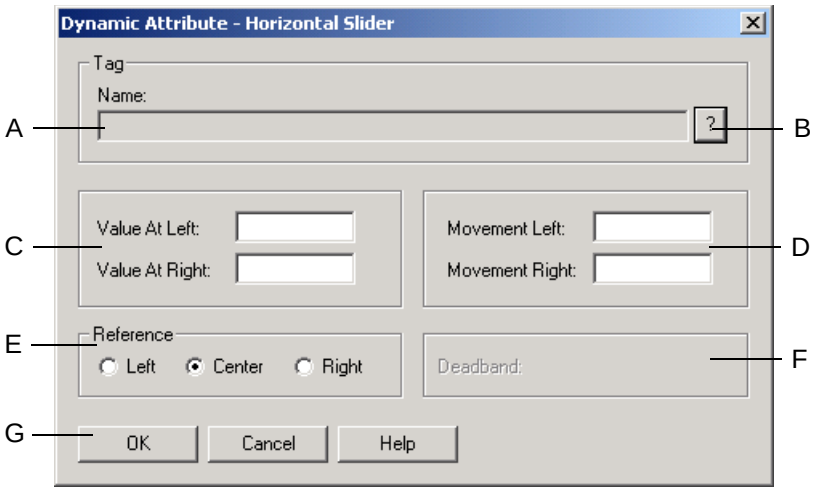
F Enter a value to be added and subtracted from the previously read tag value to determine if the graphic's width will actually change. For example, let's say the values are 0 to 100, the percentages are 0 to 100, and the deadband is 5. A tag is read and has a value of 10. The next tag reading must be greater than 15 in order for the graphic to change.

G Click OK to save your settings.

Horizontal Slider

Use this attribute to configure a horizontal slider when a graphic is clicked. This is an operator-driven attribute and is available for lines, rectangles, round rectangles, ellipses, polygons, and bitmaps.

Double-click Horizontal Slider in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. The distance the horizontal slider is moved affects the value sent to the tag in the control engine.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Enter the leftmost and rightmost value for the tagname in A. For example, if you know your tag values can be from 0 to 100, you may wish to enter a range of 0 to 100 or a subset of this range, such as 0 to 50.

D Enter the leftmost and rightmost movement the object can change. The movement units are in pixels. Suppose your left and right values are 0 and 100, and your left and right movements are 0 and 200. If you move the tag 100 pixels to the right, the tag value sent will be 50.

E Select the reference point for the object. The choices are left, center, and right. Note that you must also separately configure the Horizontal Size (Width) dynamic attribute to use this option.

F Enter a value to be added and subtracted from the previously read tag value to determine whether the tag value should be changed.

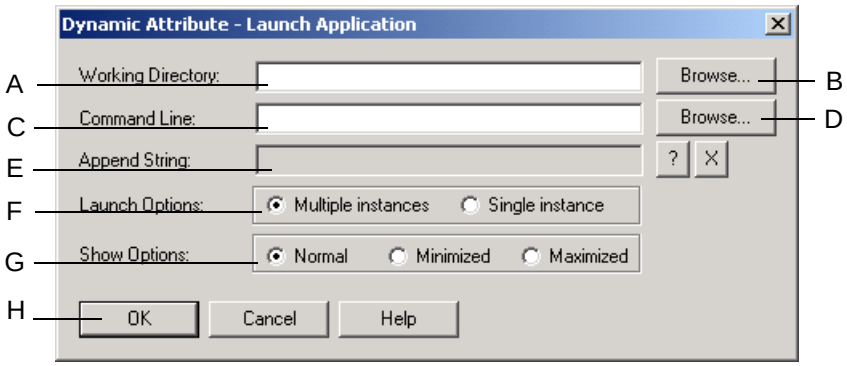
Using our previous example in D, let's say the deadband is 5. A tag is read and has a value of 50. The next graphic movement must be greater than 55 in order for the tag value to change.

G Click OK to save your settings.

Launch Application

Use this attribute to start an application when a graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Launch Application in the Graphics Dynamic Attributes dialog box to display the following:



A Enter the working directory you want to be in when you start the application. Use the Browse button B to quickly enter a directory name.

B Click to find the directory for A. The Working Directory Selection dialog box appears. Use it to navigate to the desired directory and click OK when you're done.

C Enter the complete path and file name of the application you want to launch. Use D to quickly enter the path.

D Click to find the path and application name for C. The Application Manager Executable File Selection dialog box appears. Use it to select an application and click Open when you're done.

E (Optional) Enter the name of a string tag to use to append to the path entered in C. The string appended may be a file name the launched application should open. Use the Tag Selection button ? to choose the tagname. The Tag Selection dialog box is displayed so you can select a tag. Use X to clear an entry you may have made in this field.

F Select Single Instance to have ioDisplay Runtime check whether the graphic has previously launched an application that is currently running. If the graphic has not previously started a currently running application, the application will be launched. Select Multiple instances to allow the graphic to start more than one instance of an application.

Note that the Single Instance option doesn't limit the number of active sessions of an application that is launched by other graphics and triggers. For example, if the graphic launches a Microsoft Word session, that graphic can't launch any other application until the Word session ends. However, a trigger-based event can launch a second, separate session of Word, so two Microsoft Word sessions will be running concurrently.

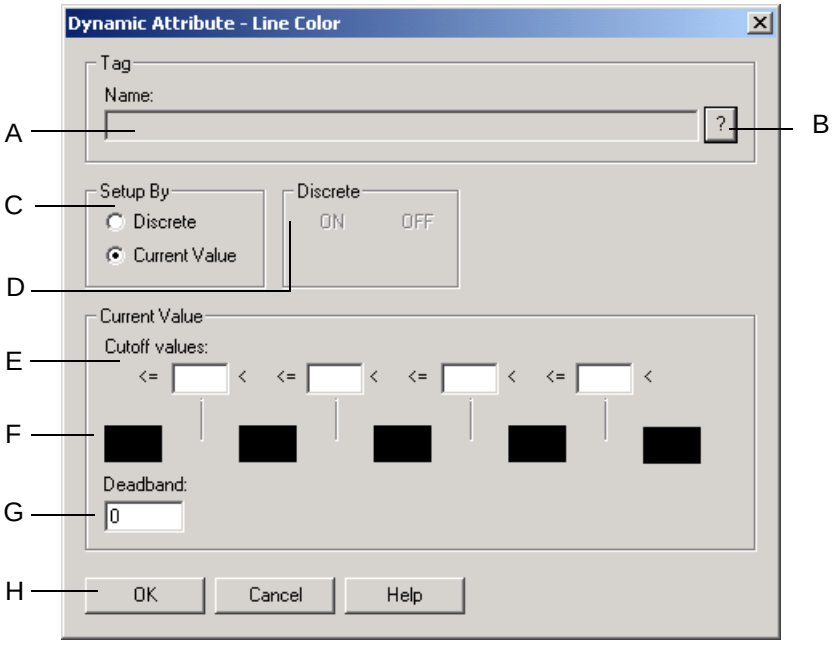
G Click here to configure how the application window will appear. Your choices are Normal, Minimized, and Maximized.

H Click OK to save your settings.

Line Color

Use this attribute to change a line color or the line color around a graphic based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines, rectangles, round rectangles, ellipses, polygons, polylines, and Bezier curves.

Double-click Line Color in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking B. As this tag value changes in the ioControl strategy, the graphic's line color changes.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Choose whether the tag you're configuring is a discrete value (with a value of ON or OFF), or whether it's a current value you're looking for. If you choose Discrete, the D group is highlighted; if you choose Current value, the E group is highlighted.

D Select the color you want for the ON state by clicking on the color field below ON. The Color dialog box appears; choose a color and click OK. Repeat this step for the OFF state.

E Enter a value in each Cutoff Value box to specify the range of values for each color group configured in the color fields. Values entered must be in increasing order. Each Cutoff value box must have a numeric value entered or you will get a warning to enter one. After the warning, the cursor blinks in the first Cutoff Value box requiring a value.

F To configure a color for the range set up in E, click on a color field. The Color dialog box appears from which you can choose a color and then click OK to accept. Repeat this step for each color field you want to change.

G Enter a value to be added and subtracted from each cutoff value to determine the actual value at which the color will change to the next color field.

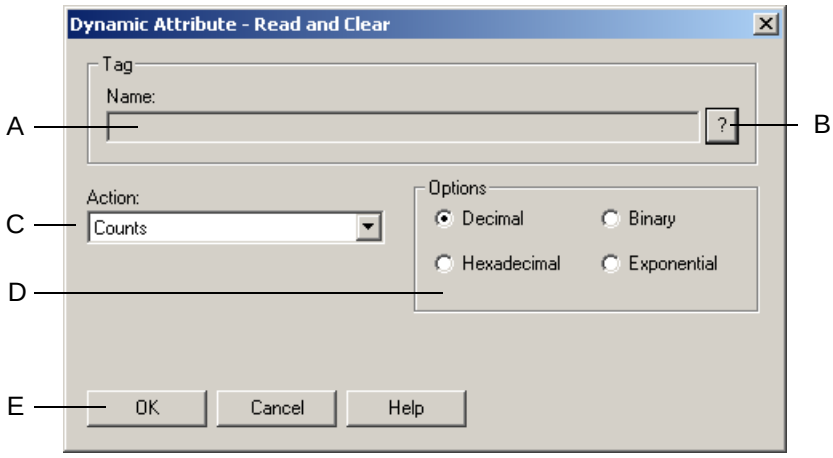
For example, let's say the cutoff values are 1, 10, 20, and 30, and the color fields are red, yellow, green, blue, and black. The deadband is 3. A tag with a value of 9 is read. The next tag value read is 11. The graphic's color remains yellow because the value read is within the deadband range, even though a value of 11 is in the green range.

H Click OK to save your settings.

Read and Clear

Use this attribute to read a tag value and then clear (reset) it. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Read and Clear in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking B.

B To enter a tagname in A, click here.

The Tag Selection dialog box is displayed so you can select a tag. Only tags that are supported by the item selected in C will be available. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Choose the item to be read and cleared when the operator clicks the graphic object:

- Counts (returns an integer value and then clears counts)
- On time total
- Off time total
- Latch (ON)
- Latch (OFF)
- On pulse measure
- Off pulse measure
- Period

D Formatting options for how values appear when read appear here. Depending on the item you selected in C, one of the following option groups will be available:

- Numerical formats—Decimal, binary, hexadecimal, and exponential formats are available for values returned as numbers (for example, the state of all 32 points on a high-density module).
- Text formats—Custom text that corresponds to a discrete value's on and off states can be entered here.

E Click OK to save your settings.

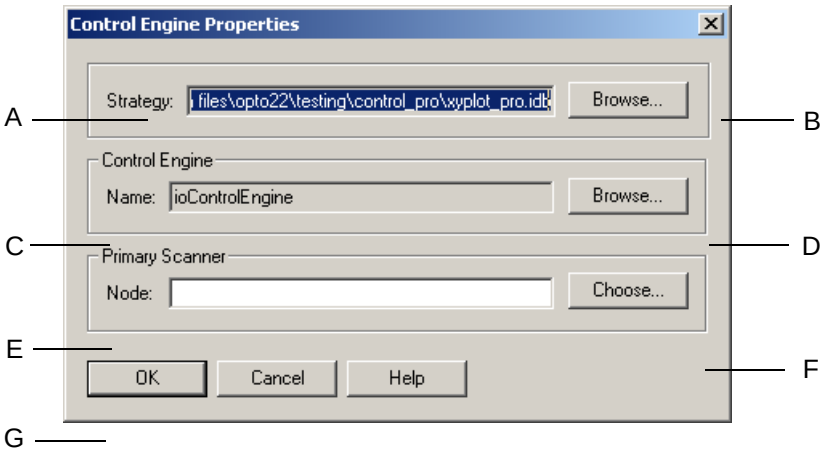
When using this dynamic attribute, please note the following:

- If the Read and Clear dynamic attribute has been assigned to an object, a "Text In from Control Engine" dynamic attribute cannot also be assigned to that same object.
- If a text placeholder (##) is used to view return values, these placeholder characters will appear that way in Runtime until the operator triggers the read and clear. Once this has happened, the graphic will not change until the read and clear is triggered again.
- If more than one graphic in an ioDisplay window is configured with the same Read and Clear dynamic attribute, each graphic is treated independently. The graphics are not linked to each other, so when one graphic is read and cleared, the other graphic will not reflect this.

Rotate

Use this attribute to rotate a line based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines only.

Double-click Rotate in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As this tag value changes in the ioControl strategy, the graphic's line rotates.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C These fields are used to specify the return data from the tag that will produce the maximum counterclockwise (CCW) and maximum clockwise (CW) rotation. The Value at Max CCW may be either less than or greater than the Value at Max CW, so that rotation may proceed in either direction as the data from the tag increases or decreases. Therefore, the term "exceeds" is used below to indicate a value that may be either greater than or less than the CCW or CW Max value.

When the tag returns data that is equal to or exceeds the Value at Max CCW to ioDisplay, the graphic will rotate counterclockwise as far as possible, as specified by the Max CCW Rotation value explained below. When the tag returns data that is equal to or exceeds the Value at Max CW Rotation to ioDisplay, the graphic will rotate clockwise as far as possible, as specified by the Max CW Rotation value explained below.

D These fields are used to specify the maximum counterclockwise and clockwise rotation angle that the graphic may undergo (in degrees from its configured location). The entered values must be non-negative numbers. The Max CCW Rotation angle is achieved when the tag returns data that is equal to or exceeds the Value at Max CCW, as explained above. The Max CW Rotation angle is achieved when the tag returns data that is equal to or exceeds the Value at Max CW, as explained earlier in this section.

E The rotation anchor point is used to specify the fixed location that the graphic rotates around. This location is specified in terms of an offset (in units of pixels) from the centerpoint of the graphic at its configured location.

In the Horizontal field, enter a negative value to specify a position that is to the left of the configured location or enter a positive value to specify a position that is to the right of the configured location. In the Vertical field, enter a negative value to specify a position that is above the configured location or enter a positive value to specify a position that is below the configured location. If a value of zero is specified for both fields, then the graphic will rotate about its centerpoint.

F Enter a value to be added and subtracted from the previously read tag value to determine if the line's rotation angle should change.

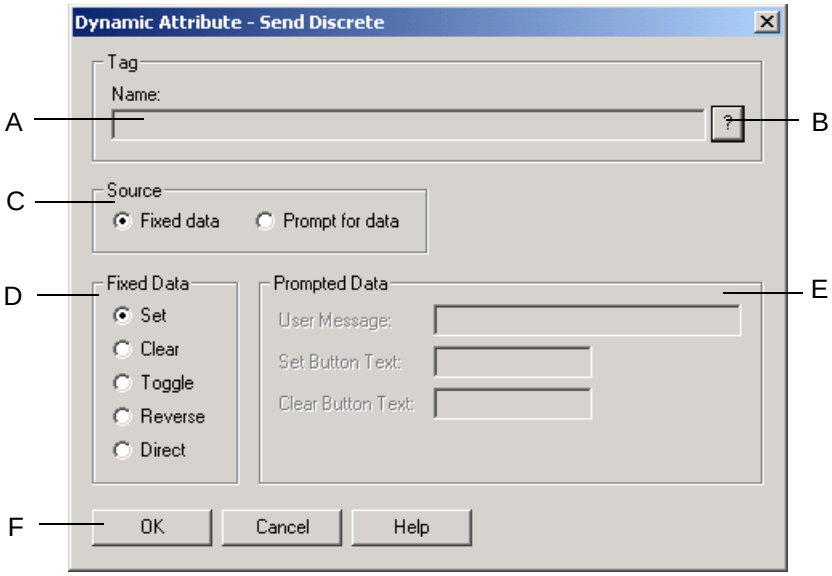
For example, let's say the deadband is 2. A tag with a value of 50 is read. The next tag value read is 51. The line does not rotate because the value read is within the deadband range.

G Click OK to save your settings.

Send Discrete

Use this attribute to send a discrete value to a tag in the control engine when a graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Send Discrete in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. This tag will receive the tag value entered by the operator or configured in this dialog box.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Select the source of the value to be sent to the tag. If you select Fixed data, the setting selected in field D is sent to the tag. If you select Prompt for data, the field in E is activated and you can enter a message that will prompt the operator to select a discrete state for the tag.

D If Fixed Data was selected in C, enter the setting to be sent to the tag. The Set option turns the tag on; Clear turns the tag off; Toggle changes the tag to the opposite of its current state (from on to off, or from off to on); Reverse changes the tag's state to off when the mouse is clicked on the graphic, and on when the mouse button is released; Direct changes the tag state to on when the mouse is clicked on the graphic, and off when the mouse is released.

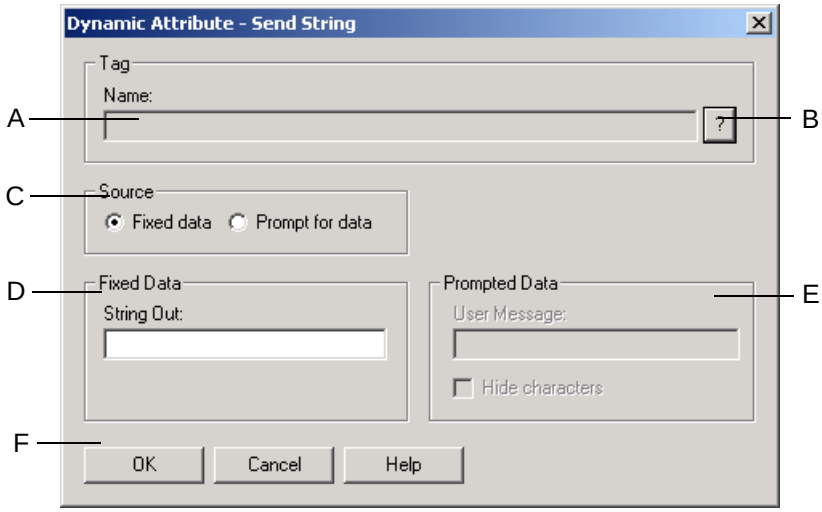
E If Prompt for Data was selected in C, enter a message here that will prompt the operator to select a tag state. Two buttons will be displayed to the operator. Enter the labels for the Set Button and the Clear Button. The Set Button sends a discrete value to set the tag to the on state, and the Clear Button sends a discrete value to set the tag to the off state.

F Click OK to save your settings.

Send String

Use this attribute to send a string to the control engine when a graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Send String in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. This tag will receive the string tag entered by the operator or configured in this dialog box.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Select the source of the value to be sent to the tag. If you select Fixed data, the setting selected in field D is sent to the tag. If you select Prompt for data, the field in E is activated and you can enter a message that will prompt the operator to enter a string to send to the tag.

D If Fixed Data was selected in C, enter the string to send to the tag.

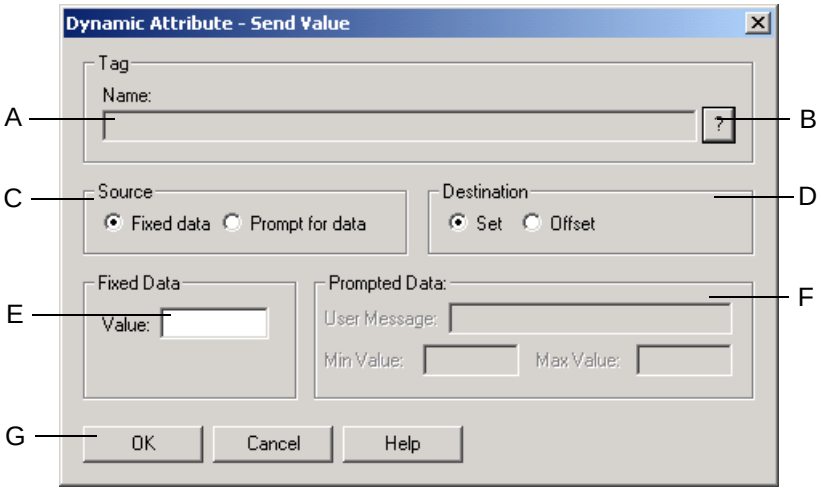
E If Prompt for Data was selected in C, enter a message to prompt the operator to enter a string. Choose the Hide characters option if you don't want the text entered by the operator to be displayed.

F Click OK to save your settings.

Send Value

Use this attribute to send a value to a tagname in the control engine when a graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Send Value in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. This tag will receive the tag value entered by the operator or configured in this dialog box.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Select the source of the value to be sent to the tag. If you select Fixed data, the value entered in field E is sent to the tag. If you select Prompt for data, the field in F is activated and you can enter a message that will prompt the operator to input a value to send to the tag.

D Choose Set to assign the sent value to the tag equal, or choose Offset to add the value to the tag.

E If Fixed Data was selected in C, enter the value to send to the tag.

This value can be either a decimal or hexadecimal number. If entering a hexadecimal number, the number must be preceded by "0x" and contain no spaces (for example, "0x7FFFFFFF" and not "0x 7FFF FFFF").

F If Prompt for Data was selected in C, enter a message to prompt the operator for a value. Enter a minimum and maximum value the entered data should be within.

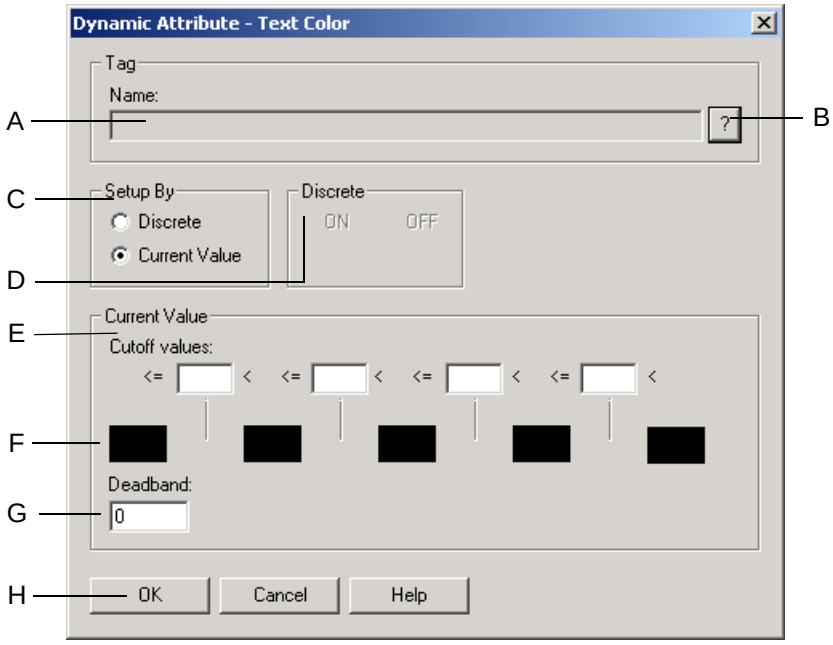
IMPORTANT: *If using hexadecimal numbers, enter a minimum number of 0x80000000 and a maximum number of 0x7FFFFFFF.*

G Click OK to save your settings.

Text Color

Use this attribute to change the color of text in the interface based on a tag value from a control engine. This is a control engine-driven attribute and is available for grouped objects and text.

Double-click Text Color in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As this tag changes value in the ioControl strategy, its value will determine the color of the text.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Choose whether the tag you're configuring is a discrete value (with ON and OFF values), or whether it's a current value you're looking for. If you choose Discrete, the D group is highlighted; if you choose Current value, the E group is highlighted. The choice you make in this selection must match the tag choices you made in A.

D If you selected Discrete in C, this field is activated and you can select the color you want for the ON state by clicking on the color field below ON. The Color dialog box appears; choose a color and then click OK to accept. Repeat this step for the OFF state.

E Enter a value in each Cutoff value field to specify the range of values for each color group configured in the color fields. Values entered must be in increasing order. Each Cutoff value field must have a numeric value entered or you will get a warning to enter one. After the warning, the cursor blinks in the first Cutoff value field requiring a value.

F To configure a color for the range set up in E, click on a color field. The Color dialog box appears; choose a color and click OK. Repeat this step for each color field you want to change.

G Enter a value to be added and subtracted from each Cutoff value to determine whether any changes in color should occur.

For example, let's say the Cutoff values are 1, 10, 20, and 30, and the color fields are red, yellow, green, blue, and black. The Deadband is 3. A tag with a value of 9 is read. The next tag value read is 11. The text color remains yellow because the value read is within the deadband range, even though a value of 11 is in the green range.

H Click OK to save your settings.

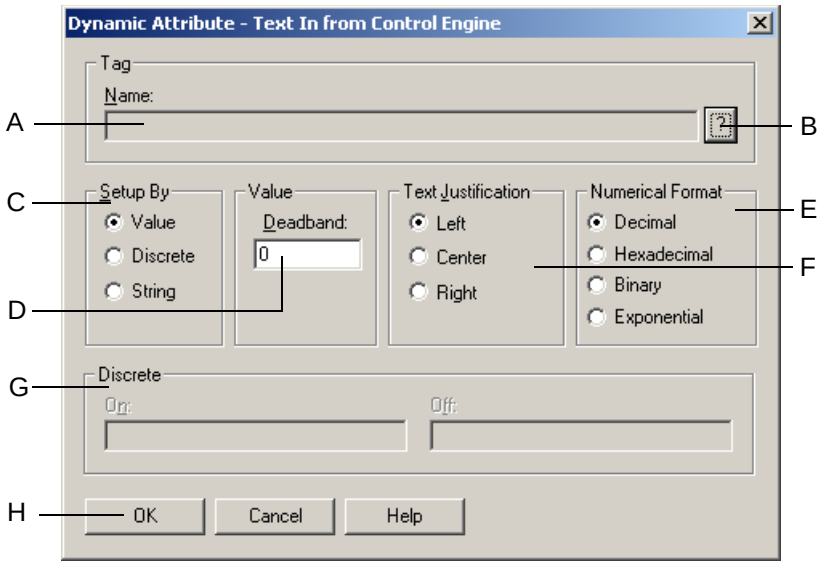
Text In (from control engine)

Use this attribute to read a tag from a control engine and display various strings to the operator. You can read in a numeric value, a string, or a discrete value, and set up a string that will be displayed based on what was received. This is a control engine-driven attribute and is available for a text object.

A # sign in the text object indicates where the text string should be displayed. Only one # is needed to display an entire string. If there's no # sign within the text object, then the string will be appended to the end of the text object. You can configure how a floating point number is displayed by using a decimal point along with the # signs. For every decimal place you want displayed after a decimal point, use a # sign. For example, your text object could say: "Low level reading: ###.#." Your tag value is 200.55, so your displayed string is: "Low level reading: 200.6."

NOTE: The extra # signs to the left of the decimal point aren't required, but are useful for determining how much space the text will require in your display.

Double-click Text In (from control engine) in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As this tag changes value in the ioControl strategy, its value will determine the effects on the graphic. The tagname you enter is affected by C.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Choose whether the tag you're reading is a numeric value, a discrete value (with ON and OFF values), or a string. If you choose Value, fields D and E are activated. If you choose

Discrete, the F group is activated. The choice you make in this selection must match the choice you made in A.

If you choose Value, the value read will be converted into a string based on the rules mentioned at the beginning of this "Text In" section.

If you choose Discrete, use F to enter the strings that will be displayed for the ON and OFF states. If you choose String, the text string from the control engine is displayed.

D If you selected Value in C, enter the value to be added and subtracted from the previously read tag value to determine whether the new value is displayed.

E Select the format in which numeric values will be displayed: decimal, hexadecimal, binary, or exponential. Hexadecimal numbers are prefixed with "0X" and appear in uppercase letters. Float values may not be displayed as hexadecimal or binary.

F Select the text justification for the string from the control engine.

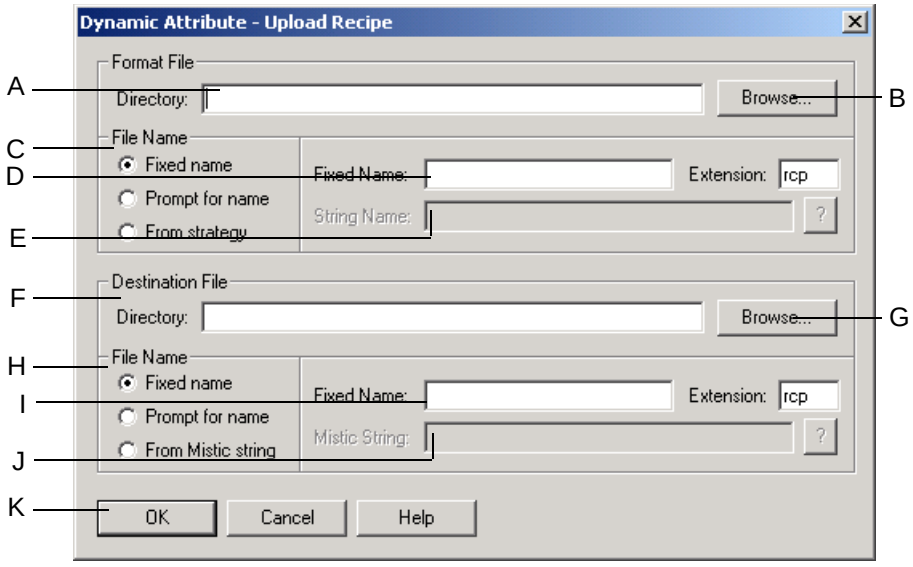
G This section is highlighted if Discrete was chosen in C. Enter the string to be displayed if the tag's discrete value is ON, and enter the string to be displayed if the tag's discrete value is OFF.

H Click here to save your settings. (Click Cancel to close the dialog box without making changes.)

Upload Recipe

Use this attribute to upload a recipe file from a control engine when graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text. For more information about recipes, see "Recipes" on page 9-19.

Double-click Upload Recipe in the Graphics Dynamic Attributes dialog box to display the following:



A Enter the directory location of the recipe format file. Use the Browse button B to quickly select a directory name. The recipe format file is used to tell the control engine what

data to read. Often, it's the original downloaded recipe. For example, the number of lines with data values represents how many values should be read from the control engine.

B Click to quickly find the recipe format file directory for A. The Select Format File Directory dialog box appears. Use it to navigate to the desired directory and click OK when you're done.

C Choose the source of the recipe format file name:

- If you choose Fixed name, D is highlighted.
- If Prompt for name is selected, the operator is prompted for the recipe format file name.
- If From strategy is selected, E is highlighted.

D If Fixed Name was selected in C, enter the name of the recipe format file located in directory A. Notice the file extension is .rcp.

E If From strategy was selected in C, use the Tag Selection button ? to enter a tagname of type string that contains the recipe format file name. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

F Enter the directory location of the recipe file that will receive the information. Use G to quickly enter a directory name.

G Click to quickly find the recipe file directory for F. The Select Destination File Directory dialog box appears. Use it to navigate to the desired directory and click OK when you're done.

H Choose the file name for the recipe data that will be read from the control engine.

- If you choose Fixed Name, I is highlighted.
- If Prompt for Name is selected, the operator is prompted for the recipe file name.
- If From strategy is selected, J is highlighted.

I If Fixed Name was selected in H, enter the name of the file to receive the information. This file will be located in F. Notice the file extension is .rcp.

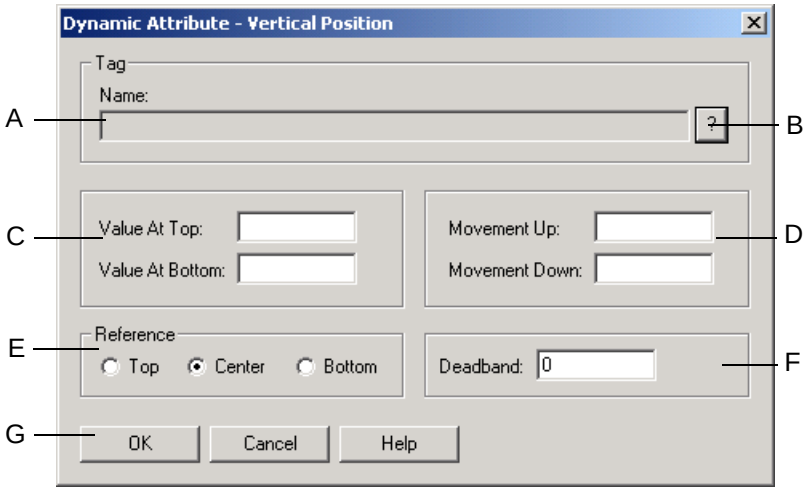
J If From strategy was selected in H, use the Tag Selection button ? to enter a tagname of type string that contains the file name. The Tag Selection dialog box is displayed so you can select a tag.

K Click OK to save your settings.

Vertical Position

Use this attribute to adjust the vertical position of a graphic based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines, rectangles, round rectangles, ellipses, and polygons, polylines, curves, bitmaps, and text.

Double-click Vertical Position in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As this tag changes value in the ioControl strategy, its value will determine the effects on the graphic.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Enter the top and bottom values for the graphic. For example, if you know your tag values will be from 0 to 100, you may wish to enter a range of 0 to 100 or a subset of this range such as 0 to 50.

D Enter the maximum top and bottom movement the object can change. The movement units are in pixels. For example, if your bottom and top values are 0 and 100, your bottom and top movements are 0 and 200, and your point of reference (E) is from the bottom when the tag value is 50, the object will be moved 100 pixels from the bottom.

E Enter the reference point for the object. This is the reference point from which the object will move. The choices are top, center, and bottom. The Vertical Size (Height) dynamic attribute must also be configured in order for this option to affect the graphic.

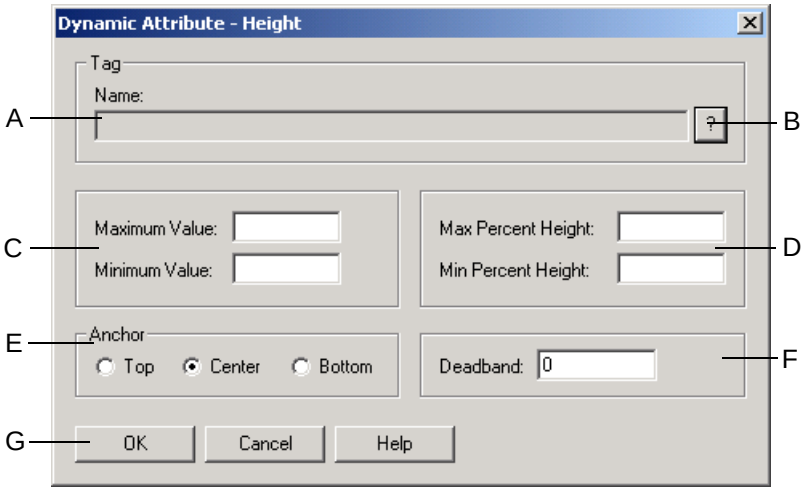
F Enter a value to be added and subtracted from the previously read tag value to determine whether the graphic's position will actually change. Using our previous example in D, let's say the deadband is 5. A tag is read and has a value of 50. The next tag reading must be greater than 55 in order for the graphic to move.

G Click OK to save your settings.

Vertical Size (Height)

Use this attribute to change the height of a graphic based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines, rectangles, round rectangles, ellipses, and polygons.

Double-click Vertical Size in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As this tag changes value in the ioControl strategy, its value will determine the effects on the graphic.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Enter the maximum and minimum value for the tagname in A. For example, if you know your tag values will be from 0 to 100, you may wish to enter a range of 0 to 100 or a subset of this range, such as 0 to 50.

D Enter the maximum and minimum percentage the object can change. The percentage range is from 0 to 1,000. Suppose your minimum value is 0 and your maximum value is 10, and the minimum and maximum percentages range from 0 to 200. When the tag value is 10, the object will be twice as big as you've originally drawn it. When the tag value is 5, the object will be the same size you've drawn it, which is 100 percent.

E Enter the anchor point for the object. The choices are top, center, and bottom.

F Enter a value to be added and subtracted from the previously read tag value to determine whether the graphic's height will actually change.

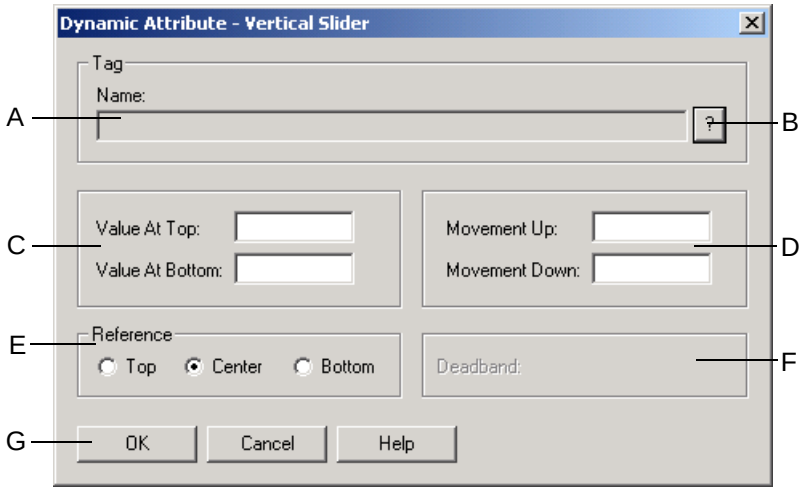
For example, let's say the values are 0 to 100, the percentages are 0 to 100, and the deadband is 5. A tag is read and has a value of 10. The next tag reading must be greater than 15 in order for the graphic to change.

G Click OK to save your settings.

Vertical Slider

Use this attribute to configure a vertical slider when a graphic is clicked. This is an operator-driven attribute and is available for lines, rectangles, round rectangles, ellipses, polygons, and bitmaps.

Double-click Vertical Slider in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. The amount of movement the vertical slider makes affects the value sent to the tag in the control engine.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Enter the top and bottom values for the tagname in A. For example, if you know your tag values can be from 0 to 100, you may wish to enter a range of 0 to 100 or a subset of this range, such as 0 to 50.

D Enter the maximum top and bottom movement the object can change. The movement units are in pixels. This means if your bottom and top values are 0 and 100, and your bottom and top movements are 0 and 200, if you move the tag 100 pixels toward the bottom, the tag value sent will be 50.

E Select the reference point for the object. The choices are top, center, and bottom. Note that you must also separately configure the Vertical Size (Height) dynamic attribute to use this option.

F Enter a value to be added and subtracted from the previously read tag value to determine whether the tag value should be changed.

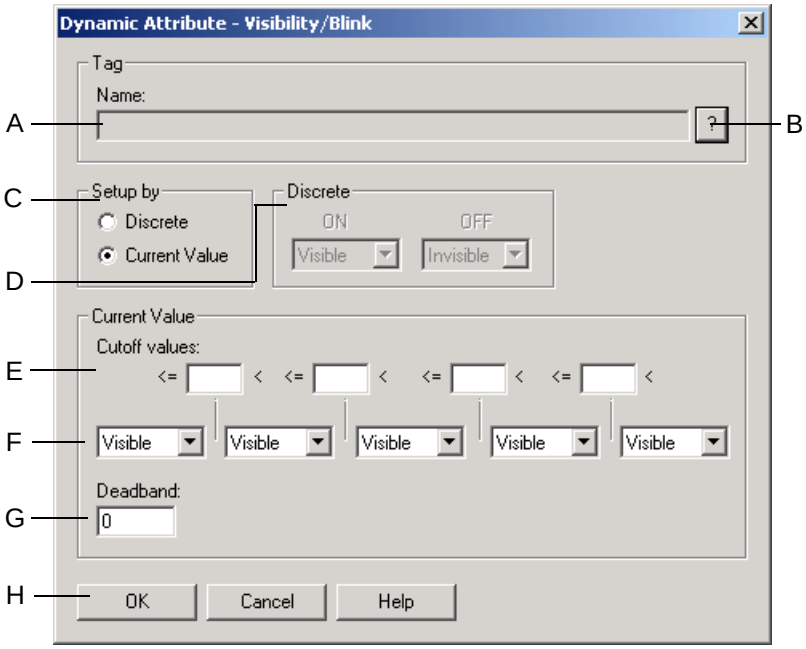
Using our previous example in D, let's say the deadband is 5. A tag is written and has a value of 50. The next graphic movement must be greater than 55 in order for the tag value to change.

G Click OK to save your settings.

Visibility/Blink

Use this attribute to make a graphic visible or invisible or to cause it to blink based on a tag value from the control engine. This is a control engine-driven attribute and is available for lines, rectangles, round rectangles, ellipses, and polygons, polylines, curves, bitmaps, and text.

Double-click Visibility/Blink in the Graphics Dynamic Attributes dialog box to display the following:



A Enter an ioControl tagname here by clicking the Tag Selection button B. As this tag changes value in the ioControl strategy, its value will determine the effects on the graphic.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 for more information about this dialog box.

C Choose whether the tag you're configuring is a discrete value (with ON and OFF values), or whether it's a current value you're looking for. If you choose Discrete, the D group is highlighted; if you choose Current value, the E group is highlighted. The choice you make in this selection must match the tag choices you made in A.

D Select the visibility state you want for the ON state from the drop-down list options. Your choices are Invisible, Visible, Slow Blink, Med. Blink, and Fast Blink. In the same manner, choose the visibility state for the OFF state.

E Enter a value in each Cutoff Value field to specify the range of values for each visibility group configured in the visibility fields. Values entered must be in increasing order. Each Cutoff Value field must have a numeric value entered or you will get a warning to enter one. After the warning, the cursor blinks in the first Cutoff Value field requiring a value.

F To configure a visibility state for the ranges set up in E, click a visibility field's drop-down button. Your choices are Invisible, Visible, Slow Blink, Med. Blink, and Fast Blink. In the same manner, choose the visibility state for each range of values.

G Enter a value to be added and subtracted from the previously read tag value to determine whether the graphic's visibility will actually change.

For example, let's say the Cutoff values are 1, 10, 20, and 30, and the visibility fields are Invisible, Visible, Slow Blink, Med. Blink, and Fast Blink. The Deadband is 2. A tag is read

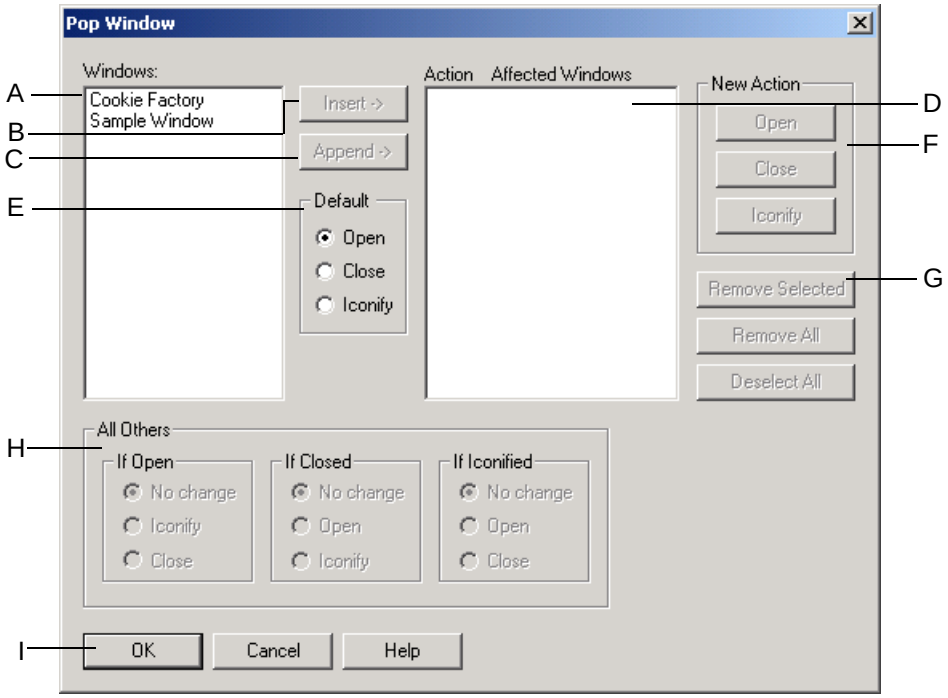
with a value of 10. The next value read must be greater than 12 in order for the graphic to change.

H Click OK to save your settings.

Windows

Use this attribute to change window states when a graphic is clicked. This is an operator-driven attribute and is available for rectangles, round rectangles, ellipses, polygons, bitmaps, and text.

Double-click Windows in the Graphics Dynamic Attributes dialog box to display the following:



A Initially, this list shows all the windows available for the project. Select the window that has the window state you want to change. You can also select more than one window by using a couple of key combinations. One way is to select a window, press SHIFT, and click on another window name in the list. This selects all the windows in between the two window names. Another way is to select a window name, press the CTRL key, and click on each window name you want selected.

B Use to insert the selected window(s) in D. The window name will be inserted above the highlighted window(s) in D.

C Use to add the selected window(s) to the bottom of the window list in D. The window name will be inserted after the last window in D.

D Lists the windows and what window state they will go to when the graphic is clicked.

E Use this group as a handy window state assignment key for the windows you've highlighted in A. When the windows are copied over to D, you will see they all have the default window state you've assigned. The window state choices are Open, Close, and Iconify.

F Use this group to change any window states listed in D. Highlight one or more windows and select the new window state action. Your choices are Open, Close, and Iconify.

G Use this group to modify the window name list in D. The Remove Selected option removes only those windows you've highlighted; the Remove All option removes all windows from the list; and the Deselect All option deselects and quickly unhighlights all window names.

H For all windows left in A, use this grouping to affect their window states. For example, if a window state is open and the graphic is clicked, the window state can stay the same, be iconified, or be closed. Likewise, you can alter the window states of closed and iconified windows.

I Click OK to save your settings.

Copying and Deleting Dynamic Attributes

Once a dynamic attribute has been added to an on-screen graphic object, you can easily assign those attributes to other graphics using copy and paste. You can also quickly delete all dynamic attributes from a graphic.

Copying Dynamic Attributes from a Graphic

To copy a set of dynamic attributes from one object to another, do the following:

1. With the Select tool  , click the graphic whose dynamic attributes you want to copy.
2. Choose Edit→Copy Dynamic Attributes to copy the attributes to the Windows clipboard.

   Another way you can do this is to right-click on the object and select Dynamic Attributes→Copy.

   You can now assign these dynamic attributes to another graphic using Edit→Paste Dynamic Attributes.

Pasting Dynamic Attributes to a Graphic

To paste a set of dynamic attributes that were copied to the Windows clipboard, do the following:

1. With the Select tool  , choose one or more graphics to which you want to paste the dynamic attributes.
2. Choose Edit→Paste Dynamic Attributes, and then do one of the following:
   - To delete any existing dynamic attributes the object may already have, choose Delete Existing.
   - To add the copied attributes to existing attributes and replace any that might be of the same type, select Replace Duplicates.

- To add the attribute to existing attributes and not change any attributes that might be of the same type, select Ignore Duplicates.

You can also perform these tasks by right-clicking on the object and selecting the paste option from Dynamic Attributes→Paste.

Deleting Dynamic Attributes from a Graphic

To delete all dynamic attributes from a graphic, do the following:

1. With the Select tool  , choose one or more graphics that have dynamic attributes you want to delete.
2. Choose Edit→Delete Dynamic Attributes.

   Another way you can do this is to right-click on the object and select Dynamic Attributes→Delete Existing from the pop-up menu.

Viewing Tags and Dynamic Attributes

As you develop or document an ioDisplay project, it can be useful to know about the dynamic attributes and tags used by the objects in the project's windows.

Dynamic Attributes for Individual Objects

To view the dynamic attributes configured for an individual graphic object or a group of objects, select one or more objects in a window, right-click the mouse, and select Dynamic Attributes→View from the pop-up menu. The dynamic attributes assigned to that object or objects will be shown in the Dynamic Attributes window.

Viewing Tags for One or More Objects

To view the tags that a particular object or objects is connected to, click the object (or group of objects), and move the cursor over it. The tags used will be shown next to the cursor.

Dynamic Attributes for All Objects

To generate a report that lists the dynamic attributes for all objects in all project windows, as well as all the alarm points configured for a project, do the following:

1. In ioDisplay Configurator, select View→Dynamic Attributes.
2. In the dialog box that opens, select one or more windows whose dynamic attributes you want to view and click OK.

The Microsoft Windows Notepad application opens and displays a report similar to the following example:

```
Project Name: C:\Opto22\ioDisplay\Examples\ioCookies\Display\cfactory.uui

------------------------------------------------------------------------
                        Window: Cookie Factory
                        **********************
Group            X = 530   Y = 332   W =  86   H =  28

Operator-Driven Dynamic Attribute Tags
Send Value:
Cookie Control Engine:Conveyor_Speed_Control.Value
Refresh Group: Group 0
  Source     : Prompt for Data
  Destination: Set
  Message    : Enter Conveyor Speed
  Min. Value : 0.00
  Max. Value : 100.00

Group            X = 133   Y = 213   W =  21   H =   3

control engine-Driven Dynamic Attribute Tags
Visibility/Blink:
Cookie Control Engine:Dough_Dispense_Valve.State
Refresh Group: Group 0
  Setup By : Discrete
  On       : Invisible
  Off      : Visible
```

Using the TagInfoView Utility Program

After you have selected View→Dynamic Attributes, you can use a small utility program called TagInfoView to sort and view the tag information in greater detail.

1. Select View→Launch TagInfoView Utility.
2. In the Tag Information Viewer dialog box that opens, verify that the TagInfo.txt file appears in the file field at the top.
3. In the Sort By sections, select how you want the tag information sorted.
4. Click Display Results.

Scanning to Update Graphics

As you configure your project and connect ioControl tags to ioDisplay objects, you are setting up the connections that will animate your graphics in ioDisplay Runtime as the tag data changes. ioDisplay acquires this tag data using an internal scanner that monitors one or more control engines. Understanding how the scanner works and how it gets its data will help you optimize your system's performance.

Scanning is ioDisplay's process of requesting data about I/O points and variables from the Opto 22 control engine. When the control engine receives this request, it must first determine if its data for the requested tags is current. If the data is not current, the control engine will access the I/O units connected to it and request the latest readings. This tag information is then sent back to ioDisplay. Depending on the data and how it is connected to objects in your ioDisplay project, graphic objects then change their attributes.

Refresh Time Groups

Each tag connected to an ioDisplay graphic belongs to a refresh time group that determines how often the tag is scanned. You define the characteristics of refresh time groups by setting a scan rate. See "Configuring Scan Rates" on page 7-35 to learn how to set these values.

System performance is affected by how a refresh time group is set up, so it's important to define refresh time groups carefully. Follow these guidelines when configuring refresh time groups for your project:

- Select scan rates that reflect the rate at which the process variables change. For example, the outside air temperature changes slowly, and could be scanned every 15 minutes or so.
- As a good engineering practice, select the slowest possible scan rate that is acceptable for your application. This will help prevent the system from being overloaded by needlessly scanning too much information too quickly.
- If the amount of data being scanned is too much for the selected scan rate, decrease the scan rate to better match what the actual throughput will allow.

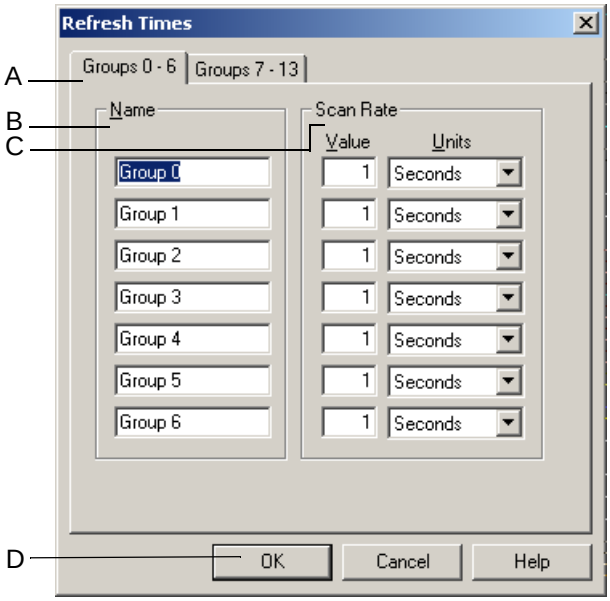
  If you have tried the preceding guidelines and scanning speed does not improve, then there may be too much data being requested from one control engine, or there may be too much data being scanned by one computer, or both. To eliminate the bottlenecks, you may need to add more control engines, divide the ioDisplay projects over multiple computers, or both.

Configuring Scan Rates

You assign refresh time groups whenever you configure a tag based on how often you think it makes sense to update the tag. (Note that the same refresh time group is usually used by several tags.) For example, an analog point reading an outside temperature would use a longer scan rate since the temperature isn't likely to change suddenly. On the other hand, a digital point that's monitoring the on/off state of a valve would need a shorter scan rate to accurately reflect whether a valve is open or closed.

To configure scan rates, choose Configure→Refresh Times.

The Refresh Times dialog box appears:



A Up to fourteen refresh time groups are available, divided between two tabs in the Refresh Times dialog box.

B In the Name column, you will see the refresh time group names. By default, their names are “Group 0", “Group 1", and on up to “Group 6." You can change these default names if you wish.

Names can be up to 15 characters long, but avoid using the ! and | characters in the name. Spaces are also valid characters, but when referencing names with spaces, don’t omit the spaces or substitute the “_” character for spaces.

C In the Value field of the Scan Rate column, enter a number from 1 to 9999. In the Units field, select a unit of time from the Units drop-down list. (Unit options are milliseconds, seconds, minutes, hours, days, and months.)

The greater the value, the more time between I/O readings and the fewer times the control engine is scanned.

D Click OK to save your settings.

CHAPTER 8

Working with Trends

Introduction

This chapter describes how you can create and configure graphs that display real-time and historical information about selected I/O points.

In This Chapter

About Trends

ioDisplay *trends* are graphical objects that visually plot control engine values, including I/O points and results of a calculation. Using a trend, you can show how real-time and historical data changes over time, or how one set of data relates to another one. In ioDisplay Runtime, trends are created when active I/O point values, or tags, are read from an ioControl strategy running on a control engine and visually plotted.

Tag values are graphed on a two-dimensional x-axis and y-axis coordinate system. Depending on the type of trend used, the x-axis can represent time or a set of tag values. The y-axis can represent either the range of values for a tag or a set of tag values. You can set features for each trend line used, and there is no limit to the number of trends that can be displayed in a window. For trends that graph data over time, the maximum time span supported is 14 days.

You can easily set or modify the following elements in a trend:

- X- and y-axis ranges and the major and minor divisions that appear on the chart
- Graph backgrounds and border colors
- Pen colors for trend lines
- Tag value scanning, which can be turned on or off for a specific trend.

Types of Trends

There are three types of trends that you can use in an ioDisplay project: basic trends, SuperTrends, and XY plots. Basic trends and SuperTrends graph control engine values over time, but support different numbers of trend lines and have other differences. XY plots graph data from two numeric tables, using one set of data for x-axis values and the other for y-axis values.

- **Basic Trends**—Using a basic trend, up to four trend lines can be displayed on any one trend chart. Unlike a SuperTrend (described below), which can graph historical data, basic trends can only graph real-time data. See "Working With Basic Trends" below for more information.
- **SuperTrends**—Using a SuperTrend, up to 16 trend lines can be displayed on any one trend chart. SuperTrends can graph both real-time and historical data. See "Working with SuperTrends" on page 8-7 for more information.
- **XY Plots**—Using an XY plot, up to six individual trend lines can be displayed on any one XY plot. XY plots can only graph data in numeric tables. See "Using XY Plots" on page 8-21 for more information.

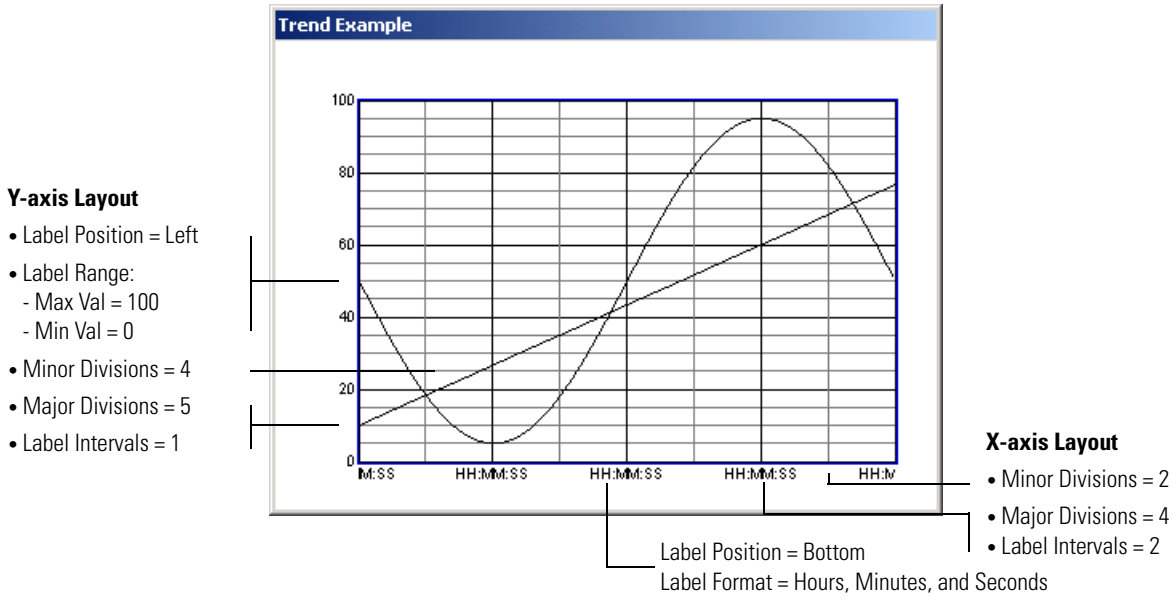
Working With Basic Trends

You can use basic trends to graph real-time tag information using up to four pens.

Creating a Basic Trend

1. Select the basic trend tool  from the toolbox and position the cursor where you want the trend to begin in the window.
2. Click the mouse button, drag the mouse to the desired size, and then release the mouse button.

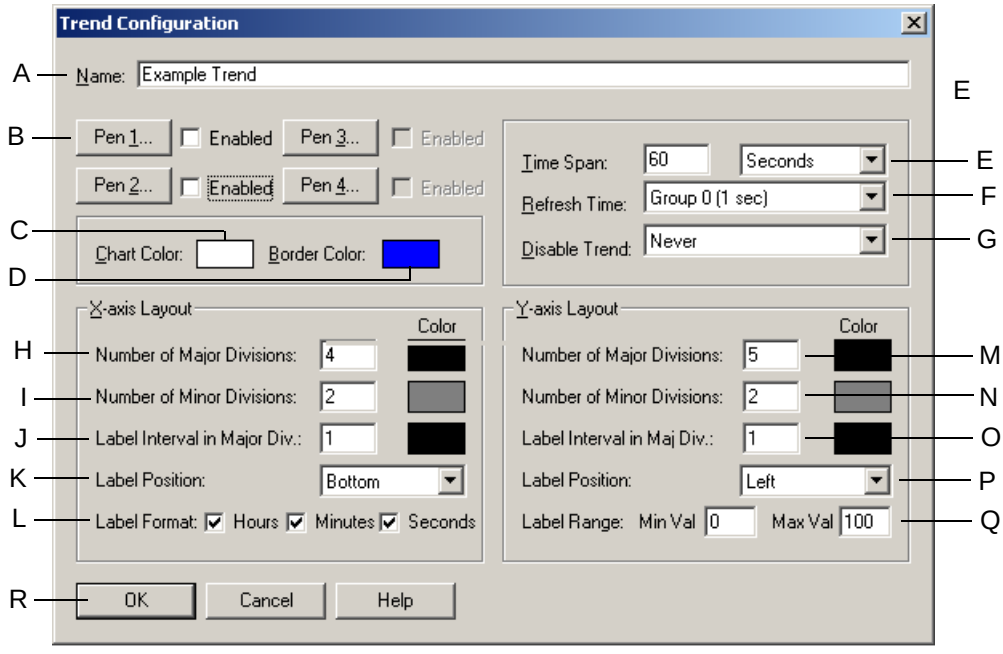
The trend that appears should resemble the example below:



Modifying a Basic Trend

Choose the Select tool and double-click the trend.

The Trend Configuration dialog box appears:



A Enter the name of the trend here.

B Use these pen configuration buttons to associate each pen with one ioControl tag, to define its pen color and width, and to specify a range of values for the tag. You can configure up to four pens per trend. See "Configuring Basic Trend Pens" on page 8-5 for more information.

C Click here to enter the chart's background color. In the Color dialog box that appears, choose a color and click OK.

D Click here to enter the chart's border color. In the Color dialog box that appears, choose a color and click OK.

E Enter the time span the trend represents. (Remember that the time span is indicated on the x-axis). Choose the time units from the drop-down list. Your choices are seconds, minutes, or hours. The maximum time span is 14 days, or 336 hours.

F Select the refresh time here. Choose from one of seven refresh time groups. The scan rate appears in parentheses alongside the refresh group number. All tags associated with the pens in B are scanned at the same rate.

You can find out more about refresh times in "Scanning to Update Graphics" on page 7-35. Also see "Optimizing Pen Settings" on page 8-6 to learn how pen settings affect how ioDisplay communicates with a control engine.

G Choose whether to disable a trend based on the state of its window. If you disable the trend, the tags associated with the pens in B won't be updated with new data from the ioControl strategy until the trend is enabled again. Disabling a trend saves the control engine processing time, since it doesn't have to respond to regular requests from ioDisplay for tag updates.

NOTE: Trends that are enabled are always updated, regardless of the window's visual state (normal, iconified, etc.). This means ioDisplay continually requests data from the control engine to update its trends. Keep this in mind when you're considering the number of enabled trends you are including in the project. The more enabled trends you have, the more the control engine has to spend time reading its I/O to update the data.

H Enter the number of major x-axis divisions for the trend. This is the number of main sections the trend is divided into. You can also choose a color for the major divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

NOTE: For both x- and y-axis divisions, the lines dividing the major divisions appear thicker than the minor divisions.

I Enter the number of minor x-axis divisions for the trend. This will be the number of sections the major divisions are divided into. The minimum number of divisions is one.

You can also choose a color for the minor x-axis divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

J Enter how often you want the major x-axis labeled.

NOTE: For both x- and y-axis labels, if you enter 1, every major division is labeled; if you enter 2, every other major division is labeled.

K Choose the label position for the x-axis. By default, the x-axis is labeled at the bottom, but you can choose to label the top, top and bottom, or have no labeling at all.

L Enter the label format by checking off any combination of hours, minutes, and seconds. The label appears in the following format: HH:MM:SS, where HH is hours, MM is minutes, and SS is seconds.

M Enter the number of major y-axis divisions for the trend. This is the number of main sections the trend is divided into. You can also choose a color for the major divisions by clicking the Color field. In the dialog box that appears, choose a color and click OK.

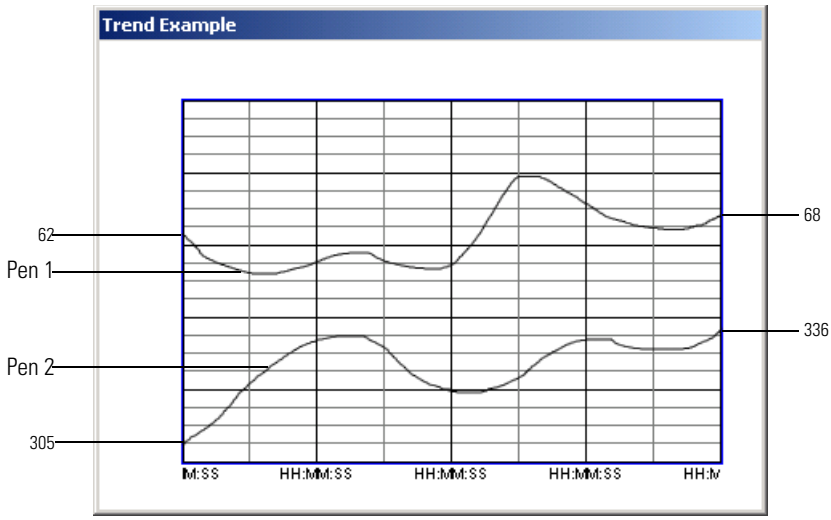
N Enter the number of minor y-axis divisions for the trend. This will be the number of sections the major divisions are divided into. The minimum number of divisions is one.

You can also choose a color for the minor y-axis divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

O Enter how often you want the major y-axis labeled.

P Choose the label position for the y-axis. If you choose Left, Right, or Left and Right, only pens that fall within the minimum and maximum values of the Label Range are displayed; pen data outside the range is not displayed.

However, if you choose None, no labels are shown on the y-axis, multiple ranges appear to overlap, and all pens are displayed. As shown in the following example, if pen A has a range of 0-100 and pen B has a range of 300-400, the y-axis has a range of 0-100 *and* 300-400 simultaneously and both pens are displayed.



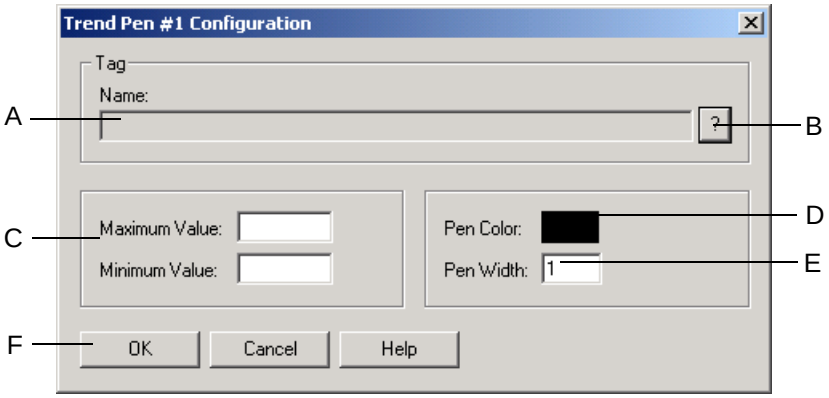
Q Enter the minimum and maximum values for the y-axis.

R Click OK to save your settings.

Configuring Basic Trend Pens

Click a pen configuration button in the Trend Configuration dialog box.

The Trend Pen Configuration dialog box appears:



A Enter an ioControl tagname here by clicking B.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag. See "Configuring Tags" on page 5-11 to learn more about configuring tags in your project.

C Enter the maximum and minimum value the tag selected in A can be.

D Click here to choose a pen color. In the Color dialog box that appears, choose a color and click OK.

E Enter the width of the line you want created by the pen. The width is specified in pixels.

F Click OK to save your settings.

Optimizing Pen Settings

The scan rates you select when configuring a pen can impact the speed and performance of your ioDisplay project and the control engine. When selecting scan rates for long trends, you should choose optimal scan times for the pens as follows:

1. Determine the trend width in pixels by using the X: value in the Coordinates window.
   a. Subtract the X: value of the left trend border from the X: value of the right trend border.
   b. Divide the trend width in seconds by the trend width in pixels.

   For example, an eight-hour trend that is 500 pixels wide yields a number of 57.6 seconds (8 x 3600/500).

2. Round this to the nearest increment of 10.

   In the example, the value would be rounded up to 60 seconds. This means that for optimum ioDisplay performance, the trend should not be updated more frequently than every 60 seconds.

   NOTE: A small compromise to increase the accuracy of the trend is to divide the result by four. In our example, this would result in a scan time of every 15 seconds. Using a 15-second scan time allows a maximum of four readings per pixel on the trend. Updating any faster than

that is counterproductive; the excess data is discarded, and the extra requests for data from the control engine add overhead to the control system.

Working with SuperTrends

SuperTrends are trends that can track both real-time and historical data. You can also use up to 16 pens with a SuperTrend; basic trends support only four pens. SuperTrends are drawn and configured in ioDisplay Configurator, like any other on-screen object, but the charts can also be manipulated by the operator in ioDisplay Runtime. See "Using Runtime" on page 10-14 to learn how to use SuperTrend options in Runtime.

Memory Requirements for SuperTrend Pens

When several SuperTrends, each using several pens, are used in an ioDisplay project, the memory requirements for the PC running the project can become very high. Keep this in mind when developing your ioDisplay project. Use the following formula to determine the amount of RAM required for each pen in a SuperTrend:

$$\text{RAM required per pen (bytes)} = \frac{19.2 \cdot \text{SuperTrend's x-axis time span (sec.)}}{\text{SuperTrend's scan rate (sec.)}}$$

For example, the memory required for each pen in a SuperTrend having a six-hour scan time (21,600 sec.) and a 500 ms scan rate (0.5 sec.) would be calculated as follows:

$$\frac{19.2 \cdot 21600 \text{ sec.}}{0.5 \text{ sec.}} = 829{,}400 \text{ bytes}$$

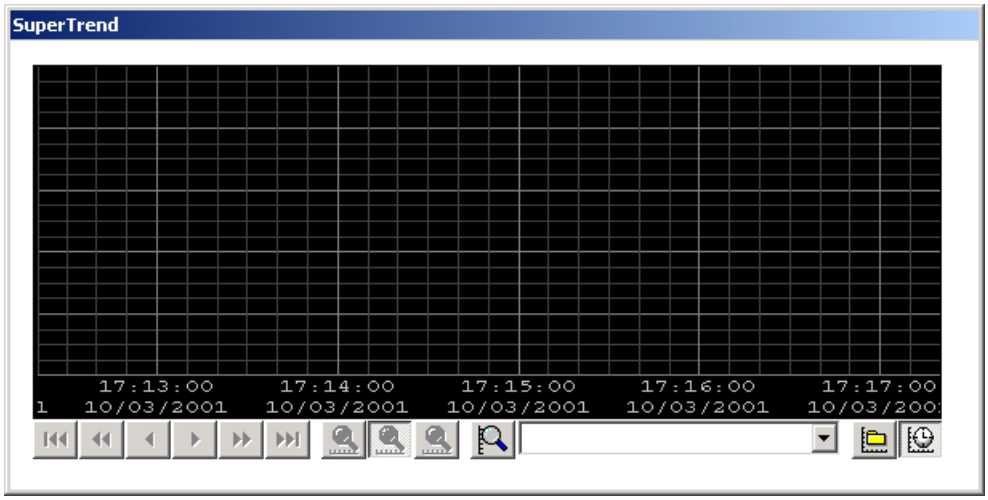
This is approximately 800 KB for each pen. If this SuperTrend contained five pens, for example, it would use 4,000 KB of RAM, or just under 4 MB.

Each time you configure a SuperTrend, the approximate amount of RAM required for that object will be shown.

Creating a SuperTrend

1. Select the SuperTrend tool  from the toolbox and position the cursor where you want the trend to begin in the window.
2. Click the mouse button, drag the mouse to the desired size, and then release the mouse button.

The SuperTrend that appears should resemble the example below:

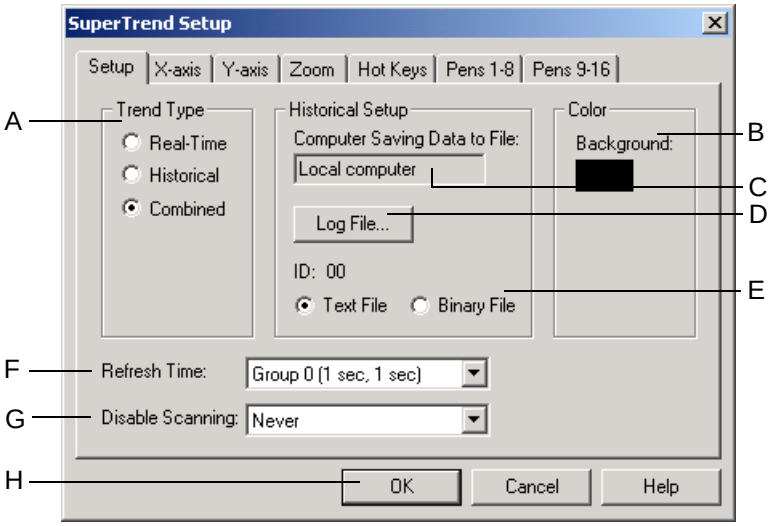


If you do not draw the SuperTrend wide enough, some of the command buttons will be placed in a second row.

Modifying a SuperTrend

Choose the Select tool and double-click the trend.

The SuperTrend Setup dialog box appears:



A Choose the trend type here. If you select Combined, you can switch between real-time and historical trending when the project is running.

B Click here to enter the chart's background color. In the Color dialog box that appears, choose a color and click OK.

C This field shows which computer will collect SuperTrend data. If "Local Computer" appears, SuperTrend data is collected by the computer running ioDisplay Runtime. If the name of another computer appears, it is a remote computer that has been previously selected. See "Using SuperTrend Log Files" on page 8-15 to learn how to choose this remote computer.

D Click here to set up the historical log file. (You don't need a historical log for real-time trends.)

E Select whether the historical log will be saved to a file in text or binary format. When the SuperTrend data is saved in ASCII text format, the file includes a header that contains information about tags assigned to the SuperTrend pens. When the data is saved in binary format, the file does not include this header. Historical trend information saved in a binary file is usually graphed more quickly than trend information that is saved in a text file.

If there are existing data files for a SuperTrend and the file format is switched between text and binary, the Select Log File To Convert To dialog box will appear. In this dialog box, you can select a log file to convert to the appropriate format. See "Saving a Log in Text or Binary Format" on page 8-19 to learn more about switching between text and binary file formats.

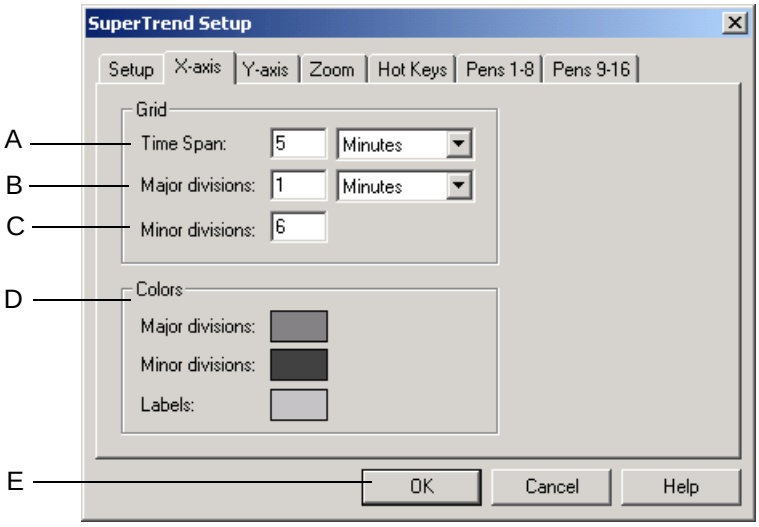
F Select the refresh time here. You can choose from one of seven refresh time groups. The scan rate appears in parentheses alongside the refresh group number. You can find out more about refresh times in "Scanning to Update Graphics" on page 7-35. All tags associated with the pens in this trend will be scanned at the same rate.

G Choose whether to disable a trend based on its window's state. If you disable the trend, the tags associated with the pens won't be updated with new data from the ioControl strategy until the trend is enabled again. Disabling a trend saves the control engine processing time by not having to respond to regular requests from ioDisplay for tag updates.

H Click OK to save your settings.

Configuring X-Axis Parameters

Click the X-axis tab to configure x-axis parameters for the SuperTrend.



A Enter the time span (x-axis) the trend represents. Choose the time units from the drop-down list. Your choices are seconds, minutes, hours, and days. The maximum time span is 49 days, or 1,176 hours.

B Enter the number of major x-axis divisions for the trend. The trend will be divided into this number of main sections. You can choose a color for the main section dividers in D. The lines dividing the main sections will appear thicker than the lines dividing the minor sections.

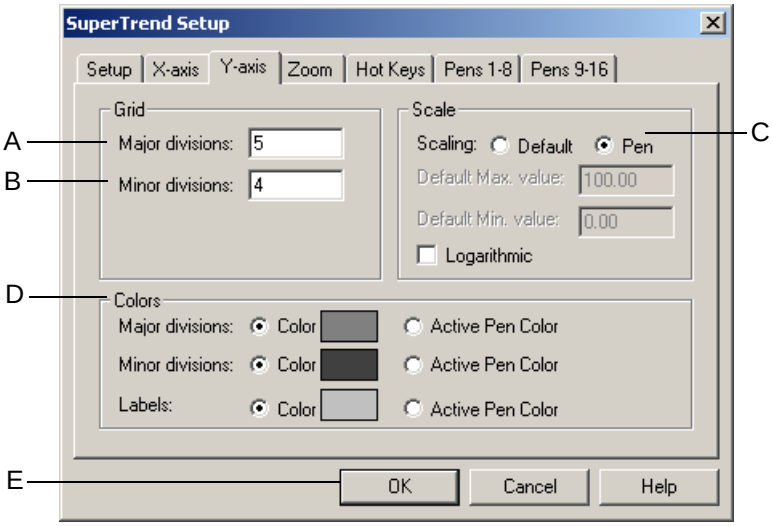
C Enter the number of minor x-axis divisions for the trend. The major sections will be divided into this number of minor sections. The minimum number of divisions is one. You can choose a color for the minor section dividers in D.

D Click on the boxes to set the colors for the major divisions, minor divisions, and labels. In the Color dialog box that appears for each item, choose a color and click OK.

E Click OK to save your settings.

Configuring Y-Axis Parameters

Click the Y-axis tab to configure y-axis parameters for the SuperTrend.



A Enter the number of major y-axis divisions for the trend.

B Enter the number of minor y-axis divisions for the trend.

C Click here to select the scaling.

- If you choose Default, the trend will always have the same fixed scale. You must enter values for the Default Max. Value and Default Min. Value. These values specify the range of pen tag values. This scaling is static and cannot be changed in Runtime.
- If you choose Pen, the trend will have the scale of the active pen.
- If you select the Logarithmic checkbox, the trend will have a logarithmic y-axis based on the scale of the active pen. The logarithmic y-axis is determined by rounding down the minimum pen value to the next lowest factor of ten, and rounding up the maximum pen value to the next greatest factor of ten. For example, if a pen's range has a minimum value of 500 and a maximum value of 8500, the y-axis will have a minimum value of 100 and a maximum value of 10,000.

D Click on the boxes to set the colors for the major divisions, minor divisions, and labels in the trend. If you want the y-axis to use the color of the pen that's currently active, select Active Pen Color.

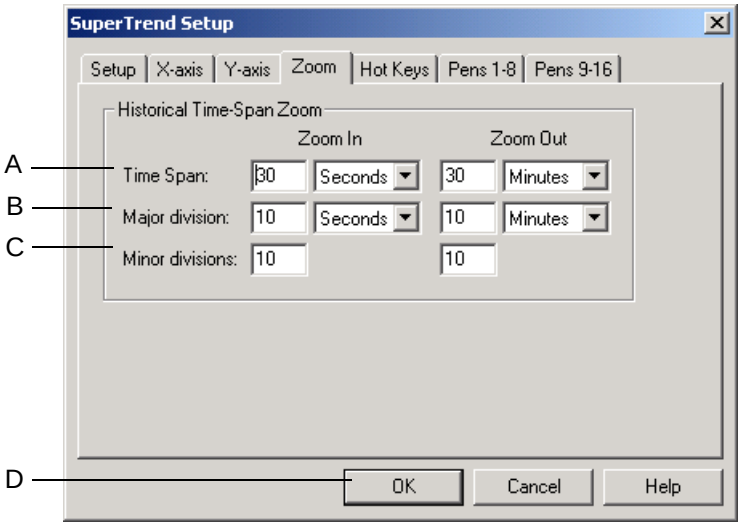
In the Color dialog box that appears for each item, choose a color and click OK.

E Click OK to save your settings.

Configuring Zoom Parameters

When a SuperTrend is in historical mode, the operator can zoom in to view a more detailed section of the trend, or zoom out to view a less detailed section of the trend. Click the Zoom tab

to set magnification levels. The Zoom page, shown below, configures the zoom levels by defining scales for the x-axis.



A Enter the amount of time to display on the trend while in zoom in/zoom out mode. Unit choices, selected from the dropdown box, are seconds, minutes, hours, or days.

B Enter the amount of time that constitutes one major division while in zoom in/zoom out mode. Unit choices, selected from the dropdown box, are seconds, minutes, hours, or days.

C Enter the number of minor divisions per major division while in zoom in/zoom out mode. This number must be 1 or greater.

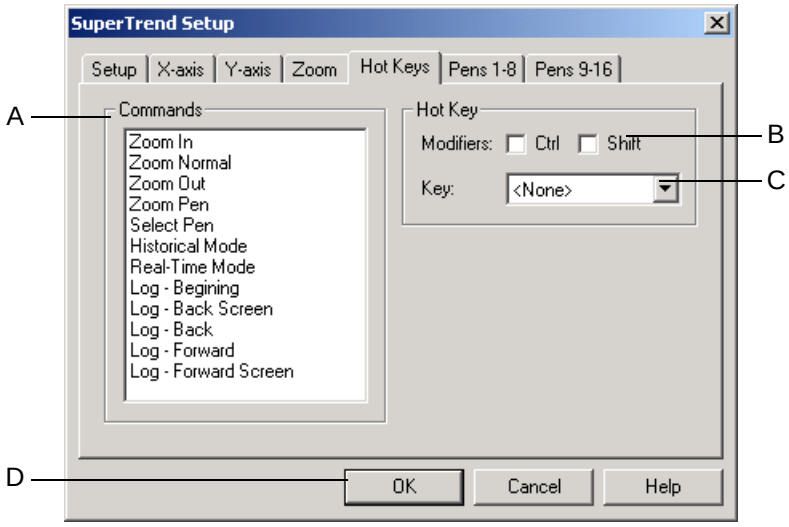
D Click OK to save your settings.

Configuring Hot Keys

All of the buttons that appear at the bottom of a SuperTrend can be associated with a combination of keystrokes called a *hot key.* If a button has a hot key configured, the operator can either click on the button or press the hot key combination. This way the operator can use the SuperTrend without needing a mouse.

NOTE: If a command is only available in historical mode, its associated hot key will also work only in historical mode.

To configure hot keys, click the Hot Keys tab.



A This field displays the Runtime commands that may be associated with a hot key. Highlight a command to configure a hot key for it, or to view the existing hot key settings.

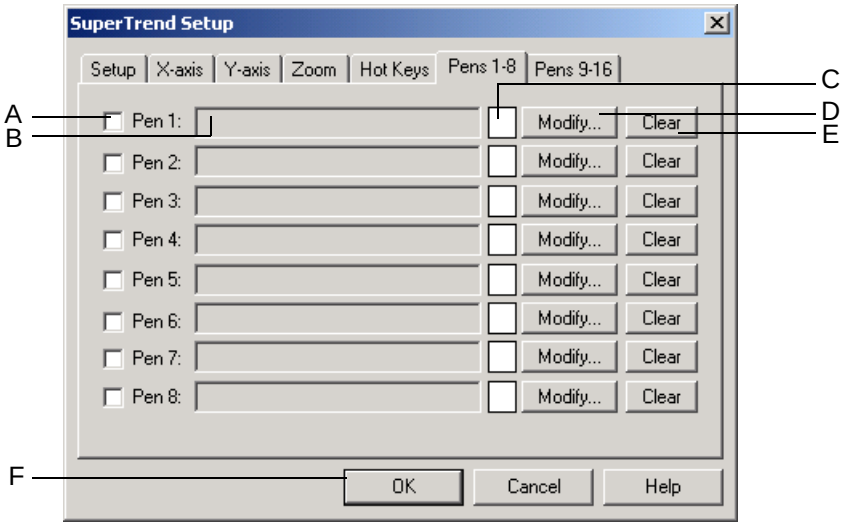
B Check the box labeled Ctrl if you want the CTRL key to be part of the hot key. Check the box labeled Shift if you wish the SHIFT key to be part of the hot key. You may select none, one, or both.

C This drop-down box lists all available keys that can be hot keys. Make sure that you do not assign a hot key to more than one SuperTrend command.

D Click OK to save your settings.

Configuring SuperTrend Pens

Unlike a basic trend, a SuperTrend can plot trend lines for up to 16 tag values (pens). Pens 1 through 8 are configured under the "Pens 1-8" window shown below. Pens 9 through 16 are configured identically under the tab "Pens 9-16." There is one row for each pen.



A Check this box to enable the pen in this row (uncheck the box to disable the pen). The pen is enabled automatically when it is first created.

B Once a pen is configured, its name appears here.

C Click here to select the pen color. In the Color dialog box that appears, choose a color and click OK.

D Click here to configure a pen or to modify its configuration. The SuperTrend Pen # configuration dialog box appears. Configure the pen and click OK. The pen name will be listed in B.

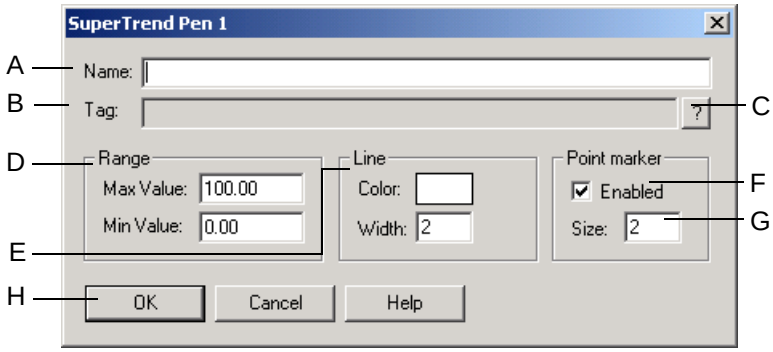
E Click here to clear the pen configuration.

F Click OK to save your settings.

Setting an Individual Pen

Click the Modify button for a pen in the SuperTrend Setup dialog box.

The SuperTrend Pen dialog box appears. (The number of the pen to be configured is shown in the title bar.)



A Enter a SuperTrend pen name, if desired. By default, when a tag is selected in B, the name will be the same as the tagname. In Runtime, this name is displayed in the SuperTrend's Active Pen drop-down list.

B Enter an ioControl tagname here by clicking the Tag Selection button in C.

C To enter a tagname in B, click here and select a tag in the Tag Selection dialog box that appears. See "Configuring Tags" on page 5-11 for more information about selecting and configuring tags.

D These fields configure the range of displayed data points:

- The Max Value, or maximum value, is the tag value that will position the pen at the top of the trend. If a scanned value of the tag is greater than the Max Value, the point will be plotted but will not be visible on the trend.
- The Min Value, or minimum value, is the tag value that will position the pen at the bottom of the trend. If a scanned value of the tag is less than the Min Value, the point will be plotted but will not be visible on the trend.

E Click in the box next to Color to choose a pen color. In the Color dialog box that appears, choose a color and click OK. Enter the width of the line you want created by the pen. The width is specified in pixels.

F Click here to have point markers displayed for every scanned point. Point markers make it easy to identify scanned data. In historical mode, you can click on a scanned point to display its data (value, time scanned, etc.).

G Choose a point marker width, which is measured in pixels.

H Click OK to save your settings.

Using SuperTrend Log Files

Tag value data that is graphed in the SuperTrend chart can be saved to historic log files for later viewing. Using the buttons and other controls on a SuperTrend chart, the operator can open and view these log files while the project is running in ioDisplay Runtime. This section shows how to configure SuperTrend settings in your project to define how and where SuperTrend log files are

saved. To learn how to use SuperTrend controls, see "Using SuperTrends in Runtime" on page 10-21.

NOTE: SuperTrend historic log files are similar to historic data logs, but are created and saved as separate files due to the additional pen information that a SuperTrend can collect.

Choosing a Computer to Save SuperTrend Log Files

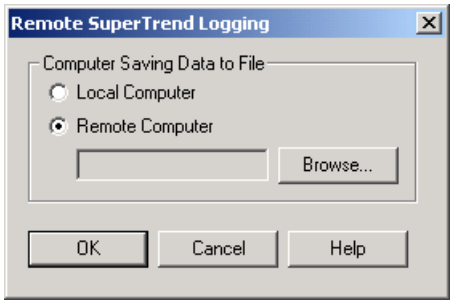
Follow the steps below to choose a local or remote computer that will save the SuperTrend log files. If you select a remote computer, it must be running the same ioDisplay project as the local computer.

*NOTE: Choosing a computer to save SuperTrend log files does **not** select the location where these files are saved; it only designates the computer that will do the work of saving the file.*

If you plan to run the same ioDisplay project on multiple computers *and* these computers will save SuperTrend data to the same location, it is recommended that you select a remote computer—also running the project—to improve the performance of SuperTrend graphing and file access.

1. Select Configure→Remote SuperTrend Logging.

   The Remote SuperTrend Logging dialog box appears.



2. Do one of the following:

   - To have the local computer save SuperTrend data to a file, select Local Computer and click OK.

     The Local Computer setting will apply to each computer that is running this ioDisplay project, which means that each computer will save a SuperTrend log file.
   - To have a remote computer running the ioDisplay project save SuperTrend data to a file, do the following:

     i. Select Remote Computer and click Browse.

     ii. Navigate in the file tree that appears (similar to Windows Explorer) until you find the computer that will save SuperTrend data.

     iii. Select that computer in the list and click OK.

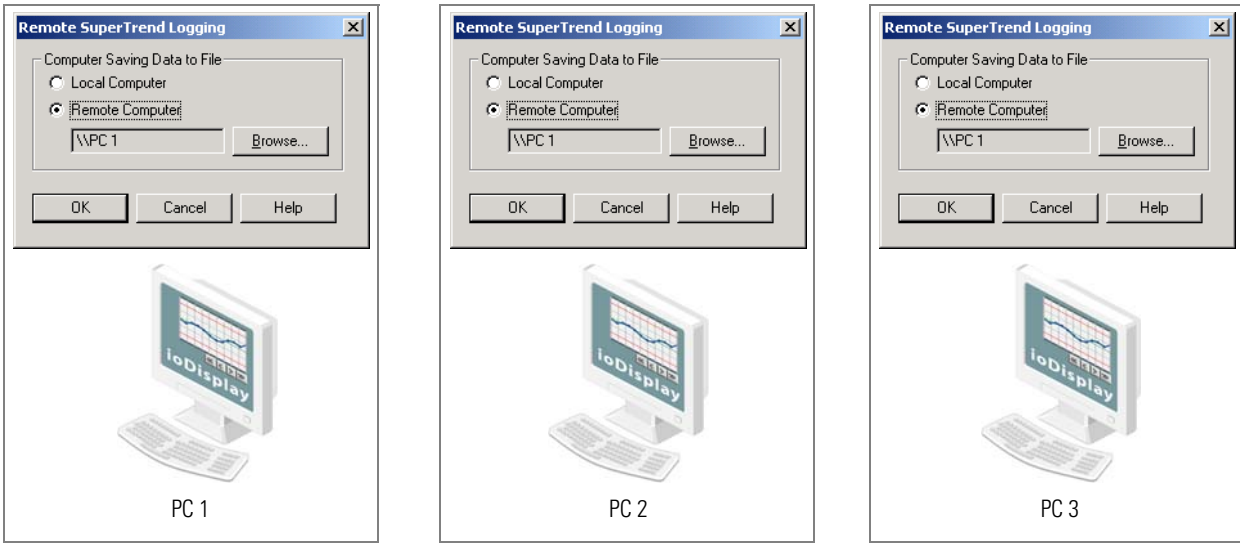
     When your project is running, only that remote computer will save SuperTrend data to the log file. The Remote Computer setting will apply to each computer that is running

this ioDisplay project. With this setting selected, each computer will display the data, but will **not** save it to a file.

Remote SuperTrend Logging Example

The following example shows one way that remote SuperTrend logging might be configured for multiple computers running the same ioDisplay project.

The illustration below shows the required settings to have the computer PC 1 save SuperTrend log files. With these settings, computers PC 2 and PC 3—all running the same ioDisplay project—would display SuperTrend data on-screen, but would not save SuperTrend log files.



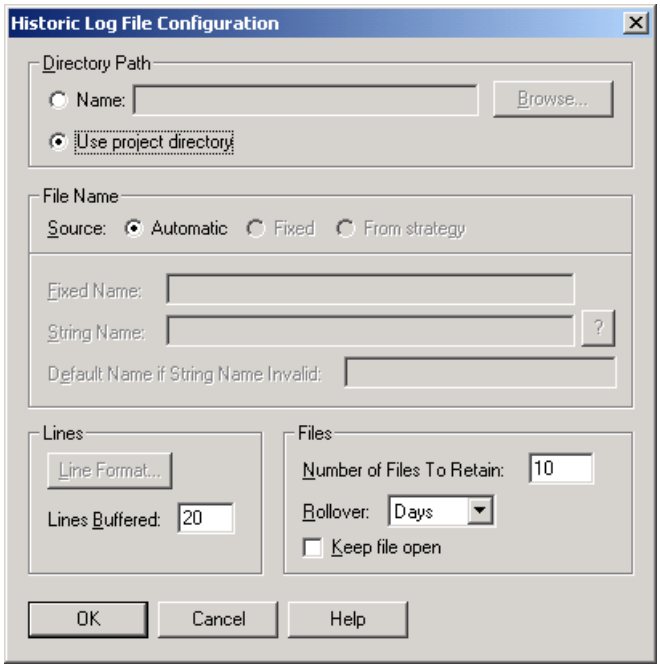
Choosing a Location for SuperTrend Log Files

After selecting a local or remote computer that will save SuperTrend log files, you must select the location where these files will be saved. SuperTrend log files can be saved in the project directory of the computer running the ioDisplay project, or in a remote directory (for example, on a network server).

To choose where SuperTrend log files will be saved, do the following:

1. Double-click on a SuperTrend to open the SuperTrends Setup dialog box.
2. Click the Log File button.

The Historic Log File Configuration dialog box appears.



3. Do one of the following:

- To save SuperTrend log files locally in the ioDisplay project directory, select Use Project Directory and click OK.

  SuperTrend log files will now be saved locally in the same directory as your ioDisplay project. The Use Project Directory setting will apply to each computer that is running this ioDisplay project, which means that each computer will save SuperTrend log files in its local project directory.

  NOTE: If you have selected a remote computer to save SuperTrend log files, the Use Project Directory option will not be available. You must click Browse and select a directory from the list of network directories that appears, even if the directory you select is the same as the project directory.

- To save SuperTrend log files in a location other than the ioDisplay project directory, follow these steps:

  i. Select Name and click Browse.

  If you have selected remote SuperTrend logging, the Choose Remote Logging Location dialog appears. Otherwise, the Select Directory dialog box appears.

  ii. Navigate in the file dialog box that appears until you find a computer and directory (Choose Remote Logging dialog box) or mapped local or network drive and directory (Select Directory dialog box) where you want to save SuperTrend log files.

  iii. Select the directory where the files will be saved and click OK.

  When your project is running, SuperTrend log files will be saved in the remote directory you specified. This setting will apply to each computer that is running this ioDisplay

project; each computer will save SuperTrend log files in the remote directory you selected.

Saving a Log in Text or Binary Format

When you configure the SuperTrend object, you can choose to save a SuperTrend historic log file in either text or binary format.

Text—When saved as an ASCII text file, a SuperTrend historic log file includes a header with information about tags assigned to the SuperTrend pens. One advantage of saving a log file in text format is accessibility—the file can be opened and viewed using any text editor, such as Windows Notepad. The disadvantage to this file format is that when a large amount of historical trend information is opened and graphed in a SuperTrend chart, the chart may be drawn slowly.

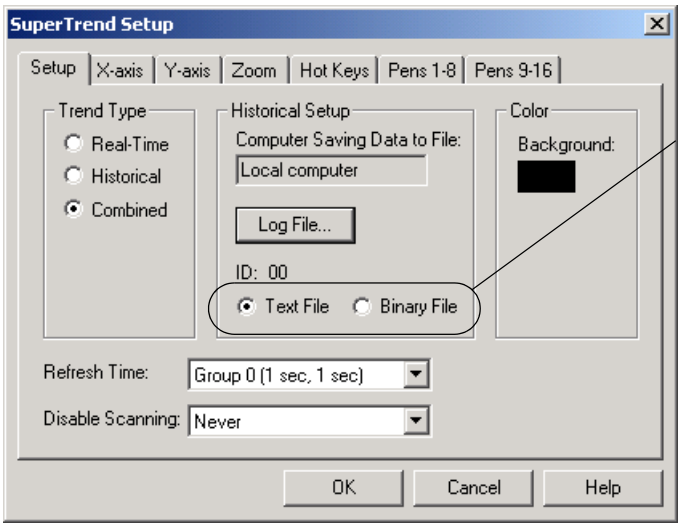
Binary—When a SuperTrend historic log file is saved in binary format, the file does not include a header containing information about tags assigned to the SuperTrend pens. A major advantage to saving a log file in binary format is speed—historical trend information from a binary file is usually graphed much more quickly than when using the same information from a text file. A disadvantage to binary file format is that the data cannot be opened and read using a text editor.

Changing Log File Formats

To change the file format of an existing SuperTrend historic log file between text and binary in ioDisplay Configurator, perform the steps listed below. (You will need to have previously run your ioDisplay project and collected historical data for log files to be present.)

1. Double-click a previously configured SuperTrend object that has generated one or more historic log files.

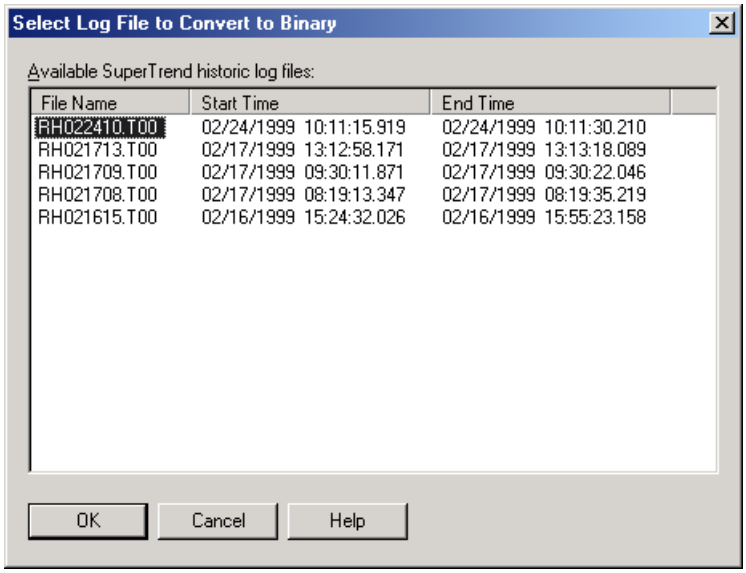
   The SuperTrend Setup dialog box appears with either Text File or Binary File selected, depending on how you originally configured the SuperTrend.



Options to choose the file format of a SuperTrend historic log file

2. If Text File is selected, choose Binary File; if Binary File is selected, choose Text File.

   The Select Log File to Convert to dialog appears. In the example below, text files that can be converted to binary format appear in the list:



3. Highlight a log file to convert and then click OK.

4. Now click OK to close the SuperTrend Setup dialog.

Viewing Binary Log Files

While you can't view the contents of a SuperTrend historic log file saved in binary format, a simple software utility (including source code) is included with ioDisplay that lets you convert binary log files to text format in order to view their contents. (The utility can also convert text files to binary format.) Once you have converted a binary SuperTrend historic log file to text format, you can easily view its contents using a text editor or other application.

NOTE: This utility creates a separate, new file in text (or binary) format that is not used by ioDisplay. *The format of the original SuperTrend historic log file is not changed. If you want to change the format in which this original log file is saved, see "Changing Log File Formats" on page 8-19.*

Important Guidelines for Using This Utility

- *Do not rename or modify any original SuperTrend historic log files.* These are files with the file extension .T*nn* (for example, .T00, .T02, etc.; the number depends on how many SuperTrends are saving historic data).
- If you rename the new files that are created by the utility, *do not use the name or file extension of the original SuperTrend historic log file.*

- *If you convert a text file to binary format, the header information in the text file will be removed.* This information cannot be recovered, even if you convert the binary file back to text format.

Converting a SuperTrend Log File for Viewing

To convert a SuperTrend historic log file, open a DOS window and at the command prompt enter:

```
STRNDCVT.EXE [log file]
```

where [log file] is the name of the SuperTrend historic log file to be converted. For example, to convert a log file named RH021615.T00, you would enter:

```
STRNDCVT.EXE RH021615.T00
```

The format change occurs automatically: If the log file is in binary format, it will be converted to text; the new text file has the extension .txt. If the file is in text format, it will be converted to binary; the new binary file has the file extension .bin.

Using XY Plots

You can use XY plots to graph real-time tag information. XY plots do not graph data over time, but instead plot points on a two-dimensional graph using data from two float or integer numeric tables in the ioControl strategy. (See Opto 22 form 1300, the *ioControl User's Guide,* for information on using numeric tables in a strategy.)

The value in each element of the numeric tables defines the coordinates for a single point on the x-axis and y-axis coordinate system. The example below shows how two numeric tables, each having three elements, would be used to draw three consecutive points on an XY plot.

Table Element	Numeric_Table_1	Numeric_Table_2	XY Coordinates
0	3	8	(3,8)
1	5.25	10	(5.25,10)
2	2	3.3	(2,3.3)

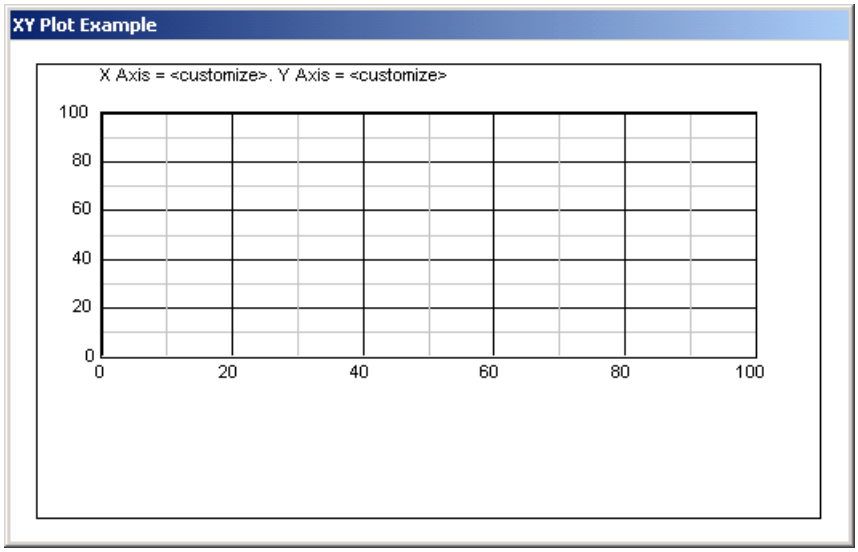
An XY plot is useful in applications where one value must be graphed against another value. Common examples in industrial settings include graphing temperature versus pressure, displacement versus input, or voltage versus current.

Creating an XY Plot

1. Select the XY Plot tool  from the toolbox and position the cursor where you want the XY plot object to begin in the window.

2. Click the mouse button, drag the mouse to the desired size, and then release the mouse button.

   The XY plot that appears should resemble the example below:



Y-Axis Layout

- Label Position = Bottom
- Max. Val = 100
- Min. Val = 0
- Major Divisions = 5
- Minor Divisions = 2

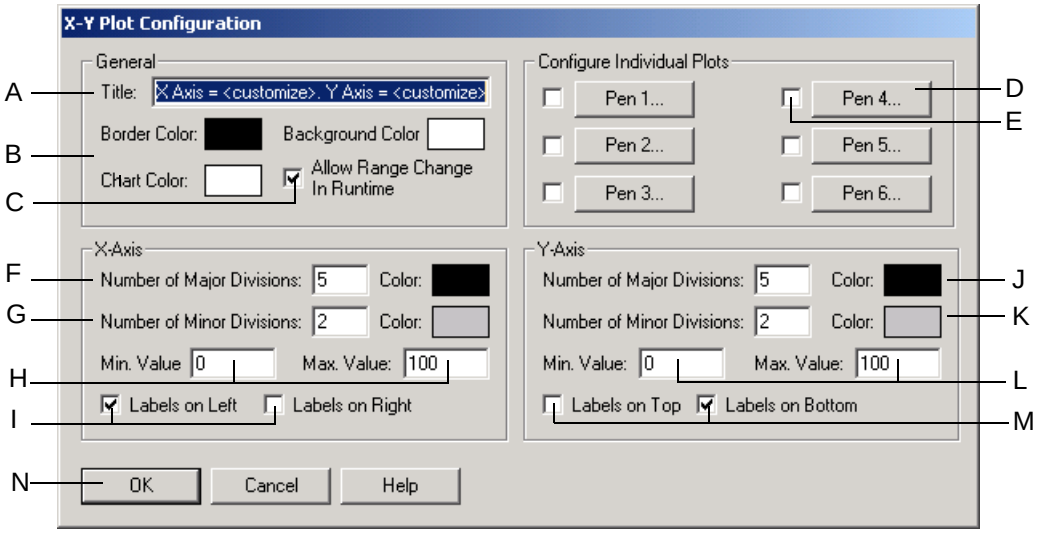
X-Axis Layout

- Label Position = Left
- Min. Val = 0
- Max. Val = 100
- Major Divisions = 5
- Minor Divisions = 2

Modifying an XY Plot

Choose the Select tool and double-click the XY plot.

The XY Plot Configuration dialog box appears:



A Enter the title of the XY plot here.

B Click a color button to enter the color for the chart, the chart's background, and the chart's border. In the Color dialog box that appears, choose a color and click OK.

C Select the Allow Range Change in Runtime checkbox to allow the user to change the x-axis and y-axis range values in ioDisplayioTerminalioMessageViewer.idbcontrol engine Runtime.

D Use the plot configuration buttons to define up to six individual plotted points in the XY plot. Each individual plot uses x-axis and y-axis coordinates from two ioControl tags. You can also define the plotted point's color and width, and specify line style and whether a legend should appear. See "Configuring Individual Plots in an Object" on page 8-23 for more information.

E Check this box to display a configured individual plot (uncheck the box to hide the plot).

F Enter the number of major x-axis divisions for the XY plot. This is the number of main sections the graph is divided into. You can also choose a color for the major divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

NOTE: For both x- and y-axis divisions, the lines dividing the major divisions appear thicker than the minor divisions.

G Enter the number of minor x-axis divisions for the XY plot. This will be the number of sections the major divisions are divided into. The minimum number of divisions is one.

You can also choose a color for the minor x-axis divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

H Enter the minimum and maximum values for the x-axis.

I Enter the label position for the x-axis. By default, the x-axis is labeled on the left, but you can also choose to have the label on the right or on both sides.

J Enter the number of major y-axis divisions for the XY plot. This is the number of main sections the graph is divided into. You can also choose a color for the major divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

NOTE: For both x- and y-axis divisions, the lines dividing the major divisions appear thicker than the minor divisions.

K Enter the number of minor y-axis divisions for the XY plot. This will be the number of sections the major divisions are divided into. The minimum number of divisions is one.

You can also choose a color for the minor y-axis divisions by clicking the Color field. In the Color dialog box that appears, choose a color and click OK.

L Enter the minimum and maximum values for the y-axis.

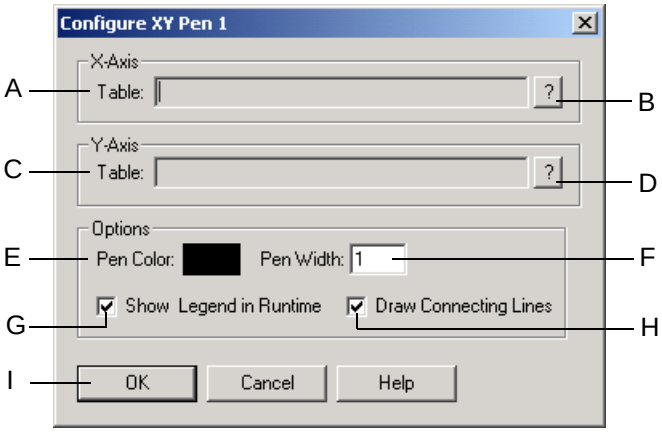
M Enter the label position for the y-axis. By default, the y-axis is labeled at the bottom, but you can also choose to have the label at the top, or at both top and bottom.

N Click OK to save your settings.

Configuring Individual Plots in an Object

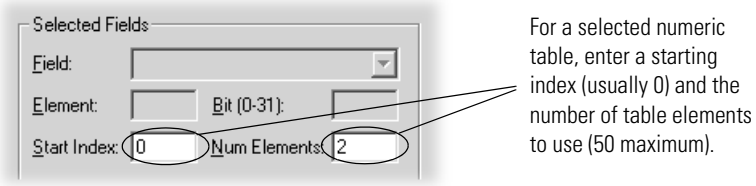
Click the configuration button for an individual plot in the XY Plot Configuration dialog box.

The Configure XY Plot dialog box appears:



A Enter an ioControl tagname here that will provide the x-axis values. Click B to open the Tag Selection dialog box.

B To enter a tagname in A, click here. The Tag Selection dialog box is displayed so you can select a tag for a numeric table. For the table you select, also enter the starting index (usually 0) and the number of table elements to be used in the plot. The fields for these values are shown in the illustration below. Up to 50 table elements can be used for an individual plot. To learn more about configuring tags in your project, see "Configuring Tags" on page 5-11.



C (Same as A above, but sets the tag used for the y-axis.)

D (Same as B above.)

E Click here to choose a pen color. In the Color dialog box that appears, choose a color and click OK.

F Enter the width of the line you want created by the pen. The width is specified in pixels.

G Select the Show Legend in Runtime checkbox to have the names of the tags used appear under the graph's x-axis.

H Select Draw Connecting lines to have lines connect each x-y coordinate.

I Click OK to save your settings.

CHAPTER 9

Configuring Trigger-Based Events

Introduction

In this chapter, you will learn how to make different types of events occur based on the value of a tag in an ioControl strategy. These events include saving a historic data log, launching an application, playing a sound, changing a window's state, downloading and uploading a recipe to the control engine, and starting an alarm.

In This Chapter

What's a Trigger-Based Event?

In ioDisplay, you can make things happen based on the value of a tag in an ioControl strategy. When the tag equals a specific value or falls within a defined range of values, it starts, or *triggers,* an event that you have specified.

You can use trigger-based events in many different ways. Here are a few examples:

- **Monitoring temperature**—Play a recorded warning sound to indicate an alarm condition if a tag exceeds a value.
- **Changing control system parameters for different products**—Download a new recipe (series of values) to a control engine when a tag reaches a set value.
- **Logging non-standard conditions or errors**—Print a log whenever monitored tags fall outside a previously defined range of values.

Historic Data Logs

A historic data log lets you collect and store data from your control process to a file on your computer, or on a remote computer on a network. Once it is saved in a file, historic data about your control process can be used by applications such as Microsoft Excel to generate reports, or the data can be archived for later reference.

NOTE: To ensure that the correct time and date information appear in a historic data log, make sure to set the time and date on your PC prior to starting ioDisplay.

By default, historic data logging begins when ioDisplay starts scanning the control engine for data, and ends when ioDisplay stops scanning the control engine for data. (Scanning starts when the ioDisplay project is opened in Runtime, and stops when the project is closed.) You can use a start or stop trigger, however, to start or stop historic data logging only when a tag, for example, is within a range of preset values, or a preconfigured number of samples have been taken. Start and stop triggers may be attached to any control engine variable.

Tag Types You Can Save to a Historic Log

You can record the following types of tags in a historic data log:

- Integers and integer tables
- Floats and float tables
- Strings
- Discretes.

With integer tables and float tables, you can select individual elements, groups of elements, or all elements in the table. See the *ioControl User's Guide* for more information on working with tables in an ioControl strategy.

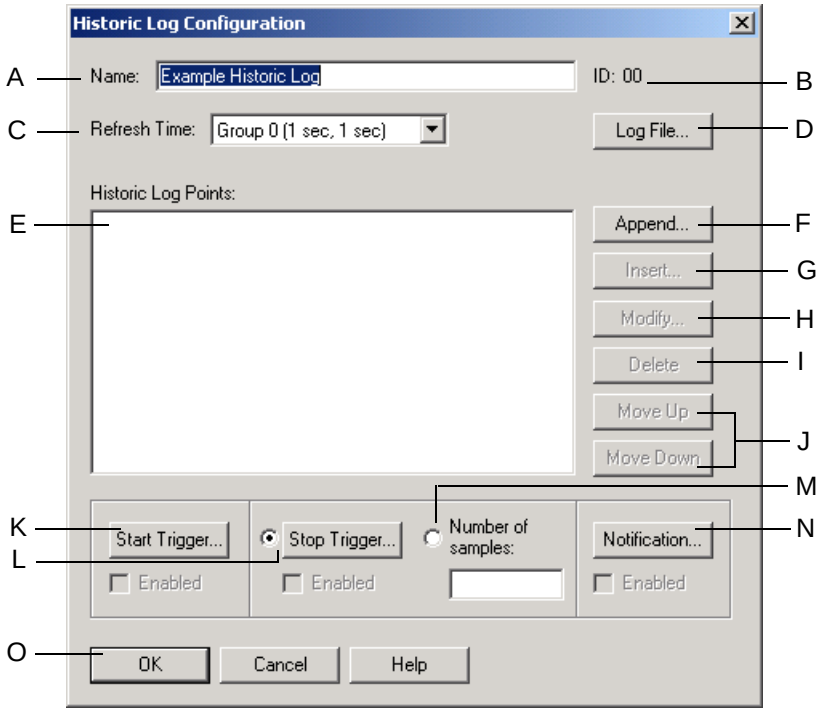
NOTE: To save data from more than one control engine, you must create a separate historic log file for each control engine that will be monitored. Data from multiple control engines cannot be saved in the same historic log file.

Configuring a Historic Data Log

To configure a historic data log in your ioDisplay project, choose Configure→Historic Data Log. In the Historic Logs dialog box that appears do the following:

- Click Add to create a new historic data log.
- Highlight an existing log and click Modify to change it.
- Highlight an existing log and click Delete to remove it.

If you are creating or changing a historic log file, the Historic Log Configuration dialog box will appear.



A Enter the name of the historic log here. Use this name to refer to the historic log group you're configuring. (This name does not affect the name of the historic log *file* that is saved to disk.)

NOTE: The name in this field must be different from all historic logs within the project or you will get an error message when you exit the dialog box.

B This field shows a unique numeric identifier for the historic log. This identifier, which starts out at 00 and increases sequentially, is used as the last two characters of the three-character file name extension for a historic log file. The first character of this file name extension is an uppercase "H", so with the identifier, a typical file name extension for a historic data log would be ".H00"

C Select a previously configured refresh time group to scan the historic log point tags. This scan rate applies to every log point configured within this historic log. For more information on setting up refresh time groups, see "Refresh Time Groups" on page 7-35.

D Click to specify the directory where the historic log file should be saved, the file name and format, and, if applicable, the rollover parameters for the file. See "Defining the Historic Data Log File" on page 9-4 for more information.

E This list shows all configured historic log points for this log file. The order of the points in the list (top to bottom) is the order in which the points will be logged. To change the order of the list, select a point, and then choose one of the Move buttons in J.

F Click to add a historic log point to the end of the historic log point list (E). See "Configuring a Historic Log Point" on page 9-7 for more information.

G Click to insert a historic log point above a highlighted historic log point.

H Lets you modify the highlighted historic log point.

I Deletes the highlighted historic log point from the list (E).

J Highlight the log point you want to move, and then click Move Up or Move Down to change the position of the point in the historic log point list.

K Click to configure the ioControl tag that will start scanning the listed historic log points. Check the Enabled box to make the start trigger active. See "Configuring a Start or Stop Trigger" on page 9-7 for more information.

Since the start trigger can be activated only from a non-triggered state, you must also configure a stop trigger at L or a number of scan times at M.

L Click to configure the ioControl tag that will stop scanning the listed historic log points. Check the Enabled box to make the stop trigger active. A stop trigger is required only when you have configured a start trigger. Additionally, a stop trigger is edge-sensitive and only activates on a false-to-true state transition. See page 9-7 for more information on setting up a stop trigger.

M Another way to stop scanning the historic log points is to set the number of samples, or times the points are scanned. Select Number of Samples, and then enter a discrete number of samples to take once the start trigger occurs.

N Click this button to assign a value to a tag when historic log sampling has stopped. Check the Enabled box to make notification active. See "Notification When a Trigger Has Stopped" on page 9-8 for more information.

O Click OK to save your settings and close the dialog box.

Now we'll look at the additional steps needed to complete the settings in the Historic Log Configuration dialog box.

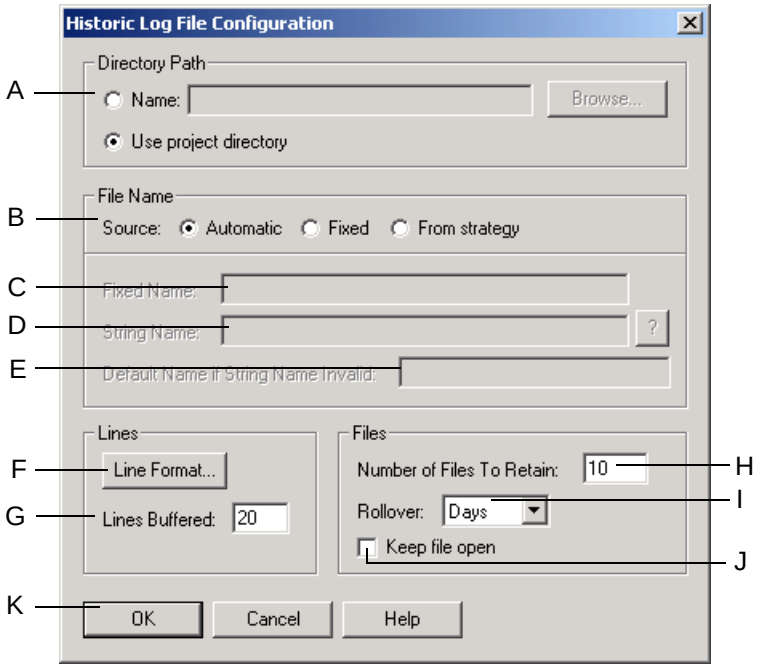
Defining the Historic Data Log File

You can name the historic data log file, determine where it will be located, configure how the data lines will appear, and define its rollover parameters in the Historic Log File Configuration dialog box.

NOTE: The historic data log file is not the same log file used to record data from a SuperTrend. A SuperTrend historic log file contains different data about log points, and is configured separately from a historic data log file. See "Using SuperTrend Log Files" on page 8-15 for more information. Alarm log files, which contain data about alarms that have been triggered, are also different from historic data log files. See "Alarm Logging Options" on page 9-42 for more information.

If you want to save data from more than one control engine, you should create a separate historic log file for each control engine that will be monitored. If data from multiple control engines is saved in the same historic log file, it is difficult to identify data that corresponds to a specific control engine.

To define a historic data log file, click the Log File button in the Historic Log Configuration dialog box and enter information in the Historic Log File Configuration dialog box that appears.



A Choose the directory where the historic log file will be saved. Click Name and enter the directory path in the field next to it, or click Browse to find a directory path. Click Use Project Directory to save the historic log file to the ioDisplay project directory. (This occurs by default if you don't specify a location.)

B Select Automatic, Fixed, or From strategy to determine how the historic log file name will be created, and then fill in additional information as needed for that option.

If the Automatic option is used, log files are named based on the rules described in "About Data Log File Names and Formats" on page 9-9. If you select this option, files are named using the rollover convention if required; this is described on page 9-10. If rollover is not used, the file is named "histlog.H*nn*," where *nn* is the two-digit historic log ID number. The Automatic option is used by default if you do not select another option.

C If you selected the Fixed option in B, enter a file name here. The file name can be any valid, eight-character DOS file name and doesn't require a three-character file extension. Note that if you don't specify an extension, one is *not* added automatically.

You must configure and enable a Start Trigger for this type of file (see page 9-16). When the trigger starts the historic data log, the new data is appended to the file if the file already exists. If the file doesn't already exist, it is created. The rollover naming convention doesn't apply to this type of file name.

D If you selected the From strategy string option, enter an ioControl string tagname here. Use the Tag Selection button ? to quickly enter the tag containing the name of the historic log file. You must configure and enable a Start Trigger for this type of file (see page 9-16). When the trigger starts the historic log, the string containing the file name is read, and the new data is appended to the log file if the file already exists. If the file

doesn't already exist, it is created. The rollover naming convention doesn't apply to this type of file name.

If the string name in D is an invalid file name, the default name of the log file is created using the following rules:

- If the string is empty, the project directory is added to E and the extension is an uppercase "H" followed by the historic log ID number.
- If the string is not empty and a project directory is not specified as the directory path, the Name in A is added to E.
- If the project directory is specified as the path, or the previous step failed, the project directory is added to E and the extension is an uppercase "H" followed by the historic log ID number. If the project directory is read-only or there is not enough room left on the drive containing the project directory, an error message indicates the file could not be created.

E Enter a default file name in E in case the file name in D is invalid. The file name can be any valid, eight-character DOS file name. The three-character file extension is assigned by ioDisplay, and will start with an uppercase "H" followed by the historic log's ID (for example: .H00 if the ID is 00).

F Click to configure the character, or delimiter, used to separate the data in the log file, to choose the type of quotes used for each data line, and where to insert carriage returns. You configure these parameters in the Line Format dialog box that appears. See page 9-9 for more information.

G Enter the number of lines of data your PC will save to a memory buffer before writing the information to the historic log file. When choosing a number, keep in mind that the lower the number of buffered data lines, the more frequently the computer has to write to the file. Alternately, the higher the number of data lines buffered in memory, the more data that will be lost if your PC loses power or has a system failure. A valid entry is any number between 0 and 999; the default is 20 lines.

H Enter the maximum number of historic log files that can be created using rollover before the oldest file is overwritten. For example, if you enter 10 and your rollover time period is set to hours, you will have 10 historic log files created for 10 hours of data before the oldest file is overwritten with new data. See page 9-10 for more information on rollover settings.

I Choose the rollover time period here. Select None to have all logged data placed in a single data file named HISTLOG.H*nn*, where *nn* is the two-character identifier assigned to the historic log. Logging begins when the Start Trigger is activated, and data collected will be appended to the existing data file. The size of the file is limited only by available disk space.

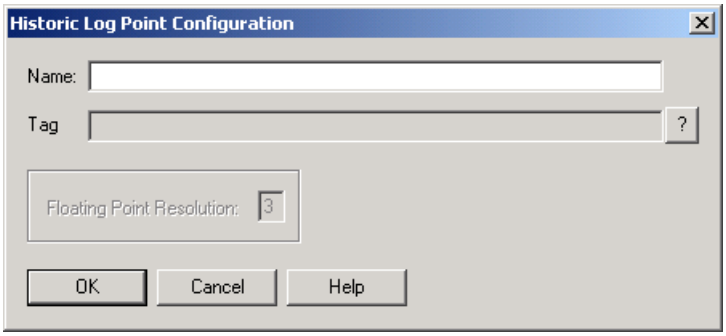
J Select Keep file open to leave the log file open to allow data to be appended to the historic log file more quickly. If you leave this box unchecked (the default setting), the file is closed after each time data is written to it. This provides greater data integrity than leaving the file open.

K Click OK to save your settings and close the dialog box.

Configuring a Historic Log Point

1. To add or change a historic log point in the Historic Log Configuration dialog box, click Append or Insert to add a point, or click Modify to change a point.

   The Historic Log Point Configuration dialog box appears:



2. (Optional) Enter a user-defined name for the historic log point.

3. Click the Tag Selection button ? and select an ioControl tag to be a historic log point. See "Configuring Tags" on page 5-11 for more information about selecting tags.

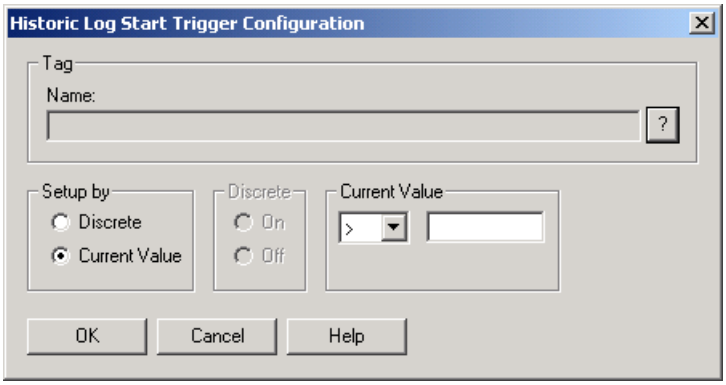
4. (Optional) If you picked a tag of type float, enter a number in the Floating Point Resolution box to specify how many places after the decimal point you want recorded.

5. Click OK to save your settings and close the dialog box.

Configuring a Start or Stop Trigger

1. To configure which tag and value will start or stop historic logging, click Start Trigger or Stop Trigger in the Historic Log Configuration dialog box.

   The Historic Log Configuration dialog box for either Start Triggers or Stop Triggers appears. These dialog boxes are identical. (The Historic Log Start Trigger Configuration dialog box is shown below.)



2. Click the Tag Selection button ? and select an ioControl tag to be a Start or Stop Trigger.

See "Configuring Tags" on page 5-11 for more information about selecting tags.

3. Select Discrete or Current Value in the Setup By group.

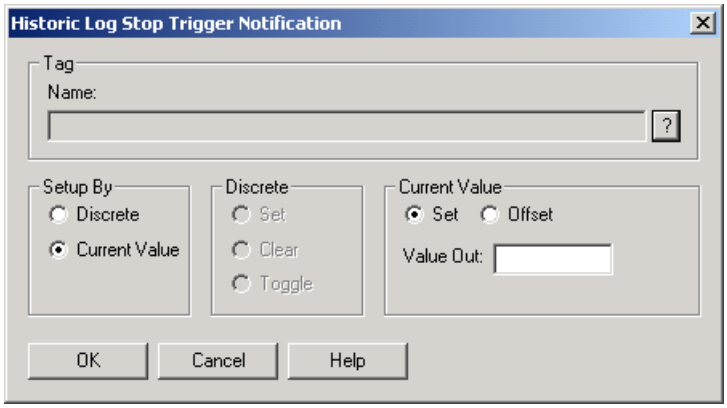
   Discrete makes the tag's on or off state trigger the historic log. Current Value sets the tag value that will trigger the historic log. To set a Current Value, select an operator in the drop-down menu, and then enter a value to compare the tag to.

Notification When a Trigger Has Stopped

When historic logging has stopped, you can set a tag to a given state or value. This tag setting will act as a flag to indicate data isn't being added to the log file anymore.

1. Click Notification in the Historic Log Configuration dialog box.

   The Historic Log Stop Trigger Notification dialog box appears:



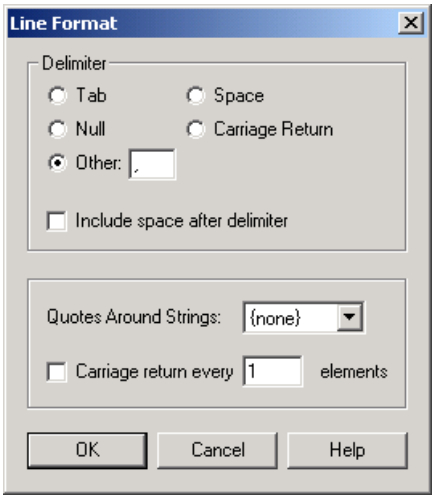
2. Click the Tag Selection button [?] to select an ioControl tagname. See "Configuring Tags" on page 5-11 for more about the Tag Selection dialog box.

3. Select Discrete or Current Value in the Setup By group.

   Discrete sets, clears, or toggles the tag's on or off state. Current Value sets the tag value. To set a Current Value, select Set or Offset in the Current Values group and then enter a tag value.

Setting Log File Line Format

1. To define how lines of data are stored in the historic log file, click Line Format in the Historic Log File Configuration dialog box. The Line Format dialog box appears:



2. In the Line Format dialog box, select from the choices in the Delimiter group a character (delimiter) that will separate data in the log file. To use a delimiter that's not listed, choose Other and enter the character you want to use. Select Include space after delimiter to put a space after each delimiter.

3. From the choices in the drop-down list "Quotes Around Strings," select the quotes that will appear around each string in the log file.

4. To insert a carriage return after a certain number of data elements, select the check box "Carriage return every" and enter the number of data elements.

   The maximum number of elements that can be entered is 99,999. This option is intended for historic logs with very long data lines which are read by programs that cannot handle long data lines. The date and time information at the beginning of a data line are not counted as data elements. See "About Data Log File Names and Formats" below for more information about data elements in log files.

About Data Log File Names and Formats

Naming Log Files

There are three ways you can name a historic log file: by using a fixed name, by building the name using a string tag, or by letting the file name be created automatically. If you let ioDisplay create the file name automatically, the file name will be histlog.H##, where ## is the two-character identifier assigned to the historic log by the Configurator. The .H## is the common extension that identifies all historic log files.

Naming Files Using Rollover

File rollover is used to divide historic log information among several data files. You can configure how often a new data file is created; after the time limit has expired, another historic log file is created and data is rolled into the new file.

Historic log files with names determined automatically by ioDisplay use the rollover format (see "Naming Log Files" on page 9-9). The rollover format does not apply to files with fixed names or file names constructed from ioControl strategy string tagnames.

Historic log files using file rollover follow this naming pattern:

- Months: RMyymm.H##
- Days: RDyymmdd.H##
- Hours: RHmmddhh.H##

where yy= year, mm= month, dd= day, hh= hour, and ## represents the two-character historic log ID number.

The rollover time period for historic log files can be based on months, days, or hours.

Hours If hours are selected as the rollover period, a new data file is created at the top of every hour. For example, if data logging were triggered at 8:30 a.m., the first data file would contain data from 8:30 a.m. to 9:00 a.m. Thereafter, data files will contain one hour's worth of data for every hour thereafter, 9:00 a.m. to 10:00 a.m., 10:00 a.m. to 11:00 a.m., etc.

Days If days are selected as the rollover period, a new data file is created every day at midnight. For example, if data logging were triggered at 7:00 p.m. on the 5th, the first data file would contain data from 7:00 p.m. to 12:00 a.m. on the 5th. Thereafter, data files will contain data from midnight the 5th to midnight the 6th, midnight the 6th to midnight the 7th, etc.

Months If months are selected as the rollover period, a new data file is created on the first day of every month at midnight. For example, if data logging started on January 27th, the first data file would contain data from the 27th of January to the 31st of January. Thereafter, data files will contain data from the 1st of February to the end of February, the 1st of March to the 31st of March, etc.

Data Log Elements

Historic data logs consist of header lines and data lines.

Header Line The first line of the file, a header line shows the name of each data field. Lines of data samples then follow the header line.

Here's an example:

```
Date,Time,CNTR1:Float.TEMP208,CNTR1:Float.PRES209,CNTR1:Float.LEVEL218
```

Date and **Time** show where timestamp information will appear in the data lines.

CNTR1:TEMP208, CNTR1:PRES209, and **CNTR1:LEVEL218** show that information will be recorded for three ioControl tags: TEMP208, PRES209, and LEVEL218. These tags are all on the control engine named CNTR1.

Data Lines Data lines follow a header line, and have the following format:

Date<delimiter>**Time**<delimiter>**TAG1**<delimiter>**TAG2**. . .**TAG1000**<crlf>

Date is the current system date with the format: YYYY/MM/DD, where YYYY=year, MM=month, DD=day.

Time is the current system time with the format: hh:mm:ss, where hh=hour, mm=minute, ss=seconds.

TAG1...TAG1000 are valid ioControl tags with the format: control engine_Name:Item_Type.Tag.

<delimiter> is any printable ASCII character.

<crlf> is a carriage return, line feed.

Here's an example of what the file may look like:

```
Date,Time,Cookie:Float.TEMP208,Cookie:Float.PRES209,Cookie:Float.LEVEL218
1996/03/01,17:00:00,120.02,14.96,12.09
1996/03/01,18:00:00,120.06,14.98,12.03
1996/03/01,19:00:00,120.03,14.99,12.02
1996/03/01, 20:00:00,120.04,15.01,12.05
```

In this sample file, data is being sampled every hour from a control engine named "Cookie," and a temperature (TEMP208), pressure (PRES209), and tank level (LEVEL218) are being recorded.

Launching Applications

You can use ioDisplay to start, or launch, other applications in two ways:

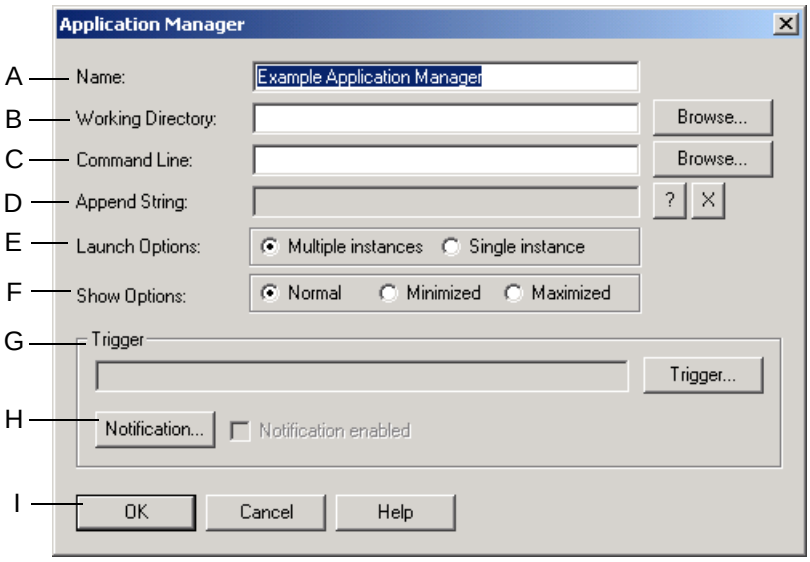
- By configuring a dynamic attribute for a graphic, and then selecting the graphic during Runtime. See Chapter 7, "Using Animated Graphics" to learn how to do this in ioDisplay Configurator.
- By configuring an application manager to associate a tag with an application, and then launching the application using triggers. This is the method of starting an application we will cover in this section.

Configuring an Application Launch

To use ioDisplay Configurator to configure an application launch using a trigger, choose Configure→Applications, and in the Application Managers dialog box that appears do the following:

- To create a new application manager, click Add. You can configure up to 1,000 application managers per project.
- To change an existing application manager, highlight it and click Modify.
- To remove an existing application manager, highlight it and click Delete.

If you are creating or changing an application manager, the Application Manager dialog box will open:



A Enter the name of the application manager here. This name refers to this application launch setup, and must be different from all application managers within the project.

B Enter the path name of the working directory to use after launching the application. If a working directory isn't specified, the current project directory is used. Click Browse to quickly choose a directory path in the Working Directory Selection dialog box (see page 9-13).

C Enter the path and file name of the program you want to run when the trigger condition occurs. Click Browse to quickly choose the path and file name in the standard Windows file selection dialog box.

D (Optional) Select a string tag from the ioControl strategy to be appended to the Command Line string in C. If the appended string is a command line option, a space must be included in the Command Line string to separate it from the main command line

Use the Tag Selection button ? to select the string tagname. See"Configuring Tags" on page 5-11 for more information on tags. Click the Clear button × to quickly remove an entry from D.

E Select Single instance to have ioDisplay Runtime check whether the trigger has already launched a session, or instance, of an application. If the trigger hasn't already launched the application, it will be started. Select Multiple instances to let the trigger start any number of sessions of the same application.

The Single Instance option doesn't prevent an application from being launched by other graphics and triggers, so multiple instances of an application can still occur. For

example, if a trigger launches a Microsoft Word session, it can't launch any other application until this Word session ends. A toggled graphic or another trigger, however, could launch another session of Word, so two instances of the same application would be running concurrently.

CAUTION: *Running multiple instances of the same application on your PC is not recommended. Just like running several different applications at the same time, running multiple instances of the same application requires additional memory and other system resources. This may slow your PC's performance, as well as that of* ioDisplay *Runtime.*

F Select how the application's window will appear on-screen: Normal, Minimized, or Maximized.

G Click Trigger to select the ioControl tag used to trigger the application launch. The trigger is edge-sensitive and only activates from a non-triggered state. See "Selecting a Trigger to Launch an Application" on page 9-14 for more information.

H Click Notification to assign a value to a tag when an application is successfully launched. See "Notification When an Application Has Been Launched" on page 9-15 for more information. A check mark in the Notification enabled box indicates a notification tag is configured.

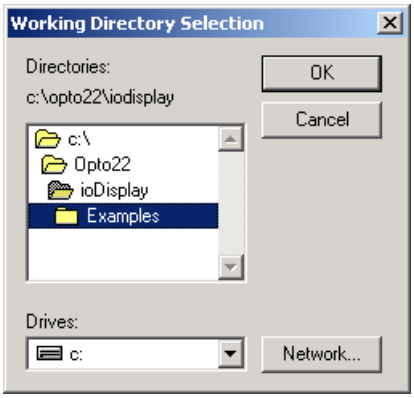
I Click OK to save your settings and close the dialog box.

Now we'll look at the additional steps needed to complete the settings in the Application Manager dialog box.

Selecting a Working Directory for a Launched Application

1. To set up the working directory the launched application should use, click Browse in the Application Manager dialog box.

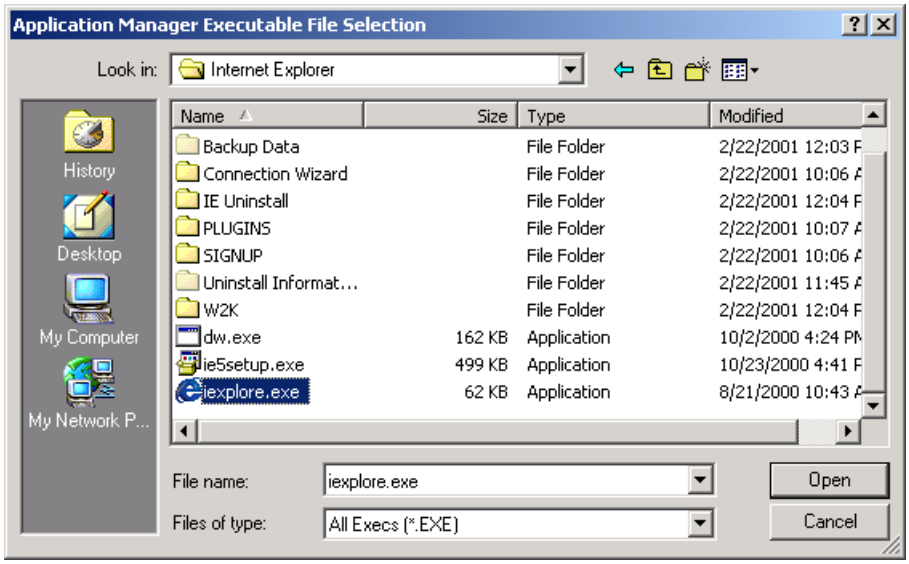
2. In the Working Directory Selection dialog box that appears, navigate to the working directory path and click OK. (Click Network to select a network drive.)



Selecting the Application File to Run

1. To choose the application you want to run, in the Application Manager dialog box click the Browse button next to the Command Line field.

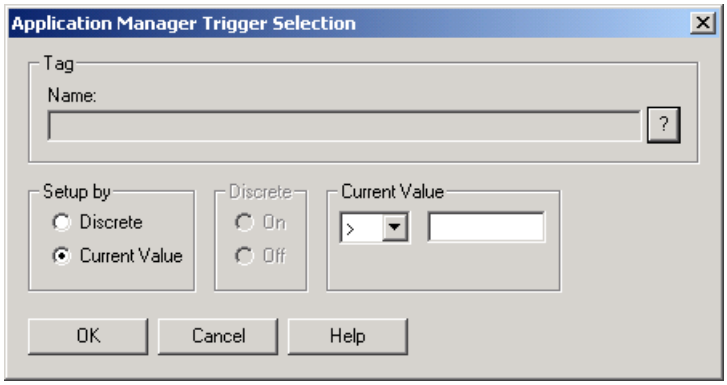
2. In the Application Manager Executable File Selection dialog box that appears, navigate to the application you'd like to run, highlight it, and click OK.



Selecting a Trigger to Launch an Application

1. To select the trigger that will launch an application, click Trigger in the Application Manager dialog box.

   The Application Manager Trigger Selection dialog box appears:



2. Enter the name of the trigger in the Name field. Use the Tag Selection button ? to quickly choose a tag from the Tag Selection dialog box.

3. Select Discrete or Current value in the Setup by group.

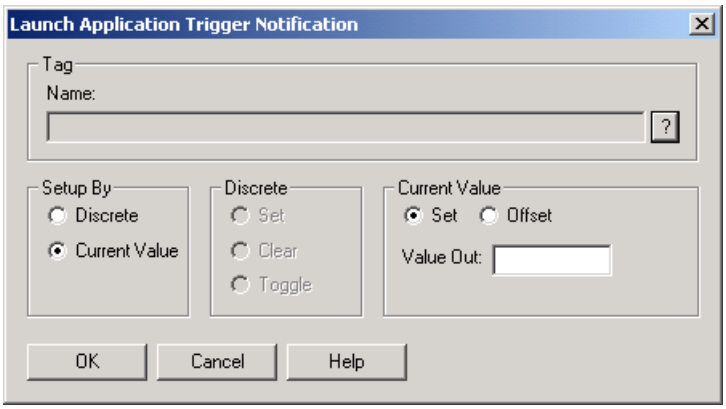
Discrete makes the tag's on or off state trigger the application. Current Value sets the tag value that will trigger the application. To enter a value in the Current Value field, select an operator in the drop-down menu, and then enter a value to compare the tag to.

Notification When an Application Has Been Launched

You can set a tag to a given state or value when an application has been successfully launched by a trigger.

1. Click Notification in the Application Manager dialog box.

   The Launch Application Trigger Notification dialog box appears:



2. Click the Tag Selection button ? to quickly select an ioControl tag as the flag to indicate the application was launched successfully. See "Configuring Tags" on page 5-11 for more information about configuring tags.

3. Select Set or Offset, and then enter the value that will be sent to the tag in the Value Out field. Choose Set to replace the tag's current value with the number in the Value Out field. Choose Offset to add that number to the tag's current value.

Sounds

Sounds can provide important feedback, such as alerts or warnings, for the operator using your ioDisplay project. You can add sounds to your ioDisplay project by configuring triggers to start and stop standard Windows sound files. To use this capability, the PC running the project must have a properly configured sound card and corresponding system software, as well as a set of speakers. You can use both digitized sound (.WAV) and MIDI music (.MID) files in your project.

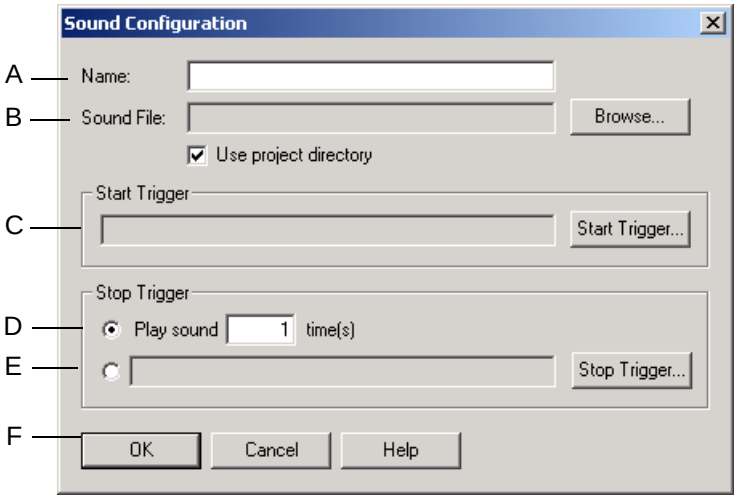
Sounds can also be used with project alarms. If both an event and an alarm occur, causing multiple sounds to play, the sounds will alternate. See page 9-45 for more information on setting up sounds to work with alarms.

Configuring a Sound

To configure a sound, choose Configure→Sounds, and in the Sounds dialog box that appears:

- Click Add to add a new sound event. You can configure up to 1,000 sound events per project.
- Highlight an existing sound event and click Modify to change it.
- Highlight an existing sound event and click Delete to remove it.

If you are creating or changing a sound event, the Sound Configuration dialog box will appear:



A Enter a name for the sound event. No two sound events in a project may have the same name.

B Click Browse and select the sound file you want to play in the standard Windows file selection dialog box that appears. If the Use project directory option is checked, you can only choose sound files located in your project directory.

C Click Start Trigger to configure a trigger to start playing the sound. See "Configuring Start and Stop Triggers for Sounds" below to learn how to configure this trigger.

D Select the Play sound option if you want the sound to play a specific number of times before stopping. Enter the number of times the sound will play in the field next to the option. The default value is one.

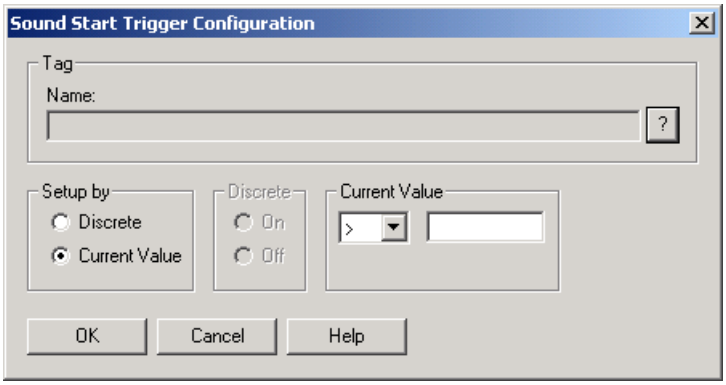
E Click Stop Trigger to configure a trigger to stop the sound that is playing. See "Configuring Start and Stop Triggers for Sounds" below to learn how to configure this trigger.

F Click OK to save your settings and close the dialog box.

Configuring Start and Stop Triggers for Sounds

1. To configure the tag and value that will start or stop a sound, click Start Trigger or Stop Trigger in the Sound Configuration dialog box

The Sound Trigger Configuration dialog box for either Start Triggers or Stop Triggers appears. These dialog boxes are identical. (The Sound Start Trigger Configuration dialog box is shown below.)



2. Use the Tag Selection button ? to quickly choose a tag from the Tag Selection dialog box.

3. Select Discrete or Current Value in the Setup by group. Current value sets the tag value that will trigger the sound. Select an operator in the drop-down menu, and then enter a value to compare the tag to. Discrete makes the tag's on or off state trigger the sound.

Window States

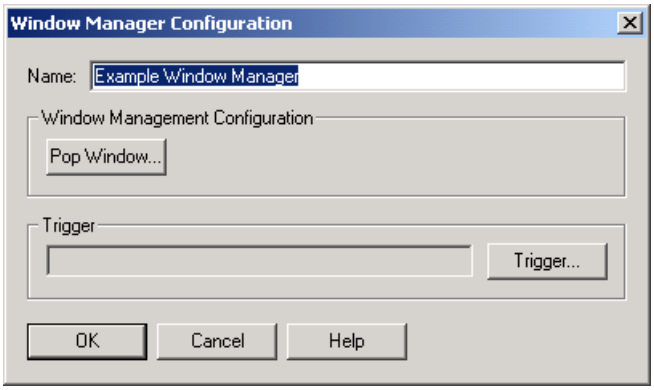
You can use a trigger to modify the appearance of windows that appear in Runtime. Changing the appearance of windows can be effective when you want to immediately attract an operator's attention, or to prompt operators to take the next action.

Configuring Trigger-Based Window States

To configure a window state, choose Configure→Window States, and in the Window Managers dialog box that appears:

- Click Add to add a new window manager. You can configure up to 1,000 window managers per project.
- Highlight an existing window manager and click Modify to change it.
- Highlight an existing window manager and click Delete to remove it.

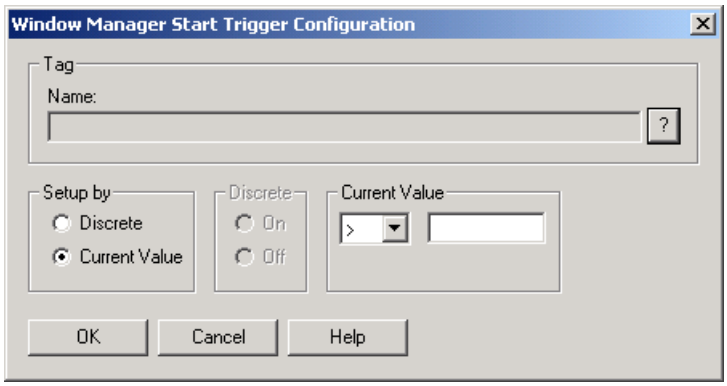
If you are creating or changing a window manager, the Window Manager Configuration dialog box will appear:



To set up a window manager, do the following:

1. Enter the name of the window manager in the Name field.
2. Click Pop Window to configure window states for one or more windows. See "Windows" on page 7-31 for more information about changing window states.
3. Click Trigger to configure the trigger that will change the window state.

   The Window Manager Start Trigger Configuration dialog box appears:



4. Use the Tag Selection button ? to quickly choose a tag from the Tag Selection dialog box.
5. Select Discrete or Current Value in the Setup By group. Current Value sets the tag value that will trigger the change in window state. Select an operator in the drop-down menu, and then enter a value to compare the tag to. Discrete makes the tag's on or off state trigger the change in window state.

Recipes

Recipes are ASCII text files used to download data to a control engine and to upload data back to a PC running ioControl. Recipes are very useful for batch processes where system variables are pre-determined and vary between runs or product types. You can also use recipes to save critical process settings and then create more recipes, or to restore a system after a failure.

Recipe *download* files are used to send process data and chart control instructions to a control engine (see "Recipe Download File" on page 9-20). Recipe *upload* files record values currently being used in table elements on a particular table (see "Recipe Upload File" on page 9-22). To create either a download or upload file, first see "Basic Recipe File Format" below.

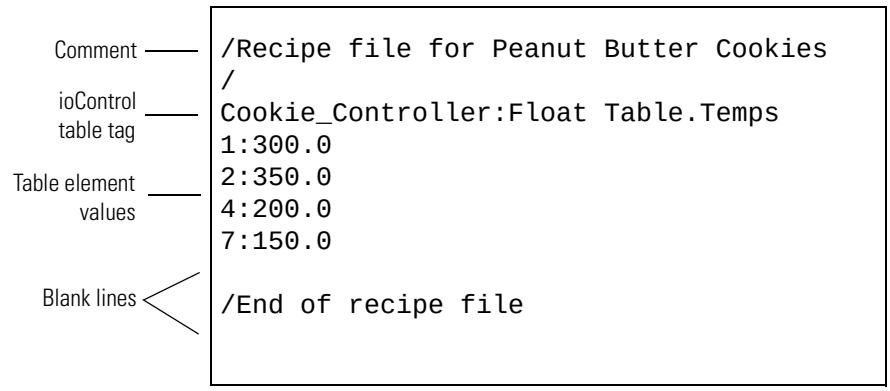
This section contains the following topics:

Basic Recipe File Format

Download and upload recipe files both use the basic recipe format described here. To create a recipe file, first read this section and then see either "Recipe Download File" on page 9-20 or "Recipe Upload File" on page 9-22.

A recipe text file contains one or more ioControl integer, float, or string table tags and values. You can create or modify a recipe file using any text editor or word processor that can save files in ASCII format.

As shown in the example, each recipe file contains at least one ioControl table tag followed by data values and blank lines. Recipes may also contain comments, and a download recipe might include chart control instructions as well.



Comment Line—Any line that starts with a / (forward slash) is a comment, and is ignored by ioDisplay. Use comments to explain the recipe and to make notes.

ioControl Tag—Identifies the table tag in the ioControl strategy that the recipe uses. Only integer, float, and string tags are valid tag types. The tag is in the following format:

<Control Engine Name>:<Table Type>.<Table Name>

- **<Control Engine Name>** is the Opto 22 control engine name.
- **<Table Type>** includes one of the following keywords: Integer Table, Float Table, or String Table. They identify the variable type and are separated from <Control Engine Name> by a colon (:).
- **<Table Name>** is the ioControl table tagname. It must be of the type specified by <Table Type>. A period (.) separates it from <Table Type>.

Element values—These are the values downloaded to the control engine or uploaded to the PC.

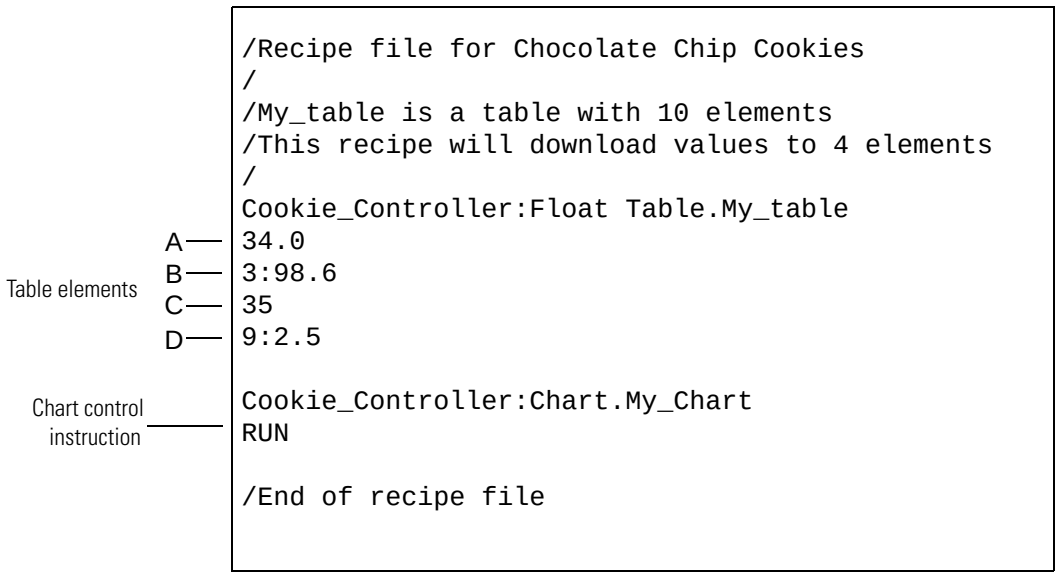
Blank line—This is a required element of a recipe file. Make sure to create a blank line at the end of a data list and at the end of the file by entering two carriage returns.

Recipe Download File

This section describes how to create a recipe download file. Before you begin, see "Basic Recipe File Format" on page 9-19. For a description of how to configure ioDisplay for a recipe download, see "Configuring a Recipe Download" on page 9-23

A download file provides an efficient method for making changes to program variables without having to manually enter the data. For example, let's say you have a cookie factory that makes chocolate chip cookies in the morning and peanut butter cookies in the afternoon. After making the chocolate chip cookies you can download a recipe to the control engine that contains new values for the peanut butter cookies.

This example has four table elements and one chart control instruction:




```
/Recipe file for Chocolate Chip Cookies
/
/My_table is a table with 10 elements
/This recipe will download values to 4 elements
/
Cookie_Controller:Float Table.My_table
34.0
3:98.6
35
9:2.5

Cookie_Controller:Chart.My_Chart
RUN

/End of recipe file
```

The **table elements** in the example are used as follows:

A No index reference is indicated for the first data value, 34.0. Therefore, by default it is assigned to zero or My_table[0].

B The next line, 3:98.6, has an index reference of 3. This means the value 98.6 would be sent to My_table[3], the fourth element of the table.

C The next value, 35, would be sent to My_table[4].

D The last data line, 9:2.5, would send 2.5 to the tenth element, My_table[9]. Make sure to leave the next line blank by entering two carriage returns after the last data line. The blank line indicates that all the data for that particular table has been specified. Do not put blank lines between lines that contain data for the table.

Chart control instructions such as RUN, STOP, SUSPEND, and CONTINUE control the execution state of one or more ioControl charts when a recipe file is downloaded. You can use a chart control instruction to start a chart that can then move downloaded table values to other program variables.

Chart control instructions have the following format:

<Control Engine Name>:Chart.<Chart Name>
<Chart State>

- **<Control Engine Name>** is the name of the Opto 22 control engine.
- **Chart** identifies the keyword identifying this line as a chart instruction. "Chart" is separated from <Control Engine Name> by a colon (:).
- **<Chart Name>** is the ioControl strategy chart name, and is separated from Chart by a period (.).
- **<Chart State>** is the chart instruction, and must include one of the following keywords: STOP, RUN, SUSPEND, or CONTINUE. This instruction must be on the line following the

<Chart Name>. The last line in a chart control instruction must be a blank line and contain only a carriage return.

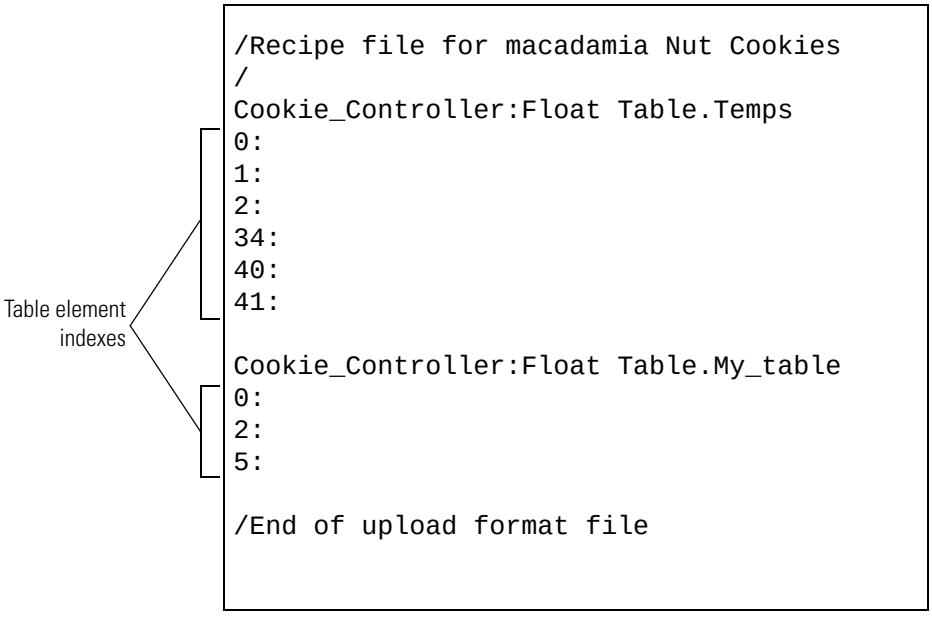
Recipe Upload File

This section describes how to create a *format file* for a recipe upload. Before you begin, see "Basic Recipe File Format" on page 9-19. For a description of how to configure ioDisplay for a recipe upload, see "Configuring a Recipe Upload" on page 9-25

A recipe upload records the values currently being used in table elements on a particular table. You can use these values to improve the production process. For example, using the cookie factory analogy, if after downloading the peanut butter cookie recipe and starting the cooking process, you then decide to add more sugar or cook the cookies longer, you can modify one or more of the table values (either by an ioDisplay SendValue, or from ioControl debugger) and then upload the new recipe. The next time you make peanut butter cookies, you can re-download the new recipe.

An upload format file tells ioDisplay which table elements to upload to a *destination file* (or *results file*) on the PC. The format file has the basic recipe file format, but does not usually contain data.

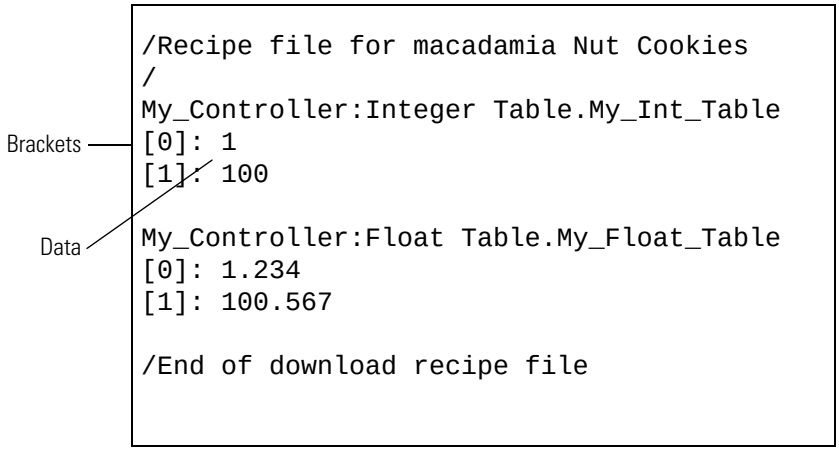
For a format file to be valid, there must be at least one table element index (e.g., 2:), and there must be an index for every table element to be uploaded, even if starting at element zero.



As shown in the example, you can include table element indexes for multiple tables. Remember to put a blank line before each new section of data and at the end of the file by entering *two* carriage returns at the end of the line.

Re-using a Destination File

After an upload, the destination file (or results file) contains data and has brackets around each table element index. If you re-use a destination file as a format file, the brackets and the data are ignored by the control engine.



You can also use the destination (or results) file as a recipe download file.

Activating Recipe Downloads and Uploads

You can configure a recipe to download or upload to a control engine in two ways:

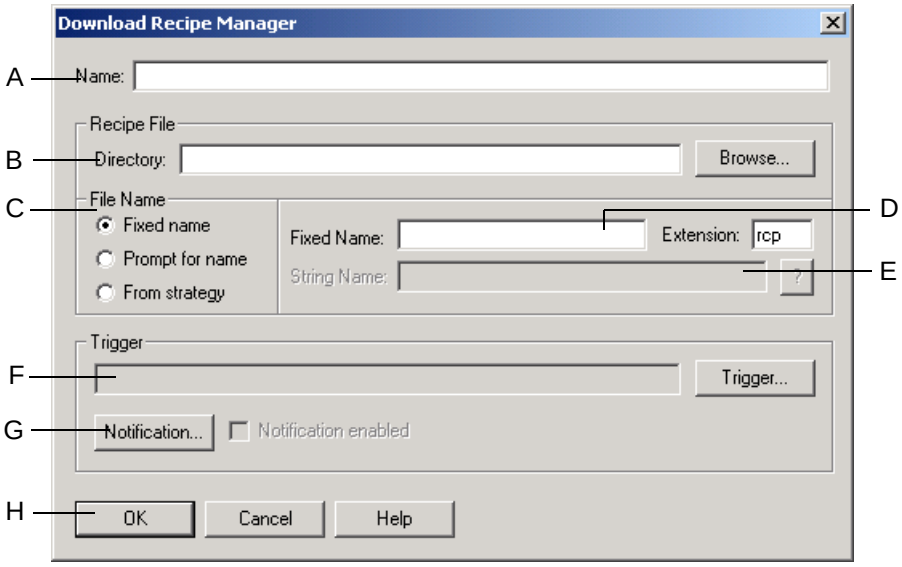
- **Animated Graphic**—By configuring a dynamic attribute for a graphic—when the operator selects the graphic, the recipe action will occur. See Chapter 7, "Using Animated Graphics" to learn how to do this.
- **Trigger**—By configuring a trigger that associates a tag with a recipe action—when the tag value meets a defined value, the recipe action will be triggered. We will cover this method of starting a recipe action in the next two sections, "Configuring a Recipe Download" and "Configuring a Recipe Upload."

Configuring a Recipe Download

To configure a recipe to download using a trigger (or "download recipe manager"), choose Configure→Recipes, and in the Download Recipes section of the Recipe Managers dialog box that appears, do the following:

- To create a new download recipe manager, click Add.
- To change an existing download recipe manager, highlight it and click Modify.
- To remove an existing download recipe manager, highlight it and click Delete.

If you are creating or changing a download recipe manager, the Download Recipe Manager dialog box appears:



A Enter the name of the download recipe manager. This name is used to refer to the recipe group you're configuring in the Configurator. The name in this field must be different from all recipe managers in this project.

B Choose the directory the recipe file resides in. You can type the directory in the edit box or click Browse to quickly find and enter the path.

C Choose the source of the recipe file name.

- If you choose Fixed name, D is highlighted.
- If you choose Prompt for name, the operator will be asked for the name of the recipe file to be downloaded.
- If From strategy is selected, E is highlighted.

D If Fixed name was selected in C, enter the name of the recipe file located in directory B. Notice the file extension is .rcp.

E If From strategy was selected in C, click the Tag Selection button ? to enter a tagname of type string that contains the recipe file name. See "Configuring Tags" on page 5-11 for more information about using tags.

F Click Trigger to select an ioControl tagname that will trigger the download recipe action. The trigger can be activated only from a non-triggered state. See "Selecting a Trigger to Start the Recipe Upload/Download" on page 9-27 to learn how to configure this trigger.

G Click Notification to assign a value to a tag when a recipe has successfully downloaded. Check the Enabled box to make notification active. See "Notification When Recipe Has Been Downloaded/Uploaded" on page 9-27 for more information.

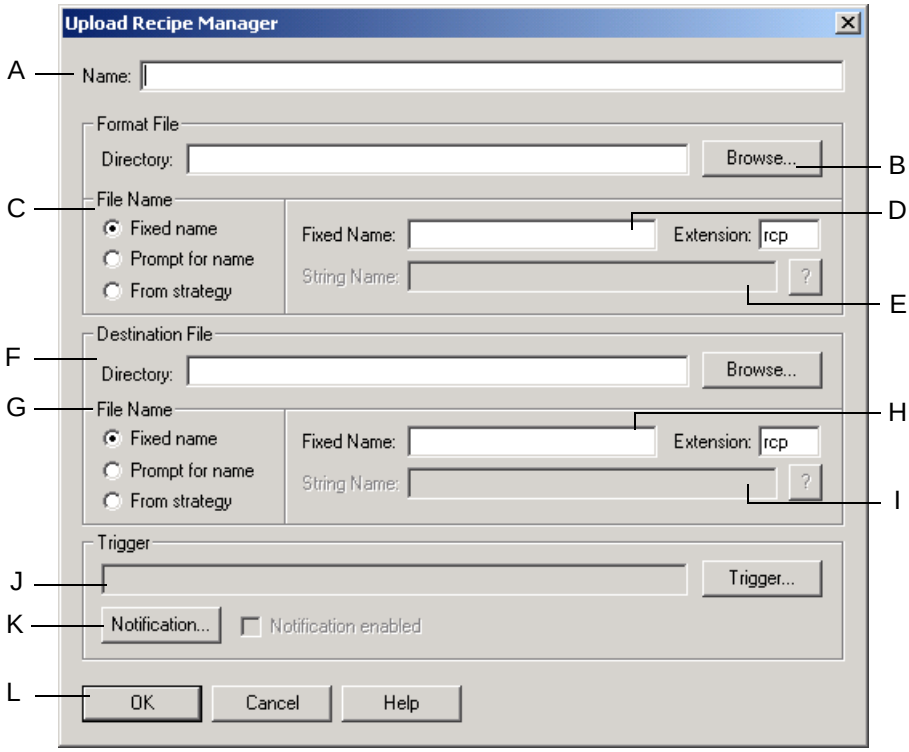
H Click OK to save your settings and close the dialog box.

Configuring a Recipe Upload

To configure a recipe to upload using a trigger (or "upload recipe manager"), choose Configure→Recipes, and in the Upload Recipes section of the Recipe Managers dialog box that appears, do the following:

- To create a new upload recipe manager, click Add.
- To change an existing upload recipe manager, highlight it and click Modify.
- To remove an existing upload recipe manager, highlight it and click Delete.

If you are creating or changing an upload recipe manager, the Upload Recipe Manager dialog box appears:



A Enter the name of the upload recipe manager. This name is used to refer to the recipe group you're configuring in the Configurator. The name in this field must be different from all recipe managers in this project.

B Choose the directory the recipe file resides in. You can type the directory in the edit box or click Browse to quickly find and enter the path.

C Choose the source of the recipe file name.

- If you choose Fixed name, D is highlighted.
- If you choose Prompt for name, the operator will be asked for the name of the recipe file to be uploaded.
- If From strategy is selected, E is highlighted.

D If Fixed name was selected in C, enter the name of the recipe file located in directory B. Notice the file extension is .rcp.

E If From strategy was selected in C, click the Tag Selection button ? to enter a tagname of type string that contains the recipe file name. See "Configuring Tags" on page 5-11 for more information about using tags.

F Click Browse and choose the directory location of the recipe file that will receive the information.

G Choose the source of the recipe file name.

- If you choose Fixed name, D is highlighted.
- If you choose Prompt for name, the operator will be asked for the name of the recipe file to be uploaded to.
- If From strategy string is selected, E is highlighted.

H If Fixed name was selected in C, enter the name of the recipe file located in directory B.

I If From strategy was selected in C, click the Tag Selection button ? to enter a tagname of type string that contains the recipe file name. See "Configuring Tags" on page 5-11 for more information about using tags.

J Click Trigger to select an ioControl tagname that will trigger the upload recipe action. The trigger can be activated only from a non-triggered state. See "Selecting a Trigger to Start the Recipe Upload/Download" on page 9-27 to learn how to configure this trigger.

K Click Notification to assign a value to a tag when a recipe has successfully uploaded. Check the Enabled box to make notification active. See "Notification When Recipe Has Been Downloaded/Uploaded" on page 9-27 for more information.

L Click OK to save your settings and close the dialog box.

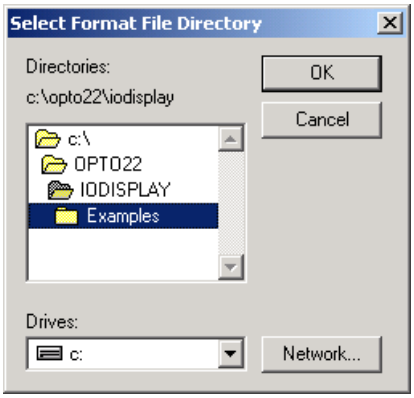
Now we'll look at the additional steps needed to complete the settings in the Download/Upload Recipe Manager dialog boxes.

Selecting a Download/Upload Recipe File Directory

1. To set up the directory that the recipe will be uploaded to or downloaded from, click Browse in the Download/Upload Recipe Manager dialog box.

   You will need to do this when you choose the directory for the download recipe file, the upload format file, or the destination upload file. Since all three options show very similar dialog boxes, we will only discuss one of them.

2. In the Upload Recipe Manager dialog box, click Browse.

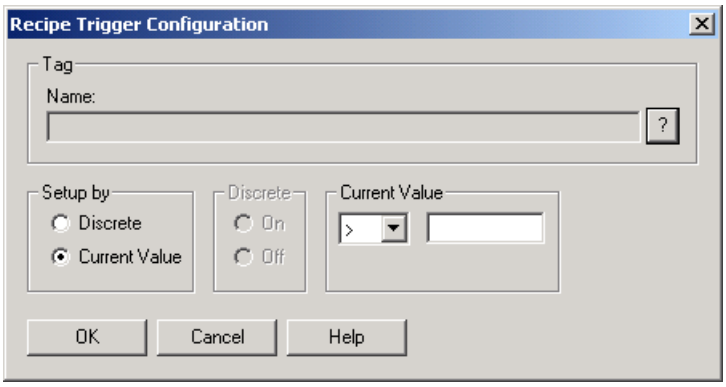


3. In the Select Format File Directory dialog box that opens, navigate to the working directory path and click OK. (Click Network to select a network drive.)

Selecting a Trigger to Start the Recipe Upload/Download

1. To select the trigger that will start a recipe upload or download, click Trigger in the Download/Upload Recipe Manager dialog box.

   The Recipe Trigger Configuration dialog box appears:



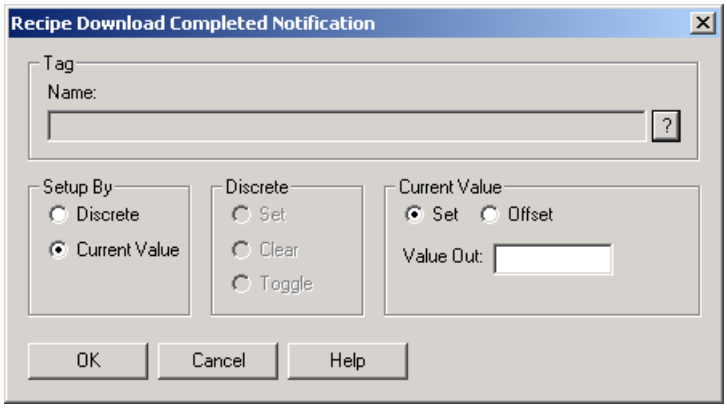
2. Enter the name of the trigger in the Name field. Use the Tag Selection button ? to quickly choose a tag from the Tag Selection dialog box. See "Configuring Tags" on page 5-11 for more information about tags.

3. Select Discrete or Current value in the Setup by group. Current value sets the tag value that will trigger the recipe download/upload. Select an operator in the drop-down menu, and then enter a value to compare the tag to. Discrete makes the tag's on or off state trigger the application.

Notification When Recipe Has Been Downloaded/Uploaded

1. Click Notification in the Download/Upload Recipe Manager dialog box.

The Recipe Download Completed Notification dialog box opens:



2. Click the Tag Selection button ? to quickly select an ioControl tag as the flag to indicate the recipe upload or download was successful. See "Configuring Tags" on page 5-11 for more information about configuring tags.

3. Select Discrete or Current Value in the Setup By group.

   Discrete sets, clears, or toggles the tag's on or off state. Current Value sets the tag value. To set a Current Value, select Set or Offset in the Current Values group and then enter a tag value.

Alarming

You can incorporate alarm features into your operator interface by adding an ioDisplay alarm graphic. In a project, alarm graphics monitor alarm points associated with ioControl tags, and alert the operator when pre-defined alarm conditions are reached. Alarm information can be logged to a file or sent to a printer.

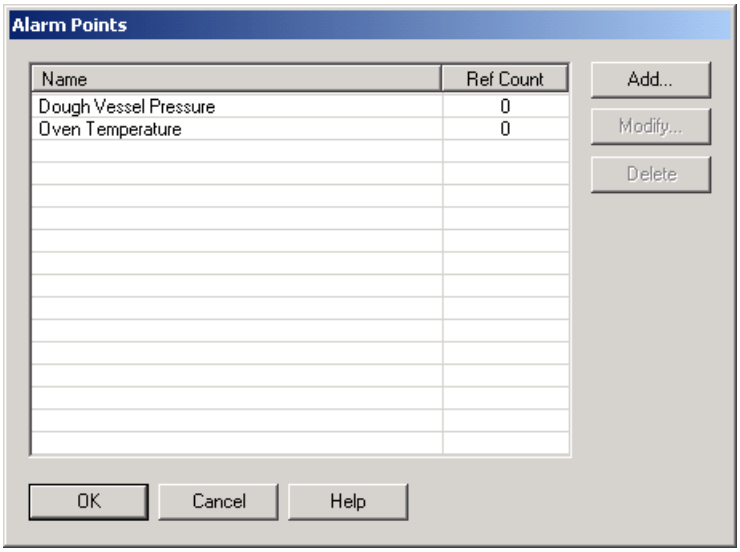
Configuring Alarm Points

Like a historic log point, an alarm point is linked to an ioControl tag. When an alarm point matches a defined alarm state, it is displayed on all alarm graphics that include that alarm point. If configured to do so, an alarm point can also be sent to any configured file or printer log.

Once it is set up, an alarm point can be included in any number of alarm graphics in an ioDisplay project. (Data for an alarm point can still be collected if the alarm point has not been included in a graphic.) You can easily generate a report that lists all the alarm points for a project; see "Viewing Tags and Dynamic Attributes" on page 7-33 for more information.

1. To configure an alarm point, choose Configure→Alarm Points.

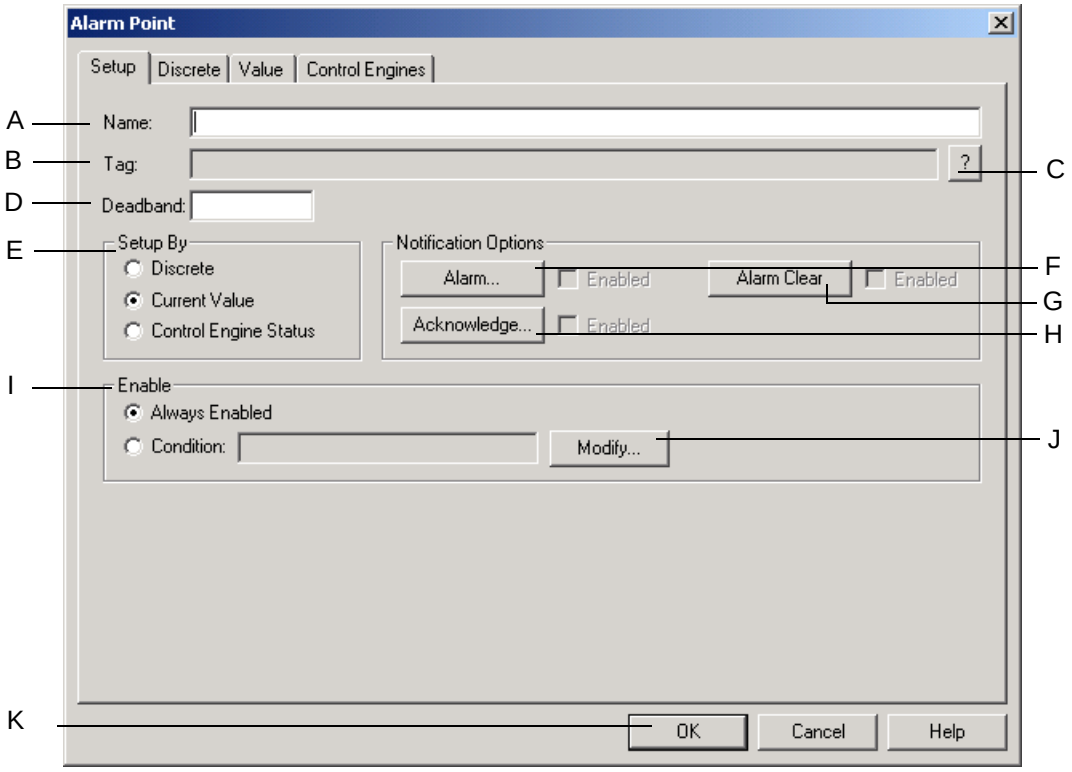
The Alarm Points dialog box that opens lists configured alarm points. The "Ref Count" column shows the number of alarm windows in which each alarm point is used.



2. Do one of the following:

- Click Add to create a new alarm point.
- Highlight an existing alarm point and click Modify to change it.
- Highlight one or more alarm points and click Delete to remove them. To select multiple alarm points, hold down either the SHIFT key (for selecting contiguous items) or the CTRL key (for selecting non-contiguous items) and click an alarm point name.

If you are creating or changing an alarm point, the Alarm Point dialog box appears:



A Enter a name for the alarm point here. The name of each alarm point in a project must be unique and must be less than 128 characters. The ioControl tagname appears by default.

B Displays the name of the ioControl tag that you select with C.

C Click the Tag Selection button ? to select an ioControl tag for the alarm point. Note that the choice of tags available is determined by the type of tag you select in E.

D Enter a deadband value to be added to or subtracted from the previously read tag value. This value adds a buffer, or deadband, that the next value read must be above or below to trigger the alarm point. For example, let's set the deadband at 5. If a tag is read and has a value of 50, the next tag reading must be greater than 55 or less than 45 for the alarm point to be triggered.

E Select the type of tag to be linked to the alarm point.

- Choose Discrete for discrete tags, such as a digital input. Selecting this option enables the Discrete page in the Alarm Point dialog box. See "Entering Discrete Alarm Conditions" on page 9-34 for configuration information.
- Choose Value for analog points, floats, or similar values. Selecting this option enables the Value page in the Alarm Point dialog box. See "Entering Alarm Values" on page 9-35 for configuration information.
- Choose Control Engine Status to link the status of a control engine to the alarm point. The alarm will be triggered whenever the linked control engine is not in

Attached state. See "Setting Control Engine Status Alarm Points" on page 9-36 for configuration information.

F Click here to display the Alarm Notification dialog box, where you can have a specified value written to a selected tag when the alarm point enters an alarm condition. See "Alarm, Acknowledge, and Alarm Clear Notifications" below for information on setting up alarm notification.

Select Enabled to make Alarm Notification active. (You can't select Enabled until Alarm Notification is configured.)

G Click here to display the Alarm Clear Notification dialog box. Use this dialog box to have a tag value be set or cleared when the current alarm returns to Normal from an alarmed state. See "Alarm, Acknowledge, and Alarm Clear Notifications" below for information on setting up alarm notification.

Select Enabled to make Alarm Clear Notification active. (You can't select Enabled until Alarm Clear is configured.)

H Click here to display the Acknowledge Notification dialog box, where you can have a specified value written to a selected tag when the alarm point is acknowledged by the operator. See "Alarm, Acknowledge, and Alarm Clear Notifications" below for information on setting up acknowledge notification.

Select Enabled to make Acknowledge Notification active. (You can't select Enabled until Acknowledge Notification is configured.)

I Choose Condition to make the alarm point dependent on the value of another tag. Click the Modify button (I) to select the tag and define the conditions it needs to meet.

Select Always Enabled for the alarm point **not** to be dependent on the value of another ioControl tag.

J If you've selected Condition in H to make the alarm point dependent on the value of another tag, click Modify and configure the tag and conditions in the Alarm Point Conditional Enabling Setup dialog box that appears. See "Setting Conditional Alarm Points" on page 9-32 for configuration information.

K Click OK to save your settings and close the dialog box.

Now we'll look at the additional steps needed to complete the settings in the Setup, Discrete, and Value pages of the Alarm Point dialog box.

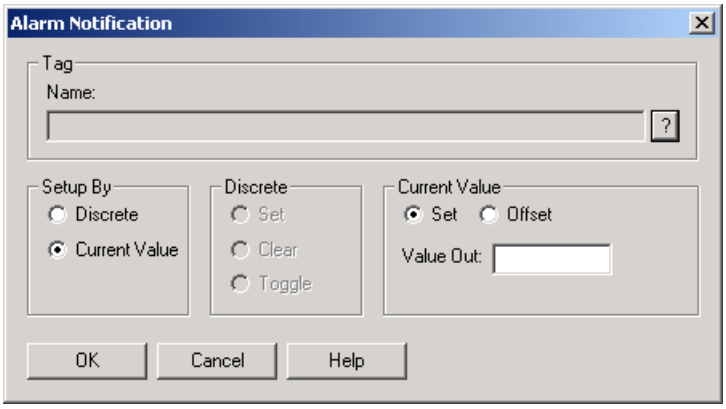
Alarm, Acknowledge, and Alarm Clear Notifications

You can have a value sent to a tag when any of the following alarm events occur:

- An alarm occurs
- An active alarm is acknowledged
- An active alarm returns to Normal from an alarmed state.

To configure an alarm point for these events, click Alarm, Alarm Clear, or Acknowledge in the Alarm Point dialog box.

The Alarm Notification, Alarm Clear, or Acknowledge Notification dialog box appears. These dialog boxes are identical. (The Alarm Notification dialog box is shown below.)



1. Click the Tag Selection button ? to select an ioControl tagname. See "Configuring Tags" on page 5-11 for more about selecting tags.

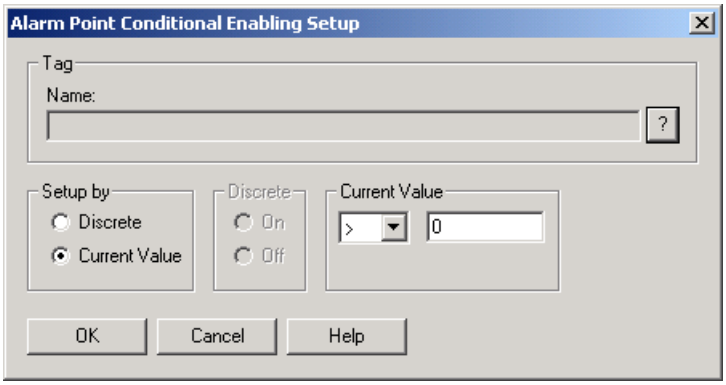
2. In the Setup By group, select Discrete or Value.
   - **Discrete** specifies an on/off trigger state for the selected tag. (The tag must have a discrete basetype.) After you select Discrete, define the trigger value by choosing one of the following options:
     - Set—Switches the trigger state to On.
     - Clear—Switches the trigger state to Off.
     - Toggle—Switches the trigger state from its current condition to the opposite (for example, On is switched to Off).
   - **Value** defines the floating point or integer value that is written to the tag. After you select Value, define the value by choosing one of the following options, then entering a value in the Value Out field:
     - Set—Replaces the tag's current value with the value you enter in the Value Out field.
     - Offset—Adds the value you enter in the Value Out field to the tag's current value.

3. Click OK to save your settings and close the dialog box.

Setting Conditional Alarm Points

1. To make the alarm point dependent on the value of another tag, select Condition in the Alarm Point Setup page, then click Modify.

The Alarm Point Conditional Enabling Setup dialog box opens:



2. Configure the tag and conditions.

3. Click the Tag Selection button ? and select an ioControl tag. This tag's value will be compared to a value you define.

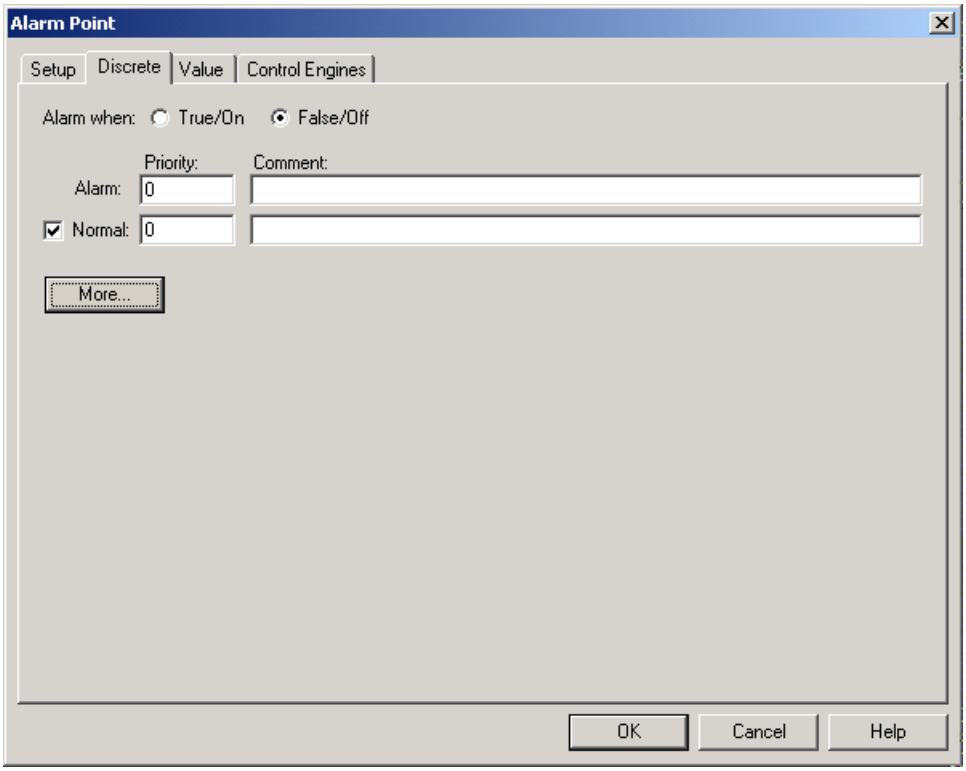
4. Select Discrete or Current value in the Setup by group. Current Value sets the tag value that will trigger the alarm point. Select an operator in the drop-down menu, and then enter a value to compare the tag to. Discrete makes the tag's on or off state trigger the alarm point.

   NOTE: Triggers are edge sensitive, and only activate on a positive transition from a non-triggered state.

5. Click OK to save your settings and close the dialog box.

Entering Discrete Alarm Conditions

If you selected Discrete in the Setup by field on the Alarm Point Setup page, complete the alarm point setup by configuring the alarm's state in the Discrete page.



1. To select the state of the alarm condition tag, select True/On to have the alarm point be in the alarm state when the associated tag is "on" for discretes, or "true" for integer bits. Select False/Off to have the alarm point be in the alarm state when the associated tag is "off" for discretes or "false" for integer bits.

2. Select the check box next to Normal to have the normal state displayed in history windows and logs. The normal state is the opposite of the alarm state.

3. When an alarm point state occurs, you can define how long it must be in that state (the *persistence time*) before the alarm point is triggered. Click More and enter the Persistence Time in milliseconds or seconds.

4. When an alarm point state remains *after* the alarm has been triggered and acknowledged, you can define how much time must elapse before the alarm point is re-triggered. Click More and enter the Re-Enable Time in milliseconds or seconds.

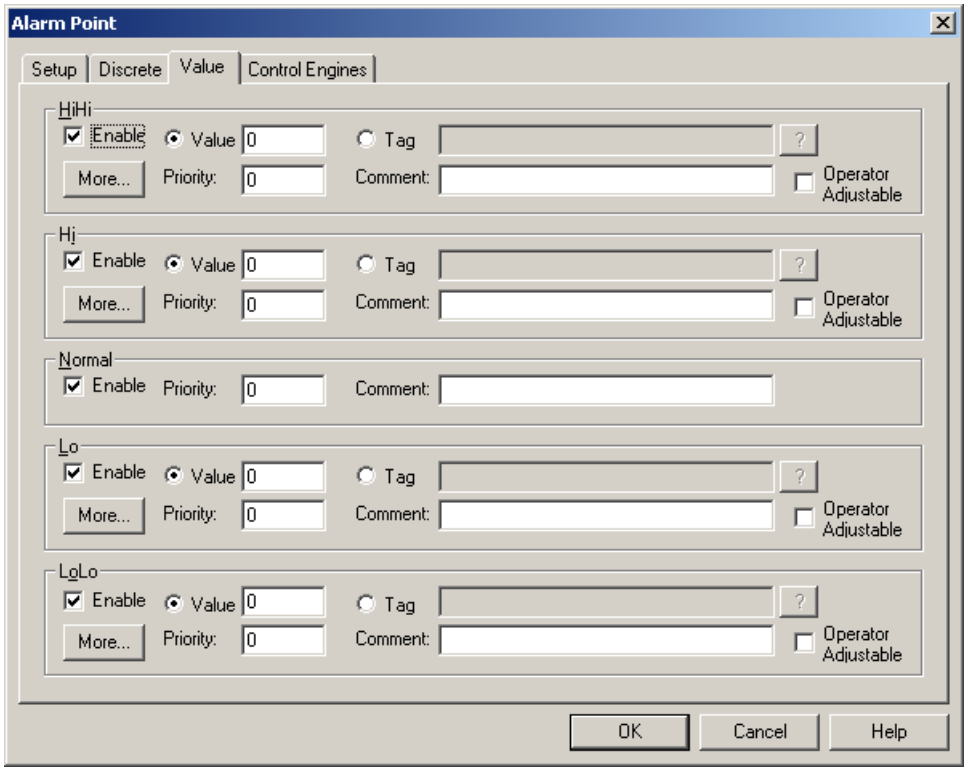
5. Enter an integer value between 0 and 999 in the Priority fields to define an alarm value for each alarm level. The highest priority is represented by 999, and 0 represents the lowest. Priority values can be useful in Runtime for displaying the relative importance of alarm points, and for filtering out alarms with lower priorities.

6. (Optional) In the Comment fields, enter text that will be displayed in alarm graphics for each alarm level. The comment can have a maximum of 256 characters. This comment can display information about the alarm point, for example, or provide instructions to the operator.

7. Click OK to save your settings and close the dialog box.

Entering Alarm Values

If you selected Value in the Setup by field on the Alarm Point Setup page, complete the alarm point setup by configuring the alarm's state in the Value page.



You must define values for each alarm level that will be used with the alarm point. The following alarm levels are available:

- **HiHi** alarms occur when the tag value is greater than or equal to the HiHi value.
- **Hi** alarms occur when the tag value is greater than or equal to the Hi value and less than the HiHi value.
- **Normal** is between the Hi and Lo values.
- **Lo** alarms occur when the tag value is less than or equal to the Lo value and greater than the LoLo value.
- **LoLo** alarms occur when the tag value is less than or equal to the LoLo level.

For each alarm level you want to use with the alarm point, do the following:

1. Select Enable to use an alarm level that you have configured. At least one alarm level (HiHi, Hi, Lo, or LoLo) must be enabled. If a level is enabled, it will be displayed in alarm graphics and logs.

2. To set an alarm level to a constant value, select Value and enter a number in the Value field.

3. To define the relative importance of an alarm point, enter an integer value between 0 and 999. (The highest priority is represented by 999, and 0 represents the lowest.)

   Priority values can be useful in Runtime for displaying the relative importance of alarm points. Additionally, in Runtime, you can filter out alarms with lower priorities.

4. When an alarm point value is reached, you can define how long it must continue at that value (the *persistence time*) before the alarm point is triggered. Click More and enter the Persistence Time in milliseconds or seconds.

5. When an alarm point value remains at a triggering level *after* the alarm has been triggered and acknowledged, you can define how much time must elapse before the alarm point is re-triggered. Click More and enter the Re-Enable Time in milliseconds or seconds.

6. To set an alarm level to the current value of a tag, select Tag and click the Tag Selection button  .

7. (Optional) In the Comment field, enter text that will be displayed in alarm graphics for each alarm level.

   The comment can have a maximum of 256 characters. This comment can display information about the alarm point, for example, or provide instructions to the operator.

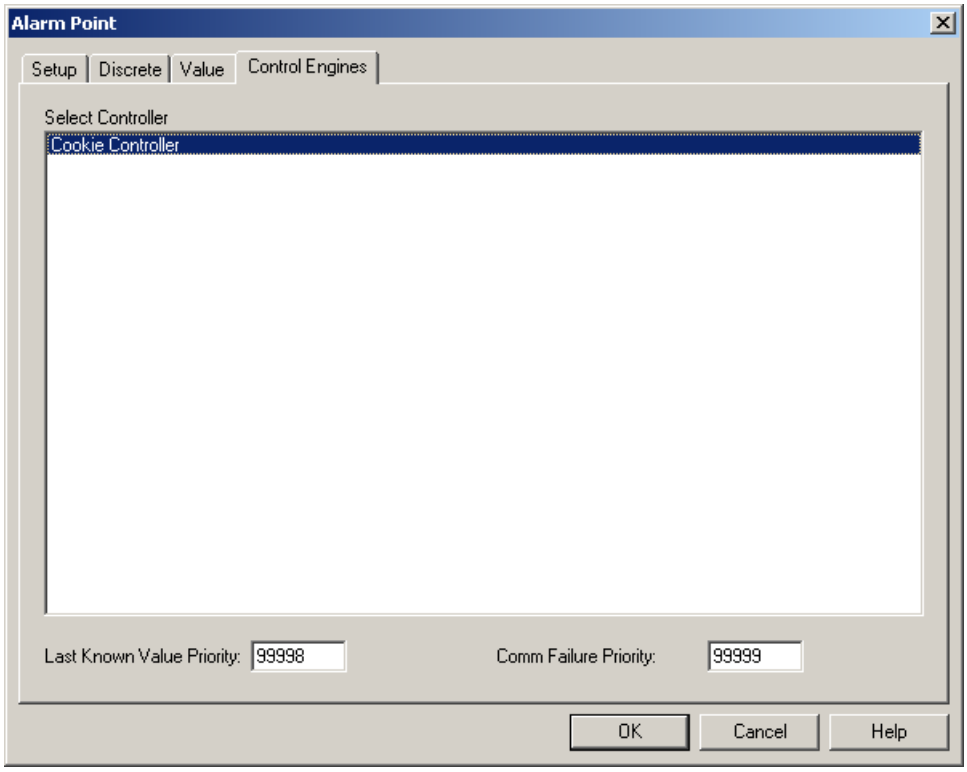
8. Select the Operator Adjustable check box to allow the operator to modify the conditions of this alarm point. See "Alarm Runtime and User Options" below to learn how to define these conditions.

9. Click OK to save your settings and close the dialog box.

Setting Control Engine Status Alarm Points

If you selected Control Engine Status in the Setup by field on the Alarm Point Setup page, complete the alarm point setup by selecting a control engine from the list on the control engines page. Only control engines that have been added to the ioDisplay project are available. See "Configuring Control Engines" on page 5-1 for more information on adding primary and backup control engines to an ioDisplay project.

NOTE: When the Control Engine Status option is selected, no other feature in the Alarm Point dialog box is available.



To configure an alarm point based on control engine status, do the following:

1. Click the Control Engines tab in the Alarm Point dialog box.
2. In the list of available control engines, select the control engine that will be linked to the alarm point.
3. If you want to change the Last Known Value Priority or Comm Failure Priority levels, enter a new value in the corresponding field.

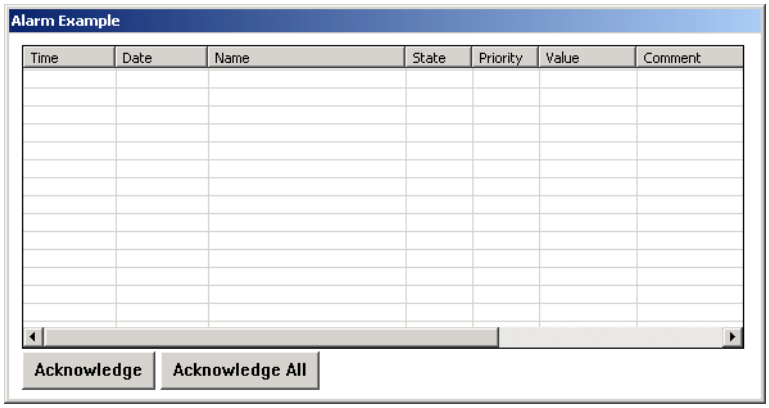
   Using the default settings, these priority levels cannot be filtered out by the user since they are higher than 999. If you want the user to be able to filter out control engine status alarms, set either—or both—values to 999 or less.
4. Click the Setup tab and enter a name for the alarm point in the Name field.
5. Click OK to close the dialog box and save your settings.

Adding Alarm Graphics

Alarm graphics can be placed and resized just like a trend or any other graphic object. You can place multiple alarm graphics in any window.

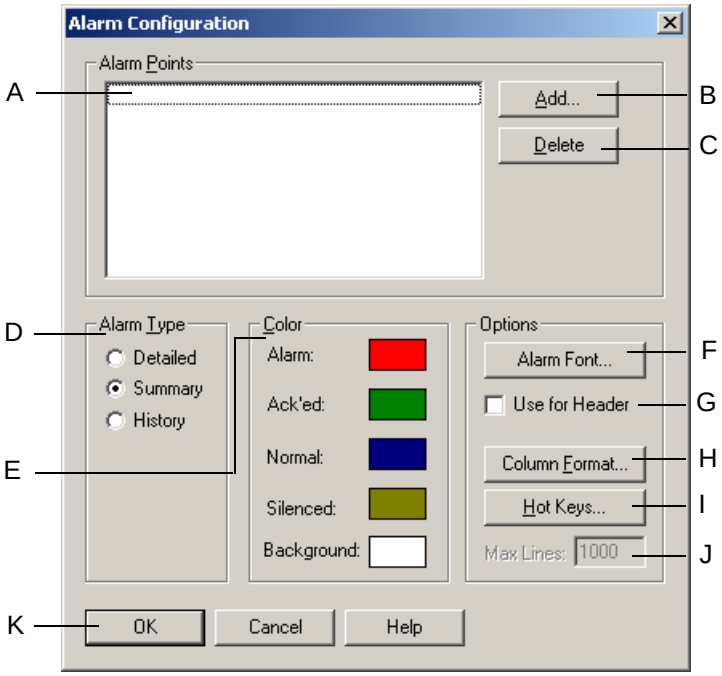
1. To create an alarm graphic, select the Alarm tool from the Toolbox.
2. Click and drag a large rectangle, and release the mouse.

   An alarm graphic similar to the example below appears:



3. To configure alarms for this graphic, choose the Select tool and double-click on the graphic.

   The Alarm Configuration dialog box opens:



A The Alarm Points list shows configured alarm points that are associated with the alarm graphic. To add an alarm point to the list, click B. To remove an alarm point from the list, click C.

B Click Add to add an alarm point to the ioDisplay project. In the Alarm Points dialog box that appears, you can select previously configured alarm points or configure new points

as needed. To select more than one alarm point at a time, press and hold the OPTION key and then click each point you want to add. See "Configuring Alarm Points" on page 9-28 for information on adding and configuring alarm points.

C Select an alarm point in the list and click Delete to remove it from the alarm graphic. The alarm point is not deleted from the ioDisplay project or from any other alarm graphic.

D Choose the type of the alarm graphic by selecting Detailed, Summary, or History.

- **Detailed alarms** treat each alarm point state as a separate alarm condition. The operator must acknowledge each alarm point before its name is cleared from the alarm graphic.

  For example, if an alarm is in the Lo state and changes to the LoLo state, alarms for both states are listed in the graphic. Alarm points can be selected and acknowledged from a detailed alarm.

- **Summary alarms** display only the state of the current alarm.

  For example, if an alarm is in the Lo state and changes to the LoLo state, only the LoLo state alarm is listed in the graphic. Alarm points can be selected and acknowledged from a summary alarm.

- **History alarms** list each change of state for each alarm point. Alarm points cannot be acknowledged from a history alarm.

E For each of the following items, click a color square and select a color in the Color dialog box that appears.

- Alarm points in an alarm condition
- Alarm points that have been acknowledged
- Alarm points that have returned to their normal state
- Alarm points that have been silenced
- Background color of an alarm graphic

F Click Alarm Font to select the font used in the alarm graphic.

G Select Use for Header to have the alarm font you selected appear in the alarm graphic's column headers.

H Click Column Format to set the information that appears on the alarm graphic. In the Alarm Format dialog box that appears, select the information the alarm graphic will display, and the width alarm graphic columns will appear on screen. See "Setting the Alarm Format" below for configuration information.

I Click Hot Keys to configure keys on the keyboard that the operator can use to acknowledge alarms. In the Alarm Hot Keys dialog box that appears, select keys or key combinations the operator can use to acknowledge one or more alarms. Only detailed and summary alarm graphics can have hot keys. See "Assigning Alarm Hot Keys" on page 9-40 for more information.

J Enter a number to set the maximum number of alarm point lines a history alarm graphic can contain. When this number is exceeded, the oldest alarm point is removed to make room for the new point. The History alarm type must be selected for this option to be available.

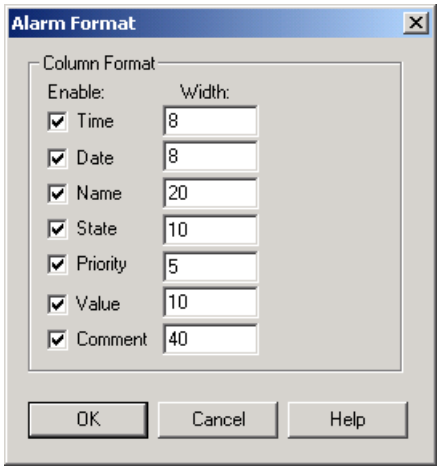
K Click OK to save your settings and close the dialog box.

Now we'll look at the additional steps needed to complete the settings in the Alarm Configuration dialog box.

Setting the Alarm Format

You can customize the alarm information that appears when an alarm graphic is displayed on screen, or when an alarm log file is sent to a printer.

1. In the Alarm Format dialog box, shown below, select the check box next to the name of each column that you want to appear in the alarm graphic or printed alarm log:



2. For each column name that you want to use, enter the desired column width (in pixels) in the Width field.

   For alarm graphics, the widths are an approximate guideline for how wide the columns will appear on screen. For printed alarm logs, the widths are absolute values. If a number or text cannot fit into a printer column, it will be truncated.

Assigning Alarm Hot Keys

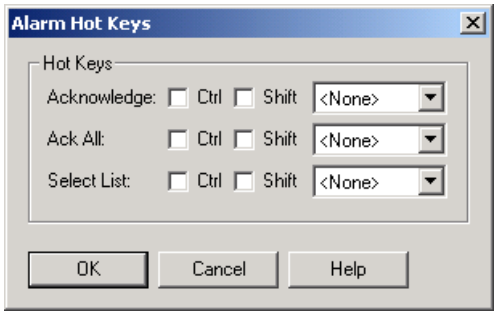
Alarm hot keys are keystrokes or keystroke combinations that the operator can quickly use to respond to alarms. When a hot key is defined for an alarm, pressing a key on the keyboard (along with an optional CTRL or SHIFT key) performs the same action as clicking the mouse on an object.

NOTE: Hot keys can also be defined for dynamic objects in an ioDisplay project, but these hot keys are defined separately from alarm hot keys. See "Adding Dynamic Attributes to Graphics" on page 7-2 to learn how to set up hot keys for a dynamic graphic.

You can define hot keys for the Acknowledge, Acknowledge All, and Select List functions.

- **Acknowledge** and **Acknowledge All** have the same effect as clicking those buttons on an alarm graphic.
- **Select List** is used to highlight the alarm point list on the display. Once the alarm point list is highlighted, cursor keys can be used to select alarm points for acknowledgment.

Alarm hot keys are configured in the Alarm Hot Keys dialog box, which is shown below:



To configure hot keys for an alarm, select a key in the drop-down list for the function you want to use. If you want to use the CTRL and/or SHIFT keys in combination with the key you've chosen, select Ctrl, Shift, or both.

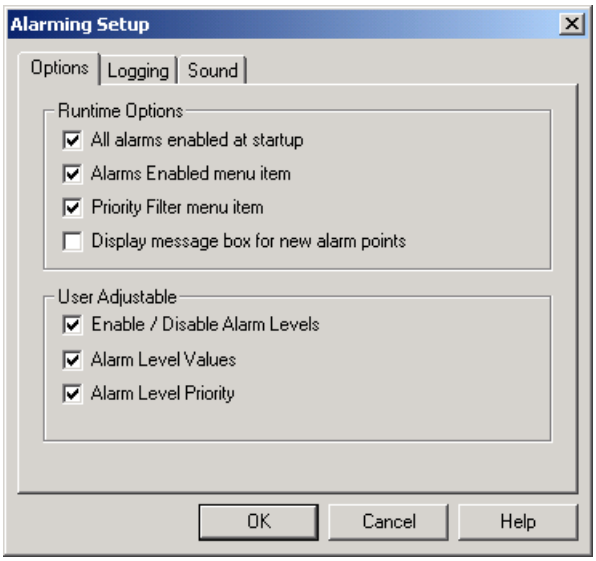
Configuring Project Alarms

To configure alarm features for the whole project, select Configure→Alarming Setup and configure the following settings as needed in the Alarming Setup dialog box:

Alarm Runtime and User Options

1. To set options for how an alarm appears in ioDisplay Runtime as well as how a user can work with alarms in an ioDisplay project, click the Options tab.

   The Options page appears. (This page appears by default when this dialog box opens.)



2. To set how an alarm appears in ioDisplay Runtime, select one or more of the following options in the Runtime Options group:

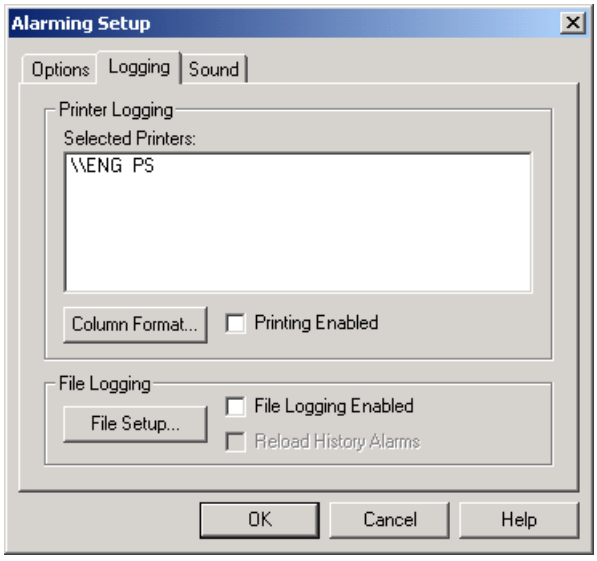
- All alarms enabled at startup—Enables all alarms when the project starts in Runtime.
- Alarms Enabled menu item—Makes the Alarms Enabled menu available for the user to enable and disable all alarms.
- Priority Filter menu item—Makes the Priority Filter menu available for the user to control whether to view all alarms, or only alarms exceeding a specified priority value.
- Display message box for new alarm points—Makes a dialog box appear containing information about the alarm point. The user can acknowledge the alarm in this dialog box, or close it and acknowledge the alarm at a later time.

3. To define the changes a user can make to an alarm in ioDisplay Runtime, select one or more of the following options in the User Adjustable group:
   - Enable/Disable Alarm Levels—Allows the user to enable or disable alarm points
   - Alarm Level Values—Allows the user to change alarm point values
   - Alarm Level Priority—Allows the user to change alarm point priorities.
4. Click OK to save your settings and close the dialog box.

Alarm Logging Options

1. To set options for how alarm data is sent to a printer or saved in a file, click the Logging tab.

   The Logging page appears:



2. To choose a printer to send alarm data to, select Printing Enabled and then choose a printer from the Selected Printers list. If you want to use a printer that does not appear on the list, you will need to install the printer so that it can be accessed from your PC. Refer to the documentation from Microsoft and your computer manufacturer if you are not sure how to do this.

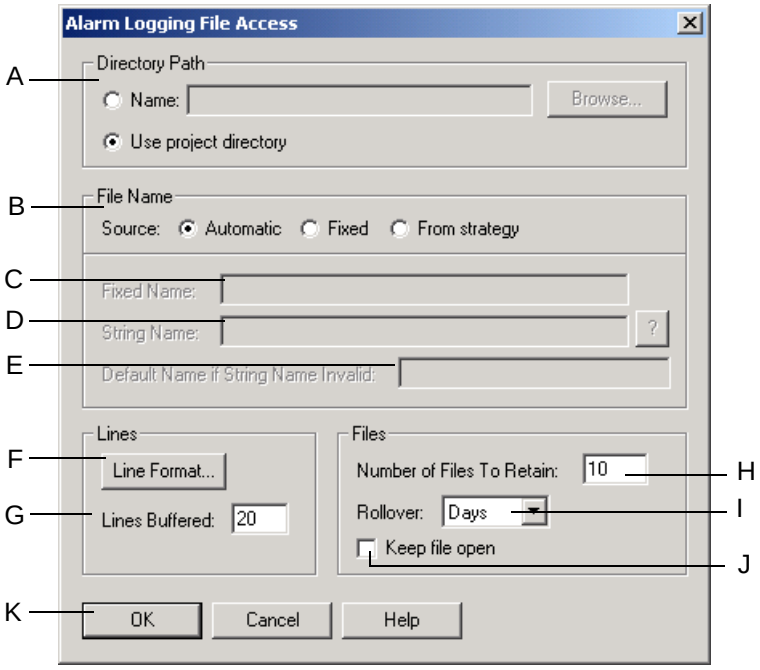
Note that if you move the ioDisplay project from one Windows operating system to another, you must reselect the printers.

3. To choose the alarm information that appears on the printed alarm log, click the Column Format button. In the Alarm Format dialog box that appears, set the format in which the alarm graphic will appear on screen and the information it will contain. See "Setting the Alarm Format" on page 9-40 for configuration information.

4. If you want History Alarm windows to be refreshed (have their contents updated) each time they open, select Reload History Alarms. This option is only available when file logging has also been enabled.

5. To save a log file of alarm data to disk, select File Logging Enabled and click File Setup.

   In the Alarm Logging File Access dialog box that opens, you can configure the name, location, line format, and other settings for the log file where the alarm data will be saved.



A Choose the directory where the alarm log file will be saved. Click Name and enter the directory path in the field next to it, or click Browse to find a directory path. Click Use Project Directory to save the alarm log file to the ioDisplay project directory. (This occurs by default if you don't specify another location.)

B Select Automatic, Fixed, or From strategy to determine how the alarm log file name will be created, and then fill in additional information as needed for that option.

If the Automatic option is used, log files are named based on the rules described in "About Data Log File Names and Formats" on page 9-9. If you select this option, files are named using the rollover convention if required; this is described in more detail on page 9-10. If rollover is not used, the file is named "alarmlog.alm." The Automatic option is used by default if you do not select another option.

C If you selected the Fixed option in B, enter a file name here. The file name can be any valid, eight-character DOS file name and doesn't require a three-character file extension. Note that if you don't specify an extension, one is *not* added automatically.

D If you selected the From strategy option in B, enter an ioControl string tagname here. Use the Tag Selection button ? to quickly enter the tag containing the name of the alarm log file. When the trigger starts the alarm log, the string containing the file name is read, and the new data is appended to the log file if the file already exists. If the file doesn't already exist, it is created. The rollover naming convention doesn't apply to this type of file name.

If the string in D is an invalid file name, the default name of the log file is created using the following rules:

- If the string is empty, the project directory is added to E and the extension .alm is used.
- If the string is not empty and a project directory is not specified as the directory path, the Name in A is added to E.
- If the project directory is specified as the path, or the previous step failed, the project directory is added to E and the extension .alm is used. If the project directory is read-only or there is not enough room left on the drive containing the project directory, an error message indicates that the file could not be created.

E Enter a default file name here in case the file name used in D is invalid. The file name can be any valid, eight-character DOS file name. The three-character file extension .alm is assigned by ioDisplay.

F Click to configure the character, or delimiter, used to separate the data in the log file, to choose the type of quotes used for each data line, and where to insert carriage returns. You configure these parameters in the Line Format dialog box that appears. See page 9-9 for more information.

G Enter the number of lines of data your PC will save to a memory buffer before writing the information to the alarm log file. When choosing a number, keep in mind that the lower the number of buffered data lines, the more frequently the computer has to write to the file. Alternately, the higher the number of data lines buffered in memory, the more data that will be lost if your PC loses power or has a system failure. A valid entry is any number between 0 and 999; the default is 20 files.

H Enter the maximum number of alarm log files that can be created using rollover before the oldest file is overwritten. For example, if you enter 10 and your rollover time period is set to hours, you will have 10 alarm log files created for 10 hours of data before the oldest file is overwritten with new data. See page 9-10 for more information on rollover settings.

I Choose the rollover time period here. Select None to have all logged data placed in a single data file named alarmlog.alm. Logging begins when the ioDisplayioControl project is loaded, and data collected will be appended to the existing data file. The size of the file is limited only by available disk space.

J Select Keep file open to leave the log file open to allow data to be appended to the alarm log file more quickly. If you leave this box unchecked (the default setting), the file is closed after each time data is written to it. This provides greater data integrity than leaving the file open.

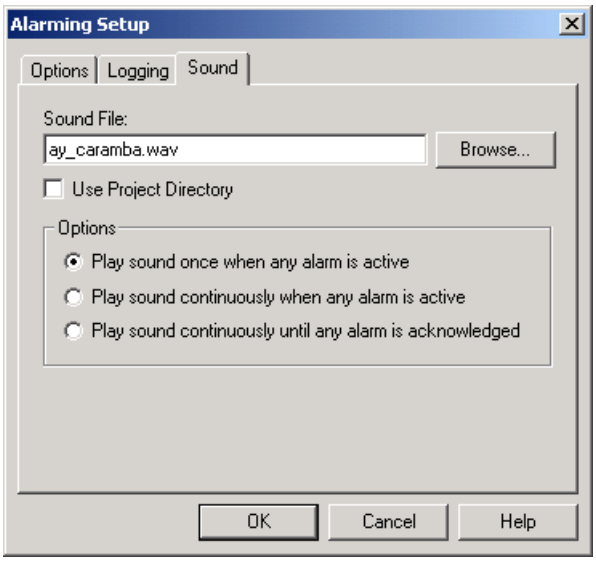
K Click OK to save your settings and close the dialog box.

Alarm Sound Options

NOTE: To play sound files, the PC running the project must have a properly configured sound card and corresponding system software, as well as a set of speakers. You can use both digitized sound (.wav) and MIDI music (.mid) files in your project.

1. To define an alarm sound and the conditions when it is played, click the Sound tab.

   The Sound page appears:



2. To enter the name of the sound you want played, click Browse and locate the sound file.

   If you select Use Project Directory, the sound file must reside in the project directory. This option is useful if the project directory might be moved to a different location.

   NOTE: Only one sound file can be selected in the Alarming Setup dialog box, but ioDisplay *has other ways of playing sounds. See "Configuring a Sound" on page 9-16 for more information.*

3. Select an option to determine how many times the sound will play when a new alarm occurs:
   - Play sound once when any alarm is active—The sound plays once, then stops.
   - Play sound continuously when any alarm is active—The sound continues to play until the operator acknowledges all active alarms.
   - Play sound continuously until any alarm is acknowledged—The sound continues to play until the operator acknowledges at least one active alarm.

4. Click OK to save your settings and close the dialog box.

CHAPTER 10

Using ioDisplay Runtime

Introduction

This chapter describes the versions of ioDisplay Runtime that can be used, and explains how to customize features that are available when your project runs in ioDisplay Runtime. It also explains how to use features that an operator sees and works with when using Runtime.

In This Chapter

Runtime Versions

Two versions of ioDisplay's Runtime application are provided with ioDisplay: the regular version and a monitor-only version. The primary difference between these two versions is that the monitor-only version of ioDisplay Runtime does not allow values to be sent to a control engine. This can be a useful feature for ioDisplay projects where operator intervention is not required or must be prohibited.

The monitor-only version of ioDisplay Runtime has the following features:

- The File and Help menus are the only menu items displayed. If the ioDisplay project has been configured to hide the menu bar, however, even these menus are not visible. See "Restricting the Operator" on page 10-7 for information on configuring the menu bar in Runtime.
- The only operator-driven dynamic attribute that can be used in the Runtime monitor-only version is opening or closing windows. Keep this in mind when developing the ioDisplay project; if there is a window that you do not want the operator to see, for example, do not use the open/close dynamic attribute with a graphic object.

Using Monitor-Only Runtime and Configurator

ioDisplay Configurator's feature Save Project and Load Runtime lets you switch quickly from Configurator to Runtime, and is a convenient way to test an ioDisplay project as you develop it. The Save Project and Load Runtime feature uses the regular version of ioDisplay Runtime. To use the monitor-only version of Runtime with this feature, you must change the file name of the application.

1. In Windows Explorer, change the file name of the regular version of ioDisplay to a temporary name.

   For example, change "ioDisR.exe" to "orig_ioDisR.exe" or something similar.

2. Now change the file name of the monitor-only Runtime application to the original name of the regular version. This is done by changing "ioDsRX.exe" to "ioDisR.exe".

After making these changes, the monitor-only version of Runtime will start when you select Save Project and Load Runtime in ioDisplay Configurator. To use the regular version of Runtime again, reverse the steps above.

Setting up Runtime

You can configure some of the ways that an ioDisplay project appears in Runtime. Using ioDisplay Configurator, you can specify which windows are open or closed, whether the menu is displayed, and whether or not the operator can exit the program. You can also customize options for the Event Log Viewer, a window that displays messages about the status and other characteristics of an ioDisplay project.

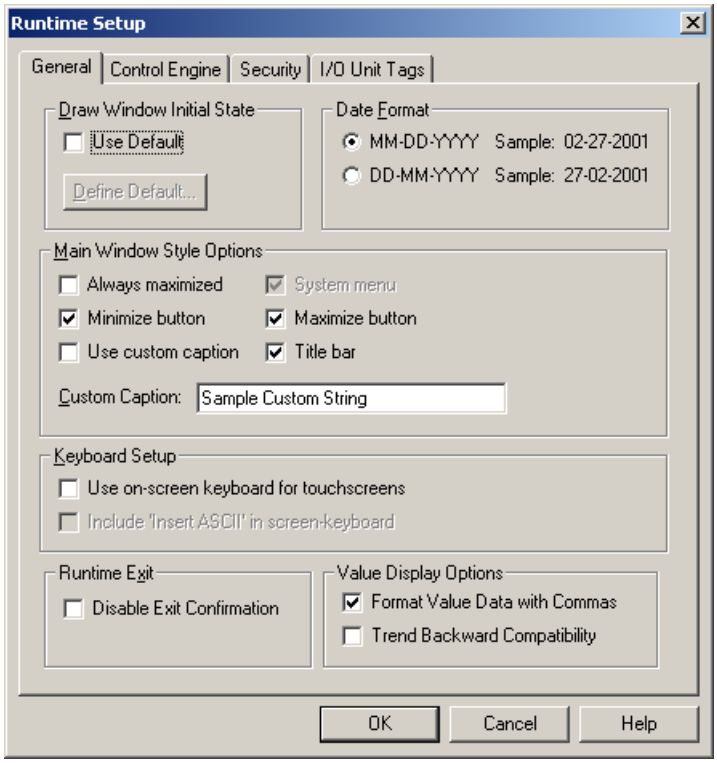
You can also control whether an ioDisplay project gathers I/O unit information, such as I/O point states and values, from tags on a control engine or directly from an I/O unit itself.

NOTE: If you want to run your ioDisplay project on a computer with multiple monitors, in ioDisplay Configurator simply extend the project's main window across the monitors you want to use. When you open the project in Runtime, the main window will appear the way you positioned it over the monitors. For more information on using multiple computer monitors, see "System Requirements" on page 1-5 and "Using Multiple Monitors" on page 3-5.

To set up a project for Runtime, select Configure→Runtime from the ioDisplay Configurator, then configure the settings in the Runtime Setup dialog box.

General Settings

Refer to the table below for instructions on configuring a particular option group under the General tab in the Runtime Setup dialog box.



For the Option Group	See
Draw Window Initial State	“Setting Up the Initial State of Windows” on page 10-3
Date Format	“Setting Date Format” on page 10-4
Main Window Style Options	“Setting Up the Main Window” on page 10-4
Keyboard Setup	“Configuring On-Screen Keyboard” on page 10-4
Runtime Exit	“Setting Up Exit Events” on page 10-4

Setting Up the Initial State of Windows

To configure how the project’s windows will look when the project is first opened in Runtime, check the Use Default check box, and then click Define Default to set the default options. In the Pop Window dialog box that opens, select windows and configure whether the window is opened, closed, or iconified. See “Using Draw Windows” on page 6-1 for additional options for configuring window states in Runtime.

Setting Up Exit Events

To keep ioDisplay Runtime from displaying a confirmation message when you close the program, select Disable Exit Confirmation.

Setting Up the Main Window

To choose the elements that appear in a project's main window, select any of the following options in the Main Window Style Options group:

- **Always Maximized** keeps the main window completely open, covering the entire screen.

  When this option is selected, the minimize and maximize button options are not available; deselect Always Maximize if you want to choose the minimize or maximize options.

- **Use Custom Caption** lets you enter a title that will appear in the title bar of the main window. Enter the title in the Custom Caption field.

  If a customized caption ends with a hyphen (-), the project file name is added to the caption.

- **Title Bar** displays the Windows title bar for the main window. If space is limited on your operator interface, deselect this option to slightly increase the viewable area that's available.

  If you deselect the Title Bar option, note that all options within the group except for Always Maximized are unavailable.

Configuring On-Screen Keyboard

To set up your ioDisplay project to run on a touchscreen terminal, select Use On-Screen Keyboard for Touchscreens in the Keyboard Setup group.

When this option is selected, you can use an on-screen keyboard for graphics that are configured using dynamic attributes to send strings and values.

Additionally, when Use On-Screen Keyboard for Touchscreens is selected, you can choose the option Include 'Insert ASCII' in Screen-Keyboard. With this second option selected, if a graphic has been configured using a dynamic attribute to send a string to the control engine, the operator can enter any character value between 0 and 255.

Setting Date Format

Use the Date Format options to change how the date appears in Alarm windows, SuperTrend objects, and historic log files. When you switch from one date format to another, any SuperTrends placed in a window are immediately updated to reflect the selected date format.

You can select one of two date formats:

- **MM-DD-YYYY** displays the date as month, day, and year.

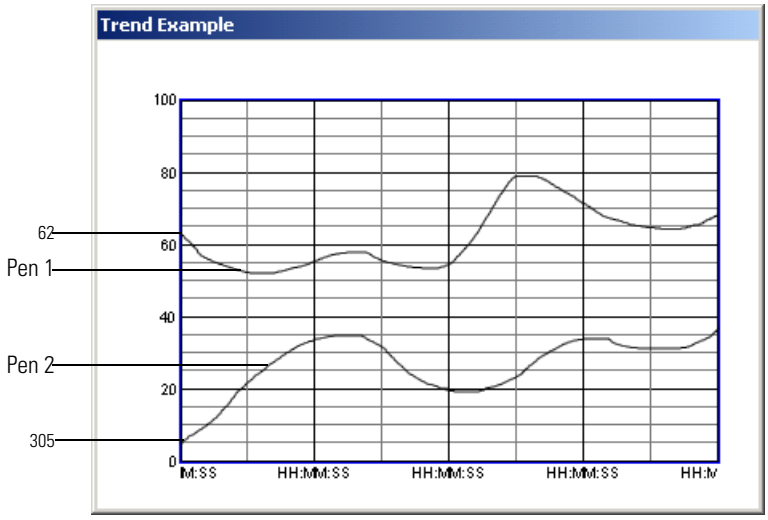
  For example, October 31, 1999 would be displayed as 10-31-1999.

- **DD-MM-YYYY** displays the date as day, month, and year.

For example, October 31, 1999 would be displayed as 31-10-1999.

Setting Value Display Options

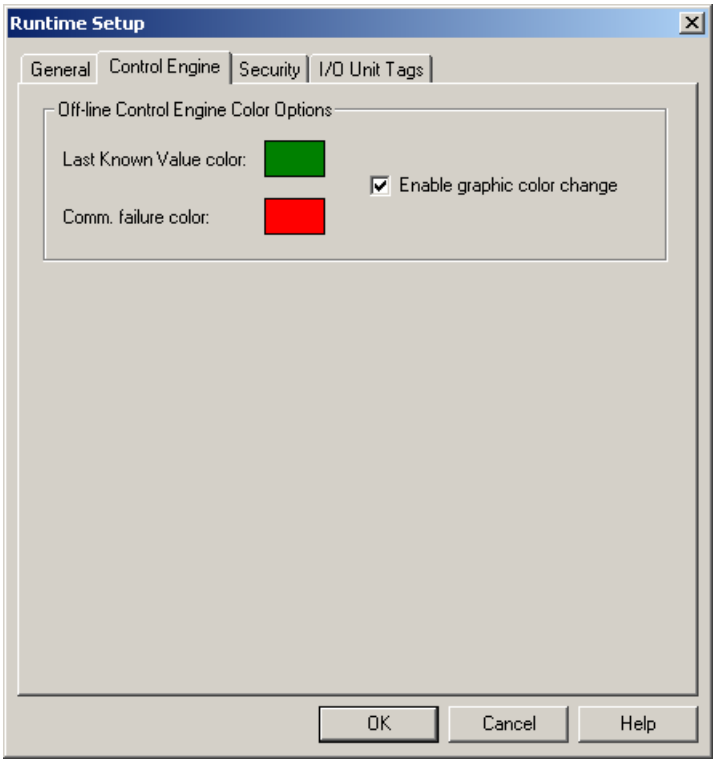
Trend Backward Compatibility displays Trend data in the same manner as it is displayed in OptoDisplay. This is the same effect as selecting None as the y-axis Label Position on the Trend Configuration dialog box, except that the labels are displayed. Multiple ranges appear to overlap, and all pens are displayed. As shown in the following example, if pen 1 has a range of 0-100 and pen 2 has a range of 300-400, the y-axis has a range of 0-100 *and* 300-400 simultaneously and both pens are displayed. By default, this option is disabled.



Format Value Data With Commas displays large numbers using commas. For example, one million displays as 1,000,000. However, if a data field in an existing project is too small to display large numbers with commas, you can de-select this option. Without commas, one million displays as 1000000.

Control Engine Settings

See "Changing Global Control Engine Color Options" on page 10-6 for instructions on configuring Control Engine options in the Runtime Setup dialog box.



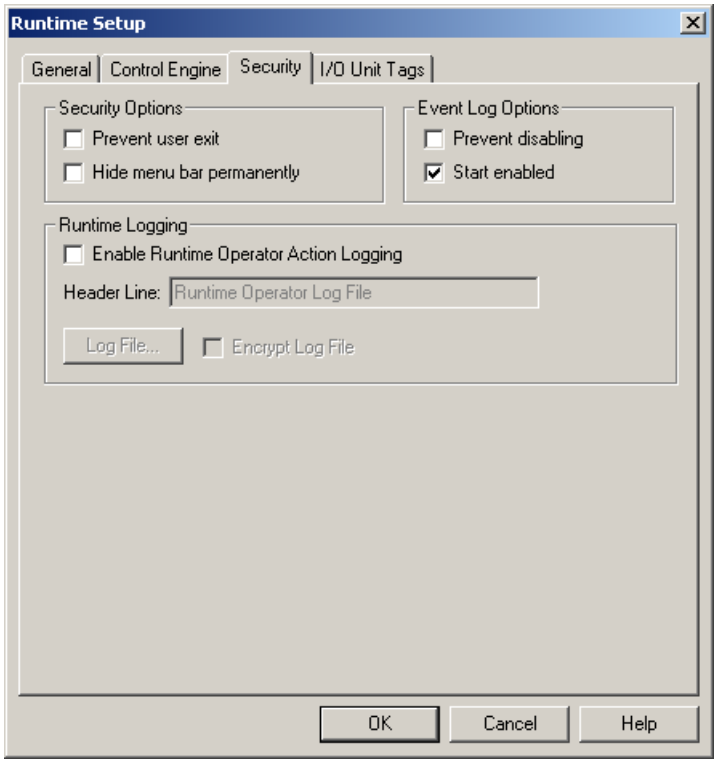
Changing Global Control Engine Color Options

If one or more control engines stops communicating, you can have all graphics with dynamic attributes tied to the control engine(s) change color to indicate the control engine's state. To do this, select a Last Known Value color and a Comm Failure color, and then click the Enable Graphic Color Change checkbox.

If any graphic in the project uses the View Control Engine Status dynamic attribute, the graphic will still change color based on control engine status. If the color used for this dynamic attribute is different from the "Offline control engine" color, you must confirm which color setting to use.

Security Settings

Refer to the table below for instructions on configuring a particular option group under the Security tab in the Runtime Setup dialog box.



For the Option Group	See
Security Options	“Restricting the Operator” on page 10-7
Event Log Options	“Enabling the Event Log Viewer” on page 10-8
Runtime Logging	“Logging Operator Actions” on page 10-8

Restricting the Operator

There are a few ways to limit how the operator can use ioDisplay Runtime:

- To prevent the operator from exiting ioDisplay Runtime, select Prevent User Exit in the Security Option group. Once the project starts in Runtime, an operator won’t be able to exit the ioDisplay Runtime application.
- To hide the menu bar from the operator, select Hide Menu Bar Permanently. This restricts operator interaction with Runtime menu commands to only what you’ve defined in the project itself. The ESC key will not activate the menu bar, and pressing the F1 key won’t invoke the Runtime online help system.

You can also configure a graphic so that Runtime commands execute when the graphic is clicked. This is done by assigning Runtime menu commands to the graphic using the Execute Menu Item dynamic attribute; see "Execute Menu Item" on page 7-9 for instructions.

Enabling the Event Log Viewer

The Event Log Viewer in ioDisplay Runtime is a window that displays messages about system events and communication transactions of a project. This window has an option for the operator to choose whether the window is enabled or disabled during the Runtime session, and another option for the operator to choose whether to display startup error messages. By default, the Event Log Viewer is enabled when Runtime starts; deselect "Start enabled" in the Event Log options group to keep the window from opening.

To keep the Event Log Viewer from being disabled (not displayed), select the option Prevent Disabling in the Event Log options group.

See "Using the Event Log Viewer" on page 10-14 for information on using the Event Log Viewer in Runtime, and "Configuring the Event Log File" on page 10-10 to learn about setting the format in which Event Log files are saved.

Logging Operator Actions

When an operator uses an ioDisplay project in Runtime, general information about how and when the project is used can be recorded in the Runtime Operator Log File. Detailed information such as which onscreen controls were used and which values or states were changed can also be recorded in this log file. For security, the log file can be optionally encrypted.

NOTE: You can also configure security settings for an object to restrict its use to authenticated users and groups. See "Security Settings for Graphics and Dynamic Attributes" on page 7-4 to learn how to configure user and group authentication for a graphic object.

Data Recorded in the Runtime Logging File When the ioDisplay project runs in ioDisplay Runtime, the following information about operator actions is recorded:

- *Date*—Date of action (month/day/year)
- *Time*—Time of action (24-hour)
- *Action Taken*—Description of action and ioDisplay project file used
- *Control Engine*—Control engine running the ioControl strategy that the ioDisplay project is accessing
- *Tag*—Complete name of tag being modified
- *Old Value*—tag value before being modified
- *New Value*—tag value after being modified
- *User*—Name of user logged into computer running ioDisplay project
- *Computer*—Computer running the ioDisplay project

In this example, the log shows that the operator "edgar" opened and closed the ioDisplay project "cfactory.UUI" using the computer "MFG-00".

ioDisplay Runtime Operator Log File

Line Formats:

Date	Time	Action Taken	User	Computer
12/11/2003	10:22:28.230	Open project: cfactory.UUI executed	edgar	MFG-00
12/11/2003	10:59:57.442	Close project: cfactory.UUI executed	edgar	MFG-00

The next example shows that the same operator "edgar" changed the setpoint value of "fTemperatureSetpoint" from 200 to 150 on the same project.

ioDisplay Runtime Operator Log File

Line Formats:

Date	Time	Control Engine	Tag	Old Value	New Value	User	Computer
12/11/2003	10:56:06.852	Cookie Controller	Cookie Controller:fTemperatureSetpoint	200.0000	150.0000	edgar	MFG-00

Configuring the Runtime Operator Action Log File To record operator actions, open the Security tab of the Runtime Setup dialog box and do the following:

1. Select Enable Runtime Operator Action Logging.
2. If necessary, change the header line to meet your application's requirements.
3. Click Log File.

   The Runtime Logging File Setup dialog box appears. Configuring a Runtime logging file is identical to configuring a Runtime event log; follow the instructions in "Configuring the Event Log File" on page 10-10 and then return to step 4 below.
4. Select Encrypt Log File if you want the log file saved as an encrypted document. See "Encrypting and Decrypting the Operator Action Log File" on page 10-9 for information on using encrypted files.

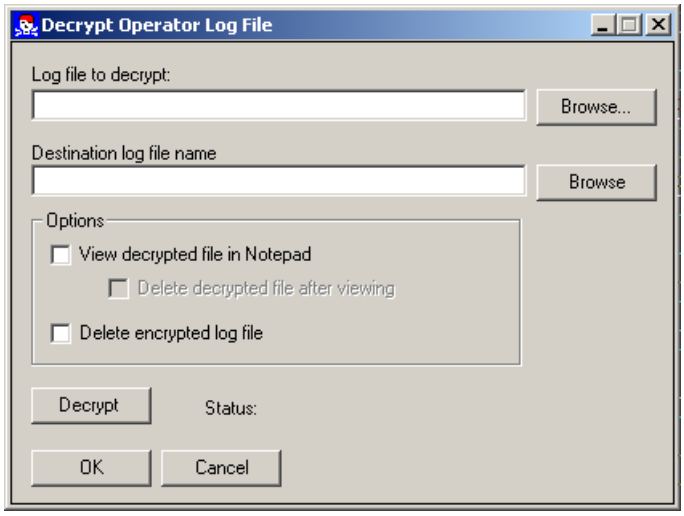
Encrypting and Decrypting the Operator Action Log File

To save a Runtime Operator Action log file in an encrypted format, select Encrypt Log File in the Security tab of the Runtime Setup dialog box.

To decrypt the log file, do the following:

1. In ioDisplay Configurator, select View→Decrypt Operator Log File.

The log file decryption window opens.



2. Click Browse next to the "Log file to decrypt" field and locate the encrypted operator action log file.

3. Click Browse next to the "Destination log file name" field and select a filename and location where the decrypted file will be saved.

   Skip this step if the default name and destination for the decrypted file is acceptable.

4. Select "View decrypted file in Notepad" to automatically open the file in Windows Notepad after it is decrypted.

   When this option is selected, you can also select "Delete decrypted file after viewing" to have the file deleted automatically when Notepad closes.

5. Select "Delete encrypted log file" to have the encrypted log file deleted automatically after it is decrypted.

6. Click Decrypt to decrypt the file.

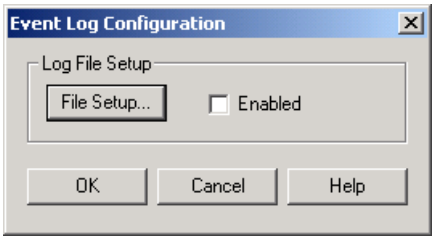
7. Click OK to close the log file decryption window.

Configuring the Event Log File

As events occur in a project in Runtime, messages with a date and time stamp are posted to the Event Log Viewer. These messages can be saved to an event log file so that the data can be used in other applications, such as Microsoft Excel or Microsoft Access. Event log files can also be archived to provide an operations record for an ioDisplay project. A project can only have one event log file active (open) at a time.

You can determine where the event log file will be located, configure how the data lines will appear, and define its rollover parameters in the Event Log File Configuration dialog box.

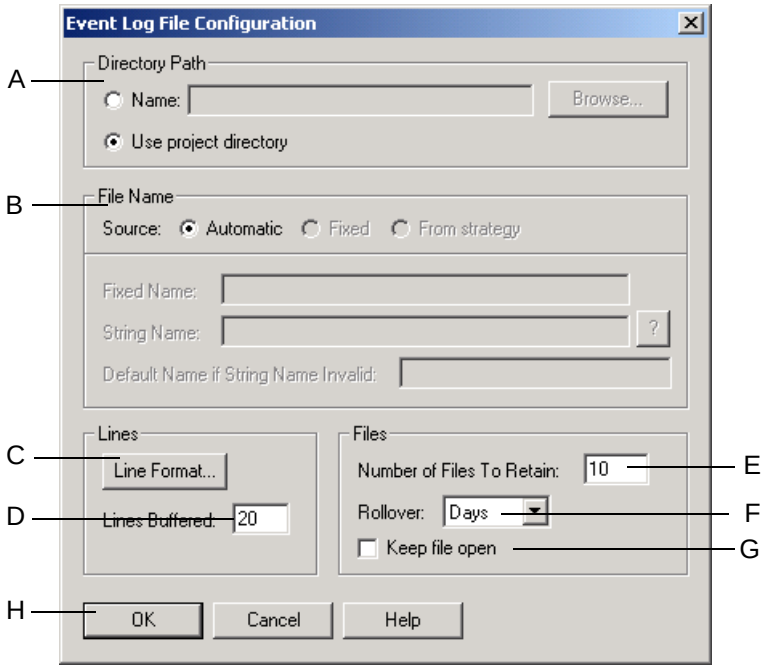
1. To configure an event log file, select Configure→Event Log.



2. Select Enabled in the Event Log Configuration dialog box to automatically create an event log file when you start the ioDisplay project in Runtime.

3. Click File Setup to customize the event log file.

   The Event Log File Configuration dialog box appears:



A Choose the directory where the event log file will be saved. Click Name and enter the directory path in the field next to it, or click Browse to find a directory path. Click Use Project Directory to save the event log file to the ioDisplay project directory. (This occurs by default if you don't specify a location.)

B Automatic is the only choice for creating the event log file name, and is selected by default.

C Click to configure the character, or delimiter, used to separate the data in the log file, to choose the type of quotes used for each data line, and where to insert carriage returns. You configure these parameters in the Line Format dialog box that appears. See "Setting Log File Line Format" on page 9-9 for more information.

D Enter the number of lines of data your PC will save to a memory buffer before writing the information to the event log file. When choosing a number, keep in mind that the lower the number of buffered data lines, the more frequently the computer has to write to the file. Alternately, the higher the number of data lines buffered in memory, the more data that will be lost if your PC loses power or has a system failure. A valid entry is any number between 0 and 999; the default is 20 files.

E Enter the maximum number of event log files that can be created using rollover before the oldest file is overwritten. For example, if you enter 10 and your rollover time period is set to hours, you will have 10 event log files created for 10 hours of data before the oldest file is overwritten with new data. See "Naming Files Using Rollover" on page 9-10 for more information on rollover settings.

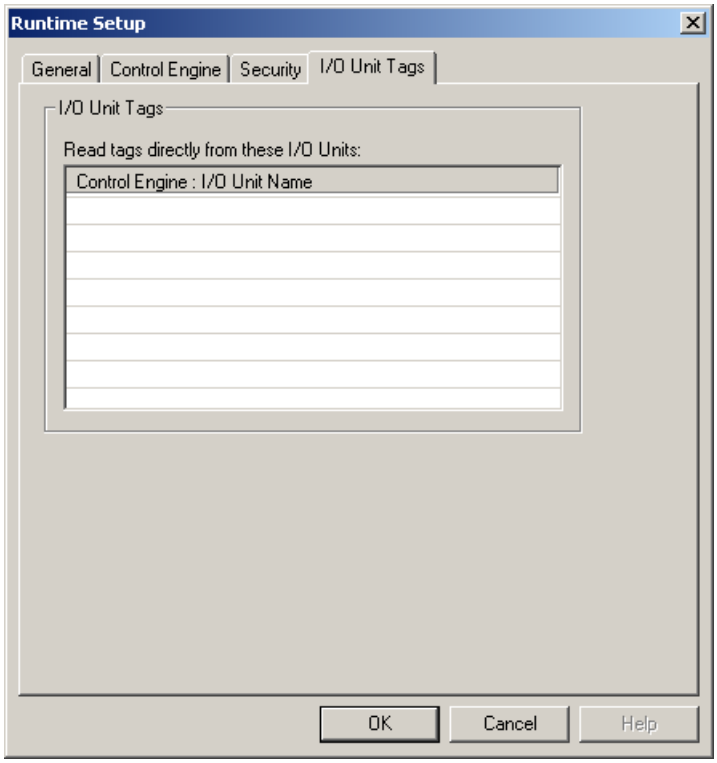
F Choose the rollover time period here. Select None to have all logged data placed in a single data file named eventlog.msg. Logging begins as soon as the project starts running, and data collected will be appended to the existing data file. The size of the file is limited only by available disk space.

G Select Keep file open to leave the log file open to allow data to be appended to the event log file more quickly. If you leave this box unchecked (the default setting), the file is closed after each time data is written to it. This provides greater data integrity than leaving the file open.

H Click OK to save your settings and close the dialog box.

I/O Unit Tag Settings

Refer to the instructions below to configure options under the I/O Unit Tags tab in the Runtime Setup dialog box.



Selecting I/O Units To Be Scanned Directly

Your control system probably uses Ethernet-based I/O units, which combine Opto 22 SNAP PAC R-series, SNAP Ultimate, SNAP Ethernet, or SNAP Simple brains with I/O modules and an I/O rack. In the I/O Unit Tags tab, you can configure how the ioDisplay project communicates with these units. Controlling how the ioDisplay project communicates with an I/O unit can improve the response time for reading and writing remote I/O points over getting that information from the controller.

Normally an ioDisplay project running on a PC gets information such as I/O point state or value by communicating directly with the I/O unit itself. Other information, however, is not obtained from an I/O unit (for example, the result of a calculation). Instead, this information comes from the tag variables on the control engine running the ioControl or OptoControl strategy. In the I/O Unit Tags tab, you can configure whether the ioDisplay project accesses I/O unit information directly from the unit, or only accesses that information from the control engine.

NOTE: Being able to specify how this communication occurs also allows you to use the dual Ethernet interfaces of the SNAP PAC R-series brains to segment part of an Ethernet network. See the Redundancy Technical Note *(Opto 22 form 1597) for more information on segmenting Ethernet networks with SNAP PAC S- and R-series controllers.*

For the current ioDisplay project, any Ethernet-based I/O units that are configured in the associated ioControl strategy will be listed in the section I/O Unit Tags. All I/O units are selected by default. For each I/O unit, do the following:

- Select the checkbox to have the ioDisplay project communicate directly with that I/O unit.
- Deselect the checkbox to have the ioDisplay project *not* communicate directly with that I/O unit, and instead get information from the control engine.

Using Runtime

Opening a Project

When your ioDisplay project is complete and you're ready to run it, there are two ways you can start ioDisplay Runtime and open a project:

- Click the Windows Start button, select Programs→ioProject Software→ioDisplay Runtime, and then select File→Open Project.

  In the Open Project dialog box that appears, select the ioDisplay project you want to open.
- With a project open in ioDisplay Configurator, select File→Save Project and Load Runtime.

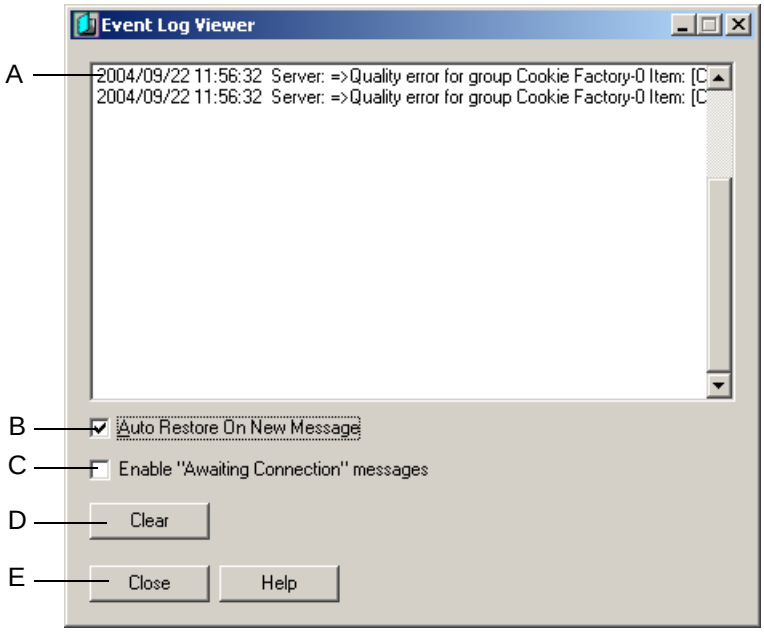
  Changes to the current project in ioDisplay Configurator are saved, and the project is opened in Runtime.

When the project opens, it will start running and you'll see the operator interface created in ioDisplay Configurator. The initial state of the draw windows that appear (open, closed, or iconified) is determined by the Runtime setup configuration. Unless you chose to hide the menu bar when you configured the project, the menu bar for the main window also appears.

Using the Event Log Viewer

The Event Log Viewer is started at Runtime by default. This window displays a list of communication transactions and error messages for a project. If the event log viewer is not open, choose View→Event Log to open it.

The window that appears will be similar to the following example:



A The list area posts event messages as they occur. Messages will have a date and time stamp, and a brief message describing the communications event that occurred. These messages are described in the appendix "ioDisplay Troubleshooting."

Messages posted to the Event Log can also be saved to a disk file. Refer to "Configuring the Event Log File" on page 10-10 for more information about doing this.

B To make the Event Log Viewer appear in the foreground whenever a new message is posted, select Auto Restore on New Message. This option won't be available if the Event Log Viewer was disabled when the project was set up in ioDisplay Configurator.

C To display or hide common error messages that occur when an ioDisplay project starts, select or deselect the Enable "Awaiting Connection" Messages option.

D Click Clear to erase the list of event messages.

After reviewing an event message, you can keep the Event Log Viewer window open, or close it using the Close button (E).

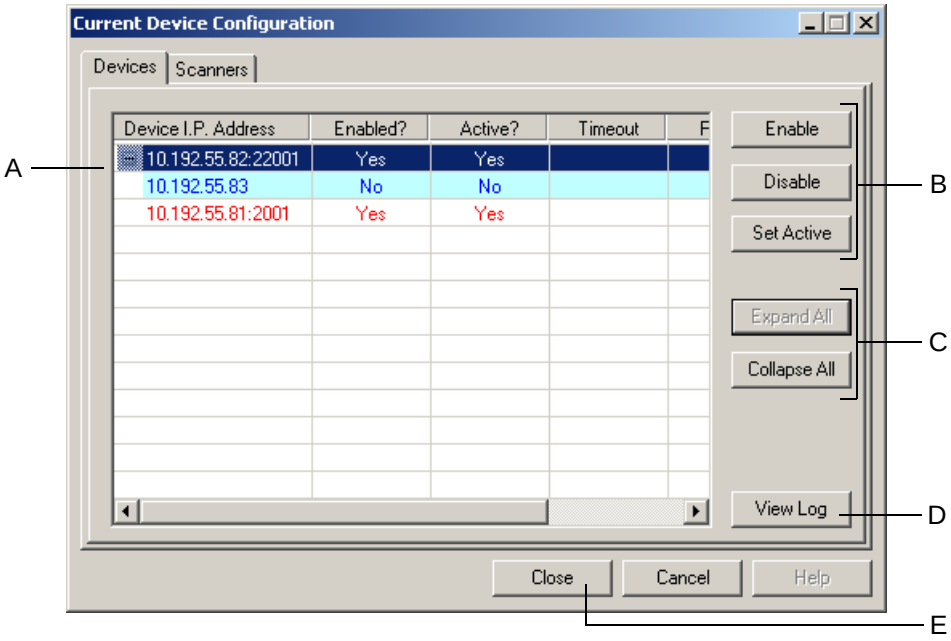
Working with Control Engines

While running an ioDisplay project in ioDisplay Runtime, there are several ways to view and work with one or more control engines that are running the strategy the project uses. When using ioDisplay Runtime Professional, you can also view and work with attached I/O units.

Displaying and Changing Device Configuration Status

To view and change the control engines and I/O units configured for the ioDisplay project, choose View→Configuration Status to open the Current Device Configuration window and then click the Devices tab. The information in this window updates automatically as control engines and I/O

units—or *devices*—are enabled or disabled, or change states between active and inactive. The Current Device Configuration window can be left open or minimized while the ioDisplay project is running.



A IP addresses for both primary and secondary control engines, as well as any Ethernet-based I/O units used with those control engines, are listed here. For each device the following information is displayed:

Device IP Address Assigned IP address for the device. If a secondary control engine or an I/O unit has been configured for that device, a plus sign "+" appears in front of the IP address. Click the plus sign "+" to see the IP address and status information for that device. Primary control engine IP addresses are displayed in red, and other devices are displayed in blue.

Enabled/Disabled State Device availability status; if the device is enabled, it is available for use with an ioDisplay project and can be made active. If the device is disabled, it cannot be made active.

Active/Inactive State Device activity status; an active device is a control engine or I/O unit that the ioDisplay project is currently trying to communicate with. An inactive device is one that is not being used for communication.

Timeout When a connection has been established with a device, this is the length of time (in ms) that the ioDisplay project will attempt to communicate with the control engine after no communications are received.

Retries When sending and receiving information to and from the primary control engine, this is the number of times that the ioDisplay project will attempt to communicate with the control engine after no communications are received.

Connect Timeout When trying to connect to the primary control engine, this is the length of time (in ms) that the ioDisplay project will try to communicate before switching to another control engine.

B Select a device and do one of the following:

- Click Enable to make that device available for use by the ioDisplay project.
- Click Disable to make the selected device unavailable to the ioDisplay project.
- Click Set Active to de-activate any currently active device and make the selected device active.

Timeouts and Retries—For any control engine that's selected, you can also change Timeout, Retries, and Connect Timeout values.

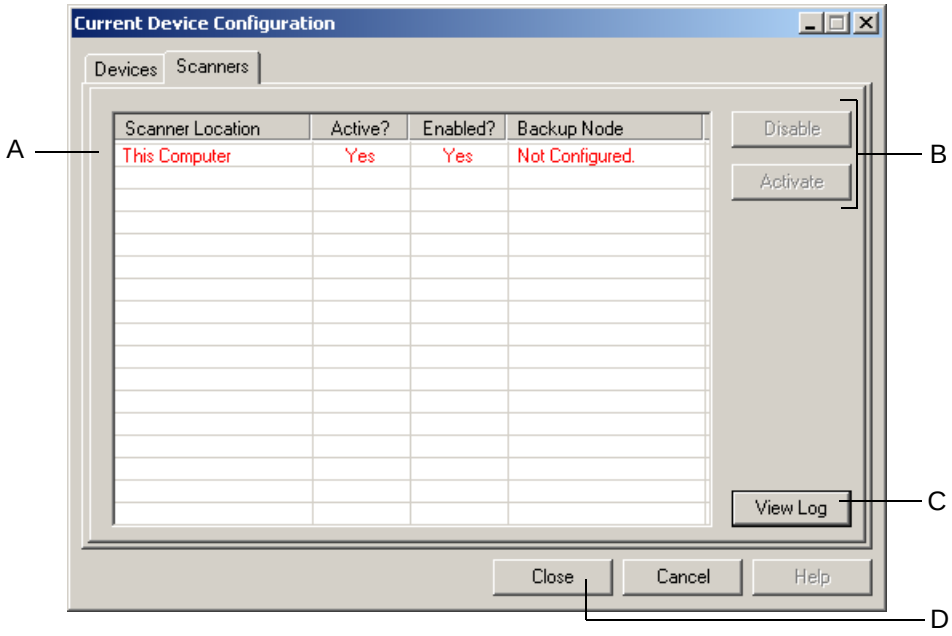
C Click Expand All to show primary and secondary control engines plus any Ethernet-based I/O units. Click Collapse All to show only the primary control engines and I/O units.

D Click View Log to open the file ConfigInfo.txt in Windows Notepad. This text file is created and updated whenever any of the device parameters listed in B are modified.

E Click Close to close the window and save any changes you made.

Displaying and Changing Scanner Configuration Status

To view and change the status of scanners used in the ioDisplay project, choose View→Configuration Status to open the Configuration Status window, and then click the Scanners tab. This window updates automatically as the scanners are enabled or disable, or change states between active and inactive, and can be left open or minimized while the ioDisplay project is running. (For information on what a scanner is, see "Configuring the Scanner" on page 5-6.)



A Scanners used by the ioDisplay project are listed here by the Windows Network name of the computer the scanner is running on. For each scanner the following information is displayed:

Scanner Location Name of the PC running the scanner. Primary scanners are displayed in red, and backup scanners are displayed in blue.

Active/Inactive State Scanner activity status; an active scanner is checking control engine tags and I/O unit values, while an inactive one is not.

Enabled/Disabled State Scanner availability status; if the scanner is enabled, it can be used with an ioDisplay project and can be made active. If the scanner is disabled, it cannot be made active.

Backup Node If a backup scanner has been configured, that scanner is listed here. (See "Configuring a Backup Scanner" on page 5-8 for instructions on doing this.)

B For an *inactive scanner,* select the scanner and do the following:

- Click Disable to prevent the ioDisplay project from using this scanner.
- Click Activate to force the ioDisplay project to switch to this scanner from the scanner listed under Backup Node.

For a *currently active scanner,* select the scanner and do the following:

- Click Disable to stop this scanner and force the ioDisplay project to switch to the backup scanner (if one has been defined).

C Click Close to close the window and save any changes you made.

Switching a Window between Control Engines

If a draw window's properties have been set to allow switching between control engines, you can access different control engines that are running the same ioControl strategy. When you switch control engines in Runtime, graphics that display control engine data will be updated to show data from the newly selected control engine. If five control engines are all running the identical strategy, for example, you only have to create one draw window instead of making a separate window for each control engine. For instructions on setting up a draw window, see "Modifying Draw Windows" on page 6-2.

There are some important considerations to note when using this feature:

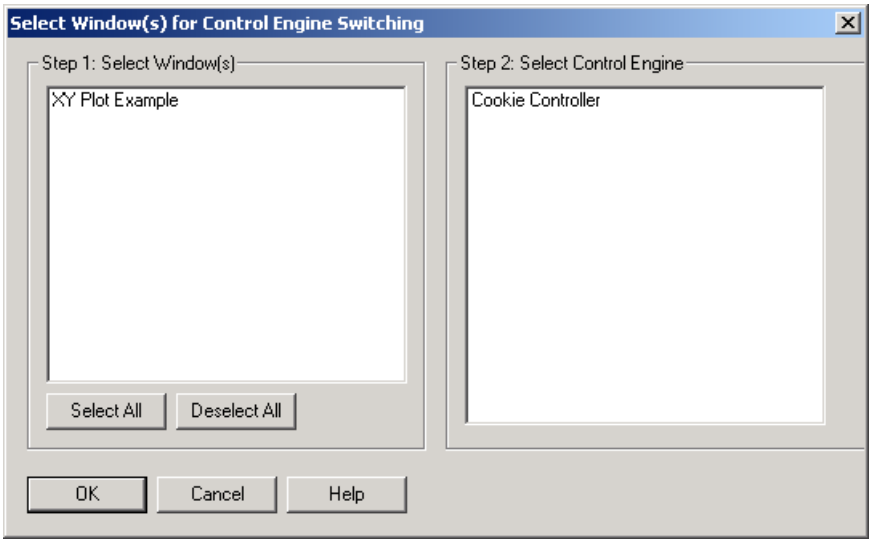
- Additional control engines that you want to switch between must be associated with the ioDisplay project using the Configure→Control Engines menu item.
- If you switch to a different control engine and then exit ioDisplay Runtime, when the project is restarted the default control engine for the project—not the one you switched to earlier—will be selected.
- Trends and SuperTrends must have the Disable Scanning option set to either "When Closed" or "When Minimized and Closed."
- All tags for the graphics in the window must reference the same control engine.
- Tagnames should not include the name of a control engine. For example, in ioControl you shouldn't name a variable "Control Engine1_flowrate" if the strategy uses a control engine named "Control Engine1."
- Control engine names must be at least three characters in length.
- The "Always in memory" windows property should not be selected.

- Alarms and graphics using the Alarm Point control engine-driven attribute may not be used.
- Recipes cannot be used.

To switch between control engines, start the ioDisplay project and do the following:

1. Select Window→Switch Control Engines.

   The Select Window(s) for Control Engine Switching dialog box appears.



2. Select the name of the window you want to view.
3. Select the name of the control engine you want to view.
4. Click OK.

   Graphics with dynamic attributes now use values from the control engine you selected. The name of the currently connected control engine appears in the window's title bar.

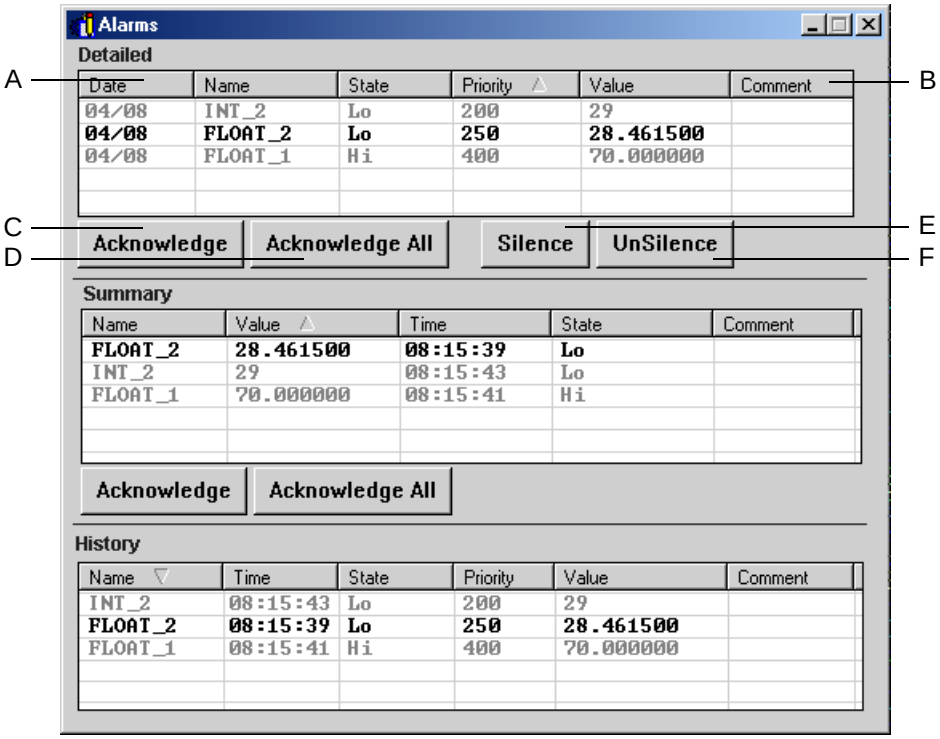
Working with Alarms

If an alarm in an ioDisplay project has been configured to let the operator do so, you can modify alarm points that appear, as well as change how the alarm information appears in the window.

Viewing Alarm Graphics

You may see detailed, summary, and history alarms in an ioDisplay project if it has been configured to let the operator do so. See "Adding Alarm Graphics" on page 9-37 to learn more about these types of alarms.

The sample alarm window below contains detailed, summary, and history alarm graphics.



A To sort the information that is displayed, click the column name of the item you'd like to sort by. All the alarm data that appears will be sorted based on the values in that column. Click the column name again to reverse the sort order.

B To change the location where columns appear in an alarm graphic, click the name of a column and drag it to a new location.

C To acknowledge and turn off a single active alarm, select an alarm and click Acknowledge. When an alarm has been acknowledged, it changes color so it can be easily identified.

D To acknowledge and turn off all active alarms, click Acknowledge All.

E (Detailed alarms only) To silence a single active alarm, select an alarm and click Silence. Silencing an alarm is similar to acknowledging an alarm, and the alarm will not re-alarm until it has been unsilenced. When an alarm has been silenced, it changes color so it can be easily identified.

F (Detailed alarms only) To unsilence an alarm that has been silenced, select an alarm and click UnSilence.

Modifying Alarm Points

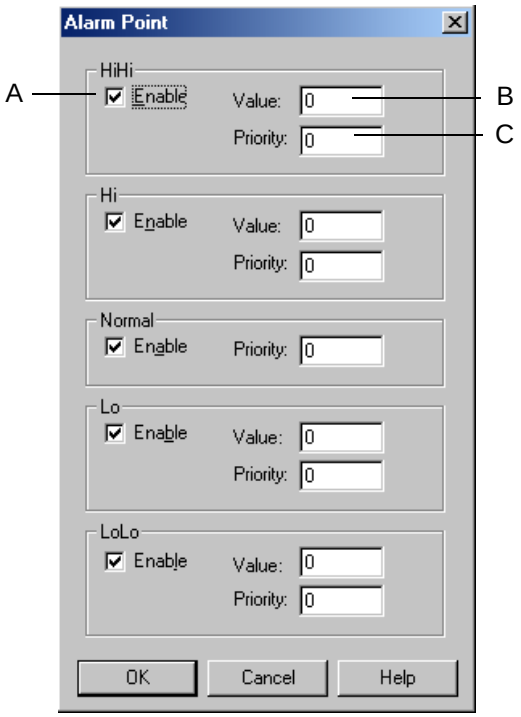
You may be able to modify alarm points in the ioDisplay project if it has been configured to let the operator do so. See "Configuring Alarm Points" on page 9-28 to learn more about configuring alarm points. Also see "Configuring Project Alarms" on page 9-41 for information about setting alarm options in Runtime.

Alarm point settings can be changed in each one of the four alarm ranges available: HiHi, Hi, Normal, Lo, and LoLo. Each alarm point state has a value that defines its range:

- **HiHi** alarms are greater than or equal to the HiHi Value.
- **Hi** alarms are greater than or equal to the Hi value and less than the HiHi value.
- **Lo** alarms are less than or equal to the Lo value and greater than the LoLo value.
- **LoLo** alarm are less than or equal to the LoLo level.

The normal state is between the Hi and Lo values. Each level can enabled or disabled, but at least one alarm state (HiHi, Hi, Lo, or LoLo) must be enabled.

To view the alarm point settings, select Alarms→Modify Alarm Points.



A Select this check box to enable the alarm level for an alarm range.

B Enter the value for an alarm level in the Value field.

C To set a priority for an alarm point, enter an integer value between 0 and 999.

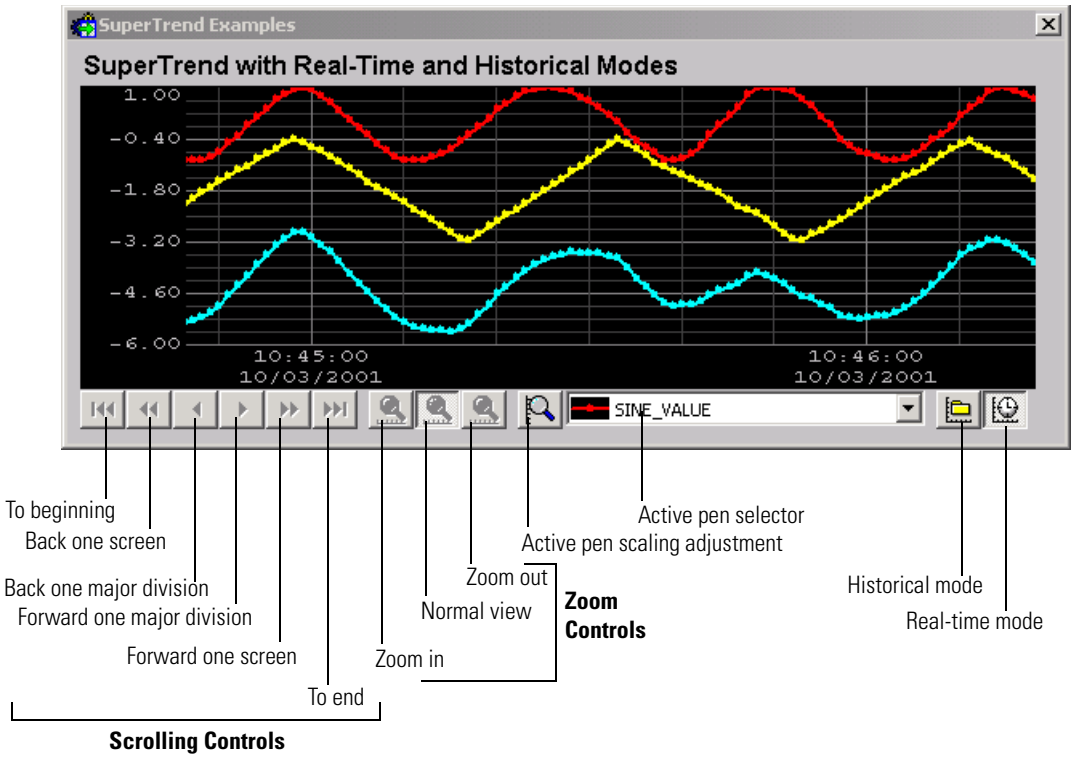
Priority fields define an integer value for each alarm level, and can be useful for displaying the relative importance of different alarm points. Additionally, you can filter out alarms with lower priorities.

Using SuperTrends in Runtime

If a SuperTrend graphic is included in an ioDisplay project, you can control how you view the trend data that appears in the SuperTrend window. For example, you can zoom in to see a smaller slice

of a trend line, or, if historical data is being collected, switch between views of real-time and historical data.

A sample SuperTrend window appears below. It shows the controls you can use to display SuperTrend information. Note that most of the controls that appear below are available only when historical mode has been selected.



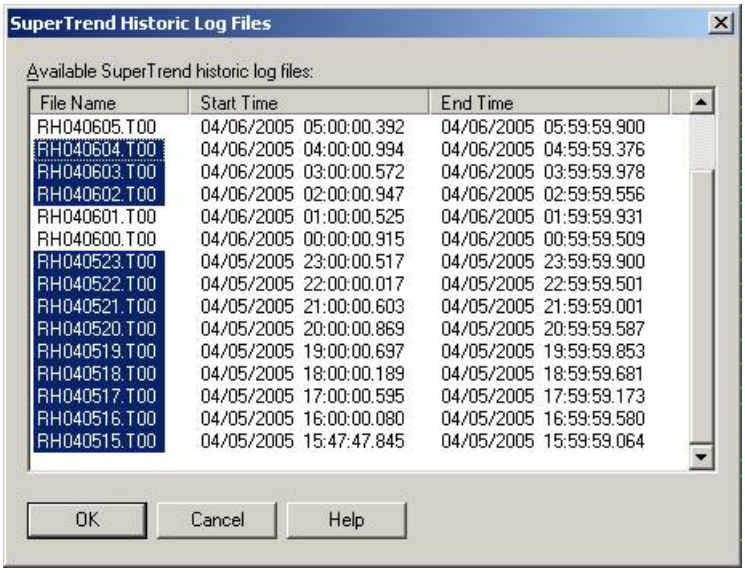
- Use **scrolling controls** to move back and forth in a chart of historical data.
- Use **zoom controls** to magnify or demagnify your view of a chart.
- To **select an active pen**, click the drop-down list and select a pen from the names that appear. If the y-axis scaling was based on pens (see "Configuring Y-Axis Parameters" on page 8-11), the scale of the active pen will be displayed.
- To **change the scale of an active pen,** click the Active pen scaling tool and enter new minimum and maximum values in the dialog box that appears.

Switching between Historical and Real-Time Modes

If a SuperTrend is both a historical and a real-time trend (that is, historical data is being collected), you can easily change views to see real-time or collected historical data.

1. To switch between historical and real-time modes, click the Historical mode button or the Real-time mode button .

When you switch from Real-time mode to Historical mode, the SuperTrend Historic Log Files dialog box opens, listing the names of SuperTrend historic log files and the time each log started and stopped recording.



2. To view SuperTrend historic log files, select one or more files and click OK.

   To select multiple log files, hold down either the SHIFT key (for selecting contiguous items) or the CTRL key (for selecting non-contiguous items) and click a file name.

   The information in the log file will be shown in the SuperTrend chart. Use the controls at the bottom of the chart to view the information. (These controls are described on page 10-22.)

3. When you are finished viewing the SuperTrend historic log file, click the Real-time Mode button to return to the real-time view of SuperTrend information.

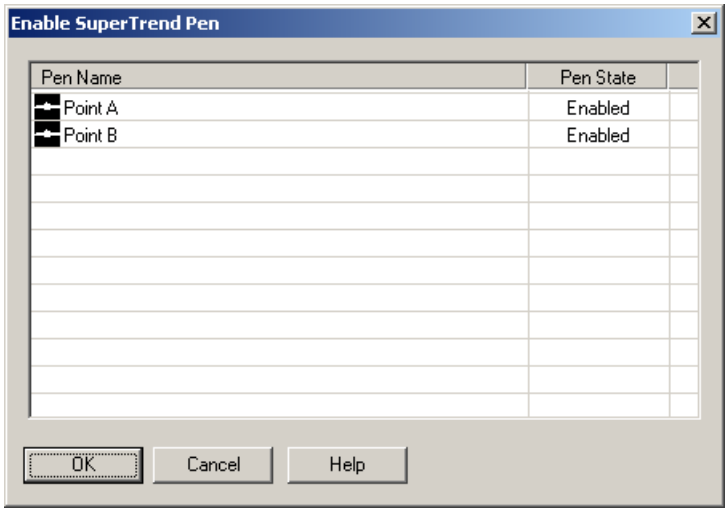
   Note that you will need to reselect log files each time you switch to historical mode.

Enabling and Disabling SuperTrend Pens

Individual SuperTrend pens can be displayed (enabled) or hidden (disabled) in Runtime.

1. Right-click on a SuperTrend.

The Enable Supertrend Pen dialog box appears.



2. Click a pen name to toggle it between these states:
   - **Enabled**—pen data is being scanned and logged to a file (if configured).
   - **Hidden**—pen data is being scanned and logged to a file (if configured), but the pen data is not graphed on the SuperTrend.
   - **Not Scanning**—pen data is not scanned or logged to a file (if configured).

When enabling and disabling SuperTrend pens, note the following:

- The active pen in the SuperTrend object cannot be toggled between enabled, hidden, and not scanning.
- Both real-time and historical data is displayed based on a pen's state. For example, if a pen is hidden, it will not be shown in either real-time or historical views.
- Only pens with enabled point markers (set in ioDisplay Configurator) can be hidden.

Using XY Plots in Runtime

You can change the range of values used for the x-axis and y-axis of an XY plot object if this option has been set in ioDisplay Configurator. To change range values, right-click on the XY plot and in the dialog box that appears enter new minimum and maximum values for each axis.

APPENDIX A

ioDisplay Troubleshooting

This appendix provides tips and procedures for resolving problems you may encounter while creating or running your ioDisplay project.

If you are having problems with creating an ioControl strategy, see Appendix A, "ioControl Troubleshooting," in the *ioControl User's Guide.* For information about types of errors and lists of error messages that may appear in ioDisplay Runtime, see Appendix B, "ioDisplay Errors."

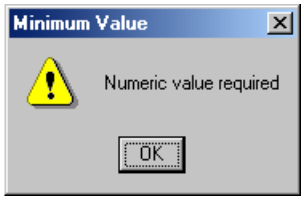
How to Begin Troubleshooting

Most errors that you encounter in ioDisplay occur when you run your project in the Runtime component. Runtime errors can be the result of several factors: problems communicating with the control engine, problems in communication between the control engine and I/O, or problems in how an on-screen object is configured. Errors occurring in ioDisplay Configurator, on the other hand, are usually due to incorrect values being entered in dialog boxes.

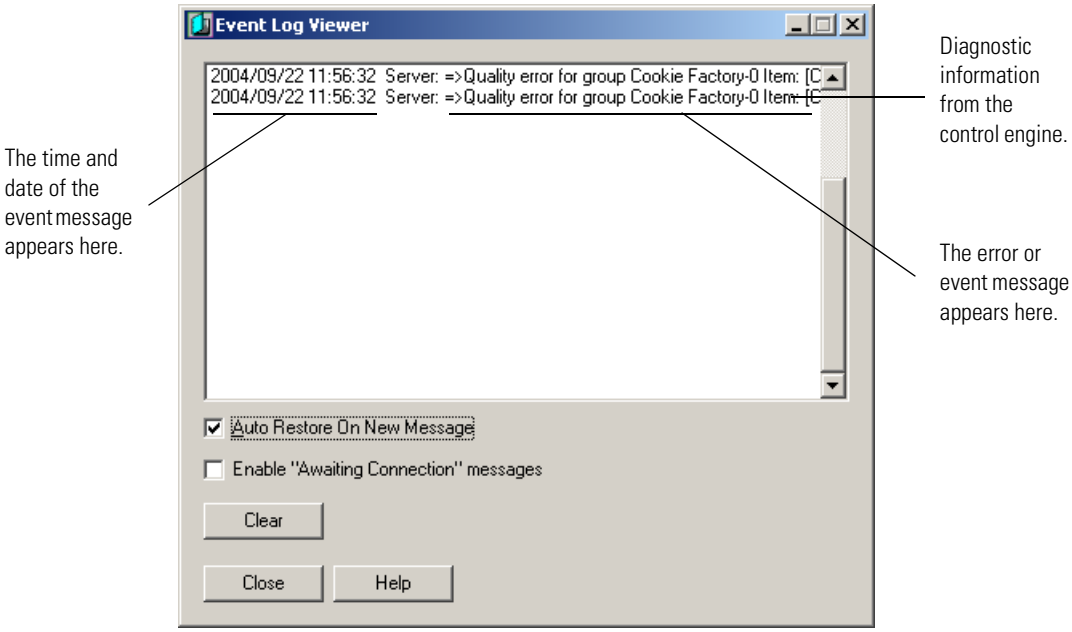
The following steps may help you track down the cause of an ioDisplay error:

1. Read any Error or Event Messages

Error messages in ioDisplay Configurator appear in standard message windows. These messages usually indicate how to correct the reported problem, as shown in the examples below:



Error or event messages in ioDisplay Runtime appear in the Event Log Viewer as the project runs. To open the Event Log Viewer window, select View→Event Log. The messages appear along with other diagnostic information related to your project, as shown in the example below:



See Appendix B, "ioDisplay Errors," for information about error messages that may appear in ioDisplay Runtime.

2. Check Communication with the Control Engine

If no error message appears, or the error indicates that there may be a communication problem, first check whether the PC running ioDisplay is communicating with the control engine. Next, check that communication settings specific to ioDisplay are configured correctly.

a. Follow the communication troubleshooting procedure in Appendix A, "ioControl Troubleshooting," in the *ioControl User's Guide.* If this does not resolve the communication problem, return to this page and continue with the step below.

b. Check the refresh time settings used in the ioDisplay project. These settings determine how frequently a tag on a control engine is scanned by ioDisplay.

See Chapter 5, "Configuring Control Engines and Tags" and "Scanning to Update Graphics" on page 7-35 to learn how these settings are configured and optimized.

3. Review Other Sections in this Appendix

Check the other sections in this appendix for the following items:

- If multiple "Bad Quality" or "Not Connected" messages appear when ioDisplay Runtime starts, see "Hiding or Displaying Runtime Startup Messages" on page A-3.

- If the colors in on-screen graphics are incorrect or change repeatedly, see "Problems Displaying a Project" on page A-4.
- If you are having problems saving project files to your hard drive or other storage location, see "Problems Saving a Project" on page A-4.
- If an on-screen text string object disappears when you run the project, see "Making an Empty String Visible" on page A-5.
- If you are having problems accessing control engines on a computer running Microsoft Windows NT, see "Problems with Windows User Permissions" on page A-5.

4. Call Product Support

If you cannot find the help you need in this book or the *ioControl User's Guide,* call Opto 22 Product Support. See "Product Support" on page 4 for contact information.

Hiding or Displaying Runtime Startup Messages

When an ioDisplay project starts in Runtime, multiple "Bad Quality" or "Not Connected" messages may appear in the Event Log Viewer. These startup messages appear when the ioDisplay project requests values from the scanner before the scanner communicates with the control engine.

ioDisplay Configurator

You can use ioDisplay Configurator to prevent these messages from appearing. To hide or display these Runtime startup messages, do the following:

1. In ioDisplay Configurator, select Configure→Scanner Location.

   The Select Scanner Location dialog box appears.

2. Select or deselect the Enable "Awaiting Connection" Messages checkbox to display or hide any "Bad Quality" or "Not Connected" messages that may occur when the ioDisplay project starts.

3. Click OK to close the dialog box and save your settings.

ioDisplay Runtime

When running the project in ioDisplay Runtime, the operator can also prevent these error messages from appearing in the Event Log viewer. To hide or display these Runtime startup errors, do the following:

1. In ioDisplay Runtime, select View→Event Log.

2. In the Event Log viewer window that appears, select or deselect the Enable "Awaiting Connection" Messages checkbox to display or hide any messages that may appear when the ioDisplay project starts.

3. Click OK to close the Event Log viewer.

Problems Displaying a Project

When running an ioDisplay project, you may encounter problems with how graphics appear on the monitor. (For example, alarm colors might not appear correctly, or might change as the project runs.) Bitmap graphics that you have imported into your project also might not appear correctly.

If these or similar errors occur, check the color depth of the monitor on which the ioDisplay project is running. If the color depth of the display is set to 256 colors (or "8-bit"), change the setting to 65,536 colors ("16-bit") or greater. The number of colors available depends on the video card installed in your PC, but most PCs can display at least 16-bit color.

Changing Monitor Color Depth

To change the color depth of your monitor in Windows NT, follow these steps:

1. From the Windows Start menu, choose Settings→Control Panel.

2. In the Control Panel window, double-click the Display icon.

3. Click the Settings tab in the Display Properties dialog box that opens.

4. In the Color Palette group, click the drop-down menu and select the number of colors that you want to use.

5. Click OK to save your settings and close the dialog box.

For additional information on changing monitor settings in Windows, see the documentation from Microsoft and your computer manufacturer.

Problems Saving a Project

When trying to save a project in ioDisplay Configurator, you may see an error message stating that no storage space is available on the computer's hard drive. If this message appears, yet you know that there is enough storage space for the ioDisplay project files, check to see if one or more files are marked "Read Only."

To do this, open the ioDisplay project folder in Windows Explorer, right-click on a project file, and select Properties from the pop-up menu that appears. If the file is marked Read Only, the check

box "Read Only" at the bottom of the Properties window is checked. Clear this check box, click OK to close the Properties window, and try saving the project again in ioDisplay Configurator.

For additional information on viewing file properties in Windows, see the documentation from Microsoft and your computer manufacturer.

Making an Empty String Visible

In an ioDisplay project, if a text string object in a display sends an empty string to a control engine, the text string object becomes invisible and can't be selected again. This might happen, for example, if an operator accidentally cleared a field while entering values in a display.

This problem occurs because the control engine's string variable is empty, so when the text string object linked to this variable checks the control engine, it has nothing to display.

To correct this condition, do the following:

1. Open the ioDisplay project.
2. Draw or import a graphic object to be used as a push button.

   The graphic should be approximately the same size as the text string object.
3. Double-click on the new graphic object and, in the Graphic Dynamic Attributes dialog box that opens, double-click "Send String" from the Operator Driven Attributes list.
4. In the Dynamic Attributes - Send String dialog box that appears, select "Prompt for Data" in the Source group and click OK.
5. Click OK to close the Graphic Dynamic Attributes dialog box and save your changes.
6. Now place the new graphic object behind the text string object as follows:
   a. Using the Select tool, select the graphic object and move it until it's directly over the text string object.
   b. With the graphic object still selected, choose Edit→Z-Order→Send to Back.
7. Save your project and run it in ioDisplay Runtime.

   If you click in the area of the text string object and new push button graphic object, the Enter String dialog box should appear, even if an empty string has been sent to the control engine.

Problems with Windows User Permissions

When you set up controllers on a computer running the Microsoft Windows NT or Windows 2000 operating system, typically you are using the computer with top-level "administrator" privileges. If someone later uses this same computer to run a FactoryFloor application, but logs in to the

computer with lower-level, non-administrator privileges, the FactoryFloor application may not recognize controllers that have been previously configured.

If this problem occurs, you can modify the Windows NT permissions to let specific users access previously configured controllers without having administrator access. This is done using the Registry Editor utility.

WARNING: *Use the Windows Registry Editor carefully. It is strongly recommended that you make a backup copy of your Windows Registry before continuing with this procedure. Without a backup copy, if you delete the wrong properties and cannot return the Registry to its original state, application and system files can become unusable and will have to be reinstalled.*

1. From the Windows Start menu, select Run.

   The Run dialog box appears.

2. Enter the following command in the Open field and press ENTER:

   ```
   regedt32
   ```

   The Registry Editor main window appears with several open windows inside it.

3. Select the HKEY_LOCAL_MACHINE window to make it active.

4. Double-click the Software folder in the HKEY_LOCAL_MACHINE window.

5. Select the Opto22 folder.

6. Select Security→Permissions.

   The Registry Key Permissions dialog box opens. Make sure that "Opto22" appears next to Registry Key at the top of the window.

7. Click Add.

8. In the Add Users and Groups dialog box, select the name of the appropriate group or domain from the List Names From drop-down list.

9. In the Names list, select the name of the user or group that will get controller access and then click Add.

10. If it is not already selected, choose "Full Control" from the Type of Access drop-down menu.

11. Click OK.

12. In the Registry Key Permissions dialog box, select the Replace Permission on Existing Subkeys checkbox and click OK.

13. Select Registry→Exit to close the Registry Editor.

14. Restart the computer.

The user or group you added can now use controllers without having administrator access.

Other Troubleshooting Tools

Checking ioProject File Versions

Sometimes problems may be caused by older or misplaced files. Product Support may ask you to run OptoVersion to check the versions and paths of your Opto 22 .DLL and .EXE files. Here's how:

1. From the Start menu, choose Programs→ioProject Software→Tools→OptoVersion.
2. In the OptoVersion window, click Find.

   The utility searches your hard drive and prints a list of Opto-related files found.
3. To see more information on any file, double-click its name. To sort the list in a different order, click any column heading.
4. To e-mail the information to Opto 22 Product Support, click E-mail.

   The utility saves the list to a file named Version.bd in the same directory that contains OptoVersion.exe. If you use Microsoft Outlook as your e-mail program, a new message automatically appears addressed to Product Support, with the version file attached.
5. If you use Microsoft Outlook, add comments to the new message and click Send.
6. If you use another e-mail program, attach the Version.bd file to an e-mail message and address the message to **support@opto22.com**, along with an explanation of the problem you're experiencing.

APPENDIX B

ioDisplay Errors

This appendix lists error messages you may see while running a project in ioDisplay Runtime. The cause of each error message is described, and, if possible, corrective action you can take to resolve the problem.

Types of Errors

While using the Configurator and Runtime components of ioDisplay, several types of errors may occur due to incorrect equipment setup, out-of-date files, or equipment failure. These errors generally fall into three categories:

- **Runtime Errors**—These may occur while running a project in ioDisplay Runtime. Most of these errors can be traced to control engine configuration problems or configuration problems with the I/O unit(s) connected to the control engine. Runtime errors can be further grouped into several subcategories based on the type of error that occurs; see "Error Messages in ioDisplay Runtime" on page B-2 for lists of error messages in each subcategory.
- **Configurator Errors**—These may occur as you use ioDisplay Configurator to create a project (for instance, adding a dynamic attribute to a graphic object). Errors most commonly occur when entering data into a dialog box; if an error occurs while doing this, simply re-enter an appropriate value and continue.
- **Windows Errors**—These may occur while using either the Runtime or Configurator components of ioDisplay. The most common Windows errors occur when too many applications are running at the same time, reducing the amount of memory available for the operating system. These errors are issued by the Microsoft Windows operating system running on your computer; see the documentation from Microsoft and your computer manufacturer for more information about Windows errors.

Error Messages in ioDisplay Runtime

Error messages that may appear in ioDisplay Runtime can be grouped into several categories, which appear in the table below. Messages for each category, as well as possible causes for the error and corrective actions to resolve it, are listed for each category.

NOTE: Many of these error messages also appear when running ioControl. For additional troubleshooting information, see Appendix A, "ioControl Troubleshooting," and Appendix B, "ioControl Errors," in the ioControl User's Guide.

File Access Errors

The File Access errors listed below are generated by ioDisplay if an error occurred while ioDisplay is working with files or with historic logs.

File access errors usually occur because ioDisplay is attempting to modify a file that is marked as Read Only. If a file access error occurs, first check the Windows file settings for the file being modified (or written to). Make sure that Read Only is *not* selected, and verify that the file has the correct access permissions for the operators who need to use the ioDisplay project.

Error Message	Possible Causes
Bad string, using default file name.	The tag name used as a source for a file name could not be used. The default file name will be used instead. The default file name was set up in the Configurator.
Can't make directory	A directory could not be created. Check if the directory is being created in a read-only directory. Change the protection to allow you to create it.
Directory now created.	Status message indicating that the directory was created.
Drive is full. Writing has been suspended.	A file was being written to a drive, but not enough free space was available to complete the transaction. Free up some space on the drive to continue.

Error Message	Possible Causes
Drive is no longer full. Writing has been resumed.	Status message indicating that the destination drive has enough space available to complete the file writing transaction.
Initial Writing of file: File name	Status message indicating that a file name is going to be created.
Invalid directory bad directory name. Using default.	The specified directory could not be used. The default directory will be used instead. The default directory was set up in the Configurator.

Launch Application Errors

The following error messages may appear if an error occurred while working with the Launch Applications feature of ioDisplay:

Error Message	Possible Causes
File not found	The executable file specified for a launch application setup could not be found. Verify the directory where the file actually resides.
Path not found	The drive/directory path specified for the executable in a launch application setup does not exist. Verify the actual path of the executable file.
WinExec error #	A Windows executable error occurred. Check Microsoft help sources for clarification about the error number.
Working directory invalid	The working directory specified for a launch application setup is incorrect. Verify the drive and path.

Port Errors

The following errors are generated by ioDisplay if a port-related error occurred:

Error Message	Possible Causes
Could not read value(s) from Registry	The registry could be corrupt. Contact Opto 22 Product Support.
Could not write new entry to Registry. Make sure you have administrative rights for the computer.	There was a problem writing information to the registry. Contact your network administrator.
Please reboot for changes in the Registry to take effect	Reboot your computer so that changes to the registry are activated.

Error Message	Possible Causes
Port setup failed	At Runtime, the ioDisplay software was unable to initialize the I/O driver. This is probably caused by a memory allocation problem. Exit Windows and restart Runtime.
WinRT: Mutual exclusion could not be created	Information could not be written to the registry. Try rebooting the PC, or contact Opto 22 Product Support.
WinRT: Mutual exclusion could not be deleted	This is a warning message. No action is required; the Windows operating system will take care of this later.
WinRT: Registry entry already exists	The control engine name and address already exist. Re-check the names of the configured control engines.
WinRT: Specified device could not be found in Registry	Configure the device again. Check port setup and addressing.

Recipe Upload/Download Errors

The following error messages are displayed if an error occurs while downloading or uploading a recipe:

Error Message	Possible Causes
Could not find the selected directory for the recipe destination file. Please check to ensure the path is correct. (Recipe Upload)	The path configured for the destination file of the uploaded recipe is invalid. Make sure the drive/directory path is correct.
Could not find the selected directory for the recipe format file. Please check to ensure the path is correct. (Recipe Upload)	The path configured for the format file of the uploaded recipe is invalid. Make sure the drive/directory path is correct.
Could not find the selected recipe format file. Please check to ensure the file name is correct.	The file name configured for the format file of the uploaded recipe was not found. Make sure you configured the correct tag name.
Could not make directory for Destination File: (Recipe Upload)	The path configured in the destination path could not be created. Check the drive specified in the path. Also check the read/write protection of the directory.
Invalid type specified. Valid types are: “Integer Table, Float Table, String Table, and Chart”	An invalid type was specified in the recipe. Only tags of types integer table, float table, string table, and chart are allowed in a recipe.
String for destination file was empty: (Recipe Upload)	An ioControl tag name was configured to contain the name of the destination file, but its contents were empty. Make sure you configure the correct tag name.

Error Message	Possible Causes
String for format file was empty	An ioControl tag name was configured to contain the name of the format file, but its contents were empty. Make sure you configured the correct tag name.
The recipe file does not exist! (Recipe Upload)	The recipe upload file does not exist. Verify the spelling of the file.
The specified chart state is invalid. Valid states are Run, Stop, Suspend, or Continue.	Make sure the chart state sent in a chart control instruction was Run, Stop, Suspend, or Continue.
The specified control engine does not exist in this project.	The control engine specified in the recipe's ioControl tag is not recognized by this project. Verify the control engine name for the tag name requested. Check the control engine's spelling.
The tag info is formatted incorrectly. Should be: "Control Engine:Tag Type.Tag Name"	The syntax for the ioControl tag is incorrect. Make sure it follows this pattern: Control Engine Name:Tag Type.Tagname, where Control Engine Name is the name of the control engine; Tag Type is "Integer Table," "Float Table," "String Table," or "Chart;" and Tagname is the name of an ioControl tag or chart name of the type specified in Tag Type.
The tag value is formatted incorrectly. Should be: "Index (optional): Value"	Check the syntax of the tag value(s) specified for the indices. Make sure a colon separates the index from the actual value, and also verify that the index is within the table's range.
Uploaded/Downloaded to File: file name	Status message indicating file name was uploaded or downloaded.

Scanner Errors

The ioDisplay Runtime error messages listed below are displayed if an error occurs with the scanner.

Error Message	Possible Causes
Non-specific	The value is bad but no specific reason is known.
Configuration Error	There is some server-specific problem with the configuration. For example, the item is question has been deleted from the configuration.
Awaiting Connection	The input is required to be logically connected to something but is not. This may reflect that no value is currently available, for reasons such as the value not having been provided by the data source.
Device Failure	The control engine has returned an Undefined Command error (-1).

Error Message	Possible Causes
Sensor Failure	A sensor failure had been detected (the Limits field can provide additional diagnostic information in some situations).
Last Known Value	Communications have failed, but the last known value is available. Note that the ‘age’ of the value may be determined from the TIMESTAMP in the OPCITEMSTATE.
Comm Failure	Communications have failed and no last known value is available.
Out of Service	The block is off scan or otherwise locked. This quality is also used when the active state of the item or the group containing the item is ‘InActive’.
Last Usable Value	Whatever was writing this value has stopped doing so. The returned value should be regarded as ‘stale’. Note that this differs from a bad value with Substatus 5 (Last Known Value). That status is associated specifically with a detectable communications error on a ‘fetched’ value. This error is associated with the failure of some external source to ‘put’ something into the value within an acceptable period of time. Note that the ‘age’ of the value can be determined from the TIMESTAMP in OPCITEMSTATE.
Sensor Not Accurate	Either the value has reached a sensor’s minimum or maximum limit (in which case the limit field should be set to 1 or 2), or the sensor is otherwise known to be out of calibration via some form of internal diagnostics (in which case the limit field should be 0).
Engineering Units Exceeded	The returned value is outside the limits defined for this parameter. Note that in this case (per the Fieldbus Specification) the ‘Limits’ field indicates which limit has been exceeded but does not necessarily imply that the value cannot move farther out of range.
Sub-Normal	The value is derived from multiple sources and has less than the required number of Good sources.
Local Override	The value has been Overridden. Typically this is means the input has been disconnected and a manually entered value has been 'forced'.

System Errors

The following error is displayed if a system error occurred:

Error Message	Possible Causes
System: Floating point error N caught by signal handler	The data returned from the control engine was detected to have a floating point error. This error could have occurred during data manipulations at the control engine. Verify that the data has been handled or cast properly according to its type.

APPENDIX C

ioDisplay Files

This appendix lists the files used in an ioDisplay project, including those created automatically when a project is saved. Use this information as a reference when you are looking through your ioDisplay files or working directory.

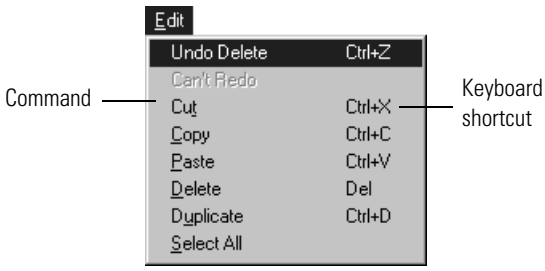
ioDisC.exe	ioDisplay Configurator executable program file.
ioDisR.exe	ioDisplay Runtime executable program file.
ioDsRX.exe	ioDisplay Runtime monitor-only version executable program file.
*.$$$	ASCII text file created by ioDisplay Configurator using the AutoCorrect Tags option. The file displays any changes made by the AutoCorrect Tags tools to tagnames in strategies that were incompatible with ioDisplay. The file also lists tagname errors that could not be corrected.
*.alm	Alarm log file.
*.bin	SuperTrend log file saved in binary format.
*.bmp	Bitmap file, created by other programs or by ioDisplay. Graphics saved as bitmaps from ioDisplayioTerminalioMessageViewer.idb are not saved with any dynamic attributes that may have been configured.
*.H##	Historic log file, created by ioDisplay Runtime.
*.idb	Main strategy file from an ioControl program. Lists all objects and other global information used in a program, as well as control engine configuration information.
*.ini	The ioDisplay initialization file created by ioDisplay Configurator. Microsoft Windows typically names the file type as a Configuration Settings file.
*.ixw	Exported window file created using ioDisplay Configurator. Contains all objects and tags in a specified window in an ioDisplay project.
*.uui	Main project file for an ioDisplay project.

*.msg	Event log file, created by ioDisplay Runtime.
*.rcp	Recipe file, created by a text editor. Used to send a set of parameters or to read a set of parameters from a control engine.
*.smb	Symbol file, created by ioDisplay Configurator. Symbol files contain graphic objects and their configured attributes for use in ioDisplay projects.
*.T##	SuperTrend historic log file, created by ioDisplay Runtime.
*.txt	Dynamic attribute (or “tag info”) report file, created by ioDisplay Configurator.
*.W##	Draw window file, generated by the ioDisplay Configurator.

APPENDIX D

ioDisplay Menu Reference

This appendix lists in detail the contents of ioDisplay menus for both Configurator and Runtime components. Note that if a keyboard shortcut is available for a menu command, the shortcut is listed next to the command in the menu as shown in the example below:



Many menu commands can also be accessed by right-clicking on an object or in a window, and then choosing the command from the pop-up menu that appears.

ioDisplay Configurator Menus

File Menu

New Project—Creates a new project. After selecting this menu option, choose a location and provide a name for the project in the dialog box that appears.

Open Project—Opens an existing project. After selecting this menu option, navigate to and select the project you want to open in the dialog box that appears. Only one ioDisplay project may be open at a time.

Close Project—Closes the project that is currently open. If the project has been modified, you are prompted to save changes.

Save Project—Saves any modifications to the files for the current project.

Save Project As—Saves any modifications to the current project to a name and project directory. This menu option is similar to Save Project, except that you can specify a new name and location for the saved project in the dialog box that appears. This is a good way to make a copy or a backup version of a project.

You can also copy an ioDisplay project to a different computer or drive without using the Save Project As menu option. To do this, create a directory on that computer or drive, then copy all the files from the original directory to the new directory using Windows Explorer. See Appendix C, "ioDisplay Files," for a complete list of files associated with an ioDisplay project.

Save Project and Load Runtime—Saves the current project and then opens it in ioDisplay Runtime. This is a quick way of switching between Configurator and Runtime when you are developing a project.

Project Path—Displays the full directory path to the project's saved location. The project's path is also displayed in the title bar, but if it is too long to fit there, you can use this command to see the directory path.

Password Protect Project—Lets you protect your ioDisplay project with a password to prevent others from opening and modifying the project using ioDisplay Configurator. The project can still be opened and run in ioDisplay Runtime.

Configurator Options—Sets a startup option that automatically opens the last project that was open the last time Configurator was run.

Choose Bitmap—Selects a bitmap file for use in the project. After selecting this menu option, in the dialog box that appears, navigate to and select the bitmap you want to include in the project. Use the Bitmap tool to place the selected bitmap in the project's draw window.

Save As Bitmap—Saves the selected graphic(s) as a bitmap. After selecting this menu option, in the dialog box that appears, specify a file name and location for the new bitmap. If no graphics are selected, then everything in the draw window is saved to the bitmap file name. Any dynamic attributes you have configured are not saved with the bitmap.

Printer Setup—Selects an available printer and sets its attributes.

Print—Prints the contents of any displayed main and draw windows. You can specify the number of copies to be printed and other options in the Printer Setup command.

(Previous File List)—Displays the names and directory paths of projects that had been previously opened in Configurator.

Exit—Closes the current Configurator windows and exits the application. If you modified the current project, you will be prompted to save it.

Edit Menu

Undo/Redo—(Available only when you have performed an action that can be undone or redone.) Reverses an earlier action you have performed, or repeats an action performed earlier. For example, if you have deleted a graphic object from a window, select Undo to restore the graphic. If, after restoring the graphic, you decide again to delete it, select Redo to repeat the earlier deletion. You can undo up to 50 actions.

Cut—(Available only when you have selected something.) Copies selected graphics onto the clipboard and removes them from the draw window. Cutting something replaces anything stored there previously.

Copy—(Available only when you have selected something.) Copies selected graphics onto the clipboard without removing them from the draw window. Copying something to the clipboard replaces anything stored there previously.

Paste—(Available only when something has been copied or cut into the clipboard.) Inserts a copy of the clipboard contents into the middle of the active draw window.

Delete—(Available only when you have selected something.) Removes selected graphic(s) from a draw window. Unlike the Cut command, Delete removes the selection without placing it in the clipboard; once you delete something, you cannot retrieve it.

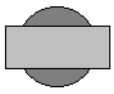
Duplicate—(Available only when you have selected something.) Creates a duplicate of the selected graphic(s). The duplicate is placed directly below the selected graphics. Duplicating a selected graphic does not use the clipboard.

Select All—Selects all the completed graphics in the active draw window. Anything that is not selected within the active draw window when you use this command may be incomplete. Incomplete graphics can be erased by using the Redraw command under the View menu.

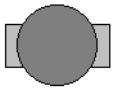
Replace—Modifies tagnames attached to a graphic or graphics. Allows you to link graphics to a different control engine, item name, table index, or bit index. You can find and replace tags in the entire project, or just in the selected graphic(s).

Z-Order—(Available only when you have selected something.) Positions selected graphics in front of or in back of other graphics. The following choices are available:

- **Bring to Front**—(Available only when you have selected something.) Positions the selected graphics in front of any other objects in the window.

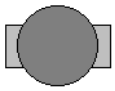


Before Bring to Front

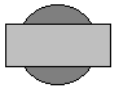


After Bring to Front

- **Send to Back**—(Available only when you have selected something.) Positions the selected graphics in back of any other objects in the window.



Before Send to Back



After Send to Back

Align—(Available only when you have selected more than one object.) Aligns selected objects in a variety of ways. The following choices are available:

- **Left**—Aligns the left edges of the selected graphics. All selected graphics are moved left to align with the left-most graphic in the group.

Before Left-Align After Left-Align

- **Center**—Aligns the vertical centers of the selected graphics. All selected graphics are moved left or right to align their centers with an imaginary vertical line down the center of the selected graphics.

Before Center-Align After Center-Align

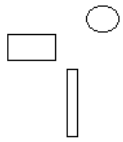
- **Right**—Aligns the right edges of the selected graphics. All selected graphics are moved right to align with the right-most graphic in the group.

Before Right-Align After Right-Align

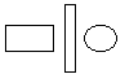
- **Top**—Aligns the top edges of the selected graphics. All selected graphics are moved up to align with the top-most graphic in the group.

Before Top-Align After Top-Align

- **Middle**—Aligns the horizontal centers of the selected graphics. All selected graphics are moved up or down to align their centers with an imaginary horizontal line running across the center of the selected graphics.

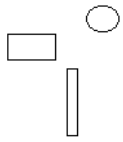


Before Middle-Align



After Middle-Align

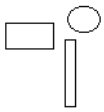
- **Bottom**—Aligns the bottom edges of the selected graphics. All selected graphics are moved down to align with the bottom-most graphic in the group.



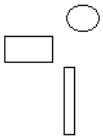
Before Bottom-Align

After Bottom-Align

- **Space Evenly Vertically**—Distributes the selected graphics so there is an equal amount of vertical space between each object.

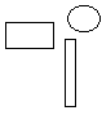


Before Space Evenly Vertically

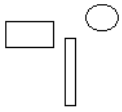


After Space Evenly Vertically

- **Space Evenly Horizontally**—Distributes the selected graphics so there is an equal amount of horizontal space between each object.



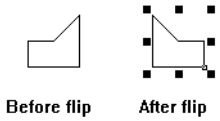
Before Space Evenly Horizontally



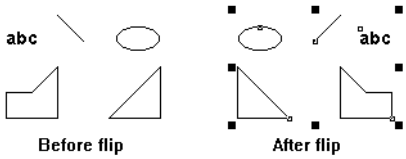
After Space Evenly Horizontally

Flip/Rotate—(Available only when you have selected one or more objects.) Changes the orientation and rotation of objects. The following choices are available:

- **Flip Horizontal**—Removes selected graphics and replaces them with mirror images of the graphics flipped over a vertical center point. Text, trends, bitmaps, and metafiles have their positions changed but are not mirrored.

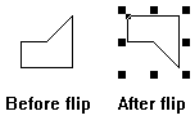


Horizontal Flip of a Single Graphic

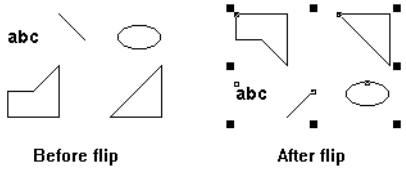


Horizontal Flip of Multiple Selected Graphics

- **Flip Vertical**—Removes selected graphics and replaces them with mirror images of the graphics flipped over a horizontal center point. Text, trends, bitmaps, and metafiles have their positions changed but are not mirrored.

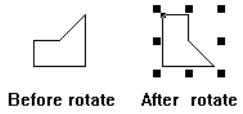


Vertical Flip of a Single Graphic

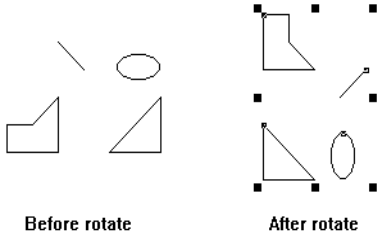


Vertical Flip of Multiple Selected Graphics

- **Rotate Clockwise**—Rotates the selected graphic 90 degrees clockwise. If more than one graphic is selected, the center of rotation is the center of the smallest rectangular area that contains all the graphics. Text, trends, bitmaps, and metafiles cannot be rotated, and any selection of multiple graphics that includes one of these cannot be rotated.

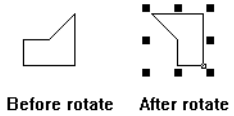


Clockwise Rotation of a Single Graphic

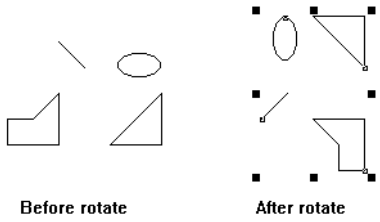


Clockwise Rotation of Multiple Selected Graphics

- **Rotate Counterclockwise**—Rotates the selected graphic 90 degrees counter-clockwise. If more than one graphic is selected, the center of rotation is the center of the smallest rectangular area that contains all the graphics. Text, trends, bitmaps, and metafiles cannot be rotated, and any selection of multiple graphics that includes one of these cannot be rotated.



Counterclockwise Rotation of a Single Graphic



Counterclockwise Rotation of Multiple Selected Graphics

Group—(Available only when you have selected more than one object.) Gathers any combination of two or more graphics into a single graphic object. You can then select the object, move it, size it, or assign dynamic attributes to it as a single entity. In Runtime, only the dynamic attributes assigned to the grouped object are processed; any dynamic attributes assigned to individual objects that make up the group are ignored, including trends.

Ungroup—(Available only when you have selected a grouped object.) Splits a graphics object on which the Group command has been used into its original individual components. This allows each graphic object to be individually selected. If any of the graphics had individual dynamic attributes prior to grouping, those dynamic attributes will be restored, and then configured and processed at Runtime.

Copy to File—(Available only when you have something selected.) Saves the selected object(s) to a file and saves any dynamic attributes you've assigned to the object(s). Specify the file name, location, and file format when prompted. The default file name extension is .smb.

Save Metafile As—(Available only when you have a metafile graphic selected.) Saves a selected Windows metafile graphic to a file. Specify the file name, location, and file format when prompted. You can save the selected metafile graphic to the Windows Metafile (WMF) format or to the Enhanced Metafile (EMF) format. Windows metafiles have the file name extensions .wmf and .emf.

Paste from File—Retrieves graphics from a file or from an included library of industrial graphics. You can select a graphic using the following menu commands:

- **Built-in symbols**—Select this to choose a graphic that has been saved as an ioDisplay symbol file. A dialog box prompts you for the file name, location, and file format of the file you'd like to open. (Symbol files have the file extension .smb.) Click the Open button to import the selected graphic.
- **Symbol Factory**—Select this to choose a graphic from a large library of graphics designed for industrial applications. These graphics are in Windows metafile (WMF) and other file formats. When the Symbol Factory window opens, browse through the categories and thumbnails provided to find an appropriate graphic, and then double-click the graphic to copy it to the Windows clipboard. (Another way to copy a graphic in the Symbol Factory is to select it and click the Copy button.)

  Now click on the ioDisplay project draw window to make it active, and then select Paste from the Edit menu to add the copied graphic to the window. (You can also paste the graphic by pressing CTRL+V on the keyboard, or by right-clicking and selecting Paste from the pop-up menu.)
- **Import metafile**—Select this to choose a graphic that has been saved in either WMF or EMF (Enhanced Metafile) format. A dialog box prompts you for the file name, location, and file format of the file you'd like to open. (Metafiles have the file extensions .wmf and .emf.) Click the Open button to import the selected graphic.You are asked to navigate to and select an appropriate file.

Edit Dynamic Attributes—(Available only when you have something selected.) Connects a graphic to an ioControl data item. After selecting this menu option, all applicable dynamic attributes are shown in the dialog box that opens. Select input dynamic attributes or output dynamic attributes when prompted. For example, you can set up connections so the value of a ioControl tag changes the color and fill size of a graphic. With output dynamic attributes, you can change the value of a tag as you "slide" the graphic on the screen.

Many different combinations of dynamic attributes are possible, and different dialog boxes are used to assign dynamic attributes to a graphic. For example, when a trend is selected, the Trend Configuration dialog box is displayed.

Copy Dynamic Attributes—(Available only when you have a single graphic selected and a control engine is configured.) Creates and stores in the clipboard a copy of the selected graphic's dynamic attributes.

Paste Dynamic Attributes—(Available only when you have something selected and have previously copied dynamic attributes to the clipboard.) Assigns copied dynamic attributes to a graphic. You can paste dynamic attributes to one or more selected graphics. You can delete existing attributes, or replace or ignore any duplicate attributes.

Delete Dynamic Attributes—(Available only when you have something selected.) Removes dynamic attributes of a selected graphic. You can delete the dynamic attributes of more than one selected graphic.

Edit Text—(Available only when you have a text object selected.) Changes text in a text object. Select the text with the Select tool, choose this menu item, and then edit the text in the dialog box that appears.

Edit Points—(Available only when you have a polygon, polyline, or Bezier curve selected.) Changes individual points in a polygon, polyline, or Bezier curve. Select the object with the Select tool, choose this menu item, and then click and move individual points on the object.

Lock/Unlock Position—(Available only when you have something selected.) Locks the position of one or more items in a draw window.

View Menu

Hide Menu Bar—Hides the menu bar. The ESC key toggles the menu bar on and off.

Hide/Show Toolbox—Hides or displays the Toolbox. The Toolbox shows the tools you need to create a project in ioDisplay Configurator. If the Toolbox is hidden, the Show Toolbox command is displayed in this menu.

Configure Grid—Displays a Grid dialog box that prompts you to toggle on or off both the Grid and the Snap On feature. Grids can aid your work in the draw window. You can also enter a Grid size in the Grains/Units area of the dialog box. The Grid size refers to the spacing of Grid points, measured in pixels. For example, a Grid size of 10 means a grid point will appear every 10 pixels. Sometimes grids do not appear because the grid size is too big for the draw window. You cannot display the Grid without specifying a Grid size first.

Hide/Show Grid—(The Grid must first be displayed using the Grids menu item.) Hides or shows the Grid. If the Grid is hidden, the Show Grid command is displayed.

Turn Snap On/Off—(The Grid must first be displayed using the Grids menu item. Also activate the Snap On feature in the same dialog box.) Toggles the Snap On feature on or off. Snap On cannot work without an active Grid. If Snap On is enabled, the Turn Snap Off command is shown.

Redraw—Redraws the contents of the active draw window. Incomplete graphics (such as an incomplete polygon) in the draw window are removed when you select this command.

Dynamic Attributes—Generates a text file listing the dynamic attributes of objects in one or more draw windows. This report also lists the configured alarm points in the ioDisplay project.

Launch TagInfoView Utility—Starts a small utility program that lets you sort the order in which tags in the ioDisplay project are displayed.

Find Tag—Used to locate specific tags used in an ioDisplay project.

Decrypt Runtime Operator Log File—Starts a small utility program that decrypts encrypted Runtime Operator Log files.

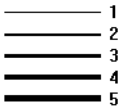
Style Menu

Use the Style Menu to control the drawing attributes of the graphic tools. Whenever a graphic such as a line or rectangle is drawn, the selected style attributes are applied. Combining different style settings allows you to draw an almost infinite variety of graphics. Text attributes, including font style, color, and size, are assigned in the Text menu.

NOTE: Trends and bitmaps are not affected by style settings.

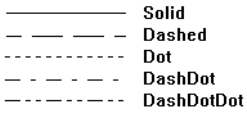
Line Color—Presents a color palette you can use to assign or change the line color of the selected graphic. If no graphic is selected, the color you choose is set as the default and is then applied as the line color to all graphics you subsequently draw.

Line Width—Assigns or changes the line width of the selected graphic. Line widths are shown in pixels. If no graphic is selected, the line width you choose is set as the default and is then applied to all graphics you subsequently draw.

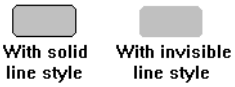


Sample Line Widths

Line Style—Assigns or changes the line style of the selected graphic. If no graphic is selected, the line style you choose is applied to all graphics you subsequently draw. Line styles other than solid apply only to objects with a line width of 1. Line widths greater than 1 are always solid. The Invisible line style is used with rectangles, round rectangles, ellipses, and polygons. If these objects are drawn with the invisible line style, the border line around the object is not displayed; in order to see them, you must apply a fill.



Sample Line Styles

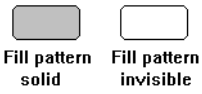


Example of Invisible Line Style

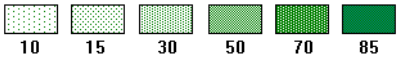
Fill Color—Assigns or changes the fill color of the selected graphic. If no graphic is selected, the fill color you choose is applied to all graphics you subsequently draw. Fill colors only affect rectangles, round rectangles, ellipses, and polygons.

Background Color—Assigns or changes the color used behind the fill pattern of the selected graphic. If no graphic is selected, the background color you choose is applied to the fill pattern of all graphics you subsequently draw. Background colors for fill patterns only affect rectangles, round rectangles, ellipses, and polygons.

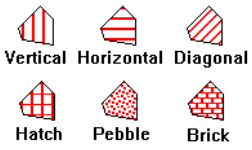
Fill Pattern—Assigns or changes the fill pattern of the selected graphic. If no graphic is selected, the fill pattern you choose is applied to all graphics you subsequently draw. Fill patterns only affect rectangles, round rectangles, ellipses, and polygons. If you select a Fill Pattern and your selected objects still remain unfilled, it may be that the Fill Color is set to white (or a color equal to its background), or that the Fill Pattern is set to Invisible. Also, you cannot apply more than one Fill Pattern to any graphic. This includes using a Percent fill, which fills an object with a percentage of black, creating levels of gray.



Example of Solid and Invisible Fill



Example of Percent Fills



Example Fill Patterns

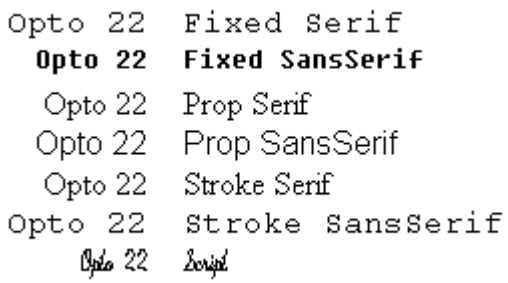
Opaque—Determines how non-solid primary graphics, such as dotted and dashed lines, interact with overlapping graphics and background colors. When the opaque style is set, overlapped graphics and background colors are overwritten.

Transparent—The opposite of Opaque, Transparent also determines how non-solid primary graphics, such as dotted and dashed lines, interact with overlapping graphics and background colors. When the transparent style is set, overlapped graphics and background colors are overwritten only by the solid portion of the line.

Text Menu

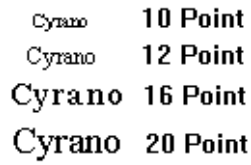
The Text menu items allow you to control text attributes. Text attributes may be set before the text is placed in the draw window, or changed after it is placed. Style attributes selected from the Style menu do not affect text.

Font—Assigns or changes the font of the selected text. If no text is selected, the font you choose is set as the default and is then used for all text you subsequently place. ioDisplay supports all TrueType fonts, as well as the ones shown below:



Examples of Supported Fonts

Size—Assigns or changes the size of the selected text. If no text is selected, the size you choose is set as the default and is then used for all text you subsequently place. Specify the size of the text in points; any value between 5 and 500 may be used.



Examples of Prop Serif Font at Different Point Sizes

Color—Assigns or changes the color of the selected text. If no text is selected, the color you choose is applied to all text you subsequently place. ioDisplay also supports custom color creation.

Background—Assigns or changes the background color of the selected text. If no text is selected, the background color you choose is applied to all text you subsequently place. Background colors only apply to opaque text; transparent text is not affected by this setting.

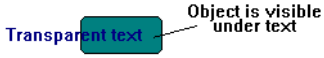
Text Style—Assigns or changes the style of the selected text. If no text is selected, the style you choose is applied to all text you subsequently place. ioDisplay supports normal, bold, italics, underline, and strikeout text styles. You can apply multiple styles to text. For instance, you can apply both bold and italics to any text.

Examples of Text Style

Opaque—Determines how text appears when overlapping other objects. Opaque style is applied to text just like any other style. If the text is set to Opaque, objects under the text will be overwritten by the text background color.

Example of Opaque Text

Transparent—Determines how text appears when overlapping other objects. The opposite of opaque, if the text is set to transparent, objects under the text will remain visible and unaffected by the text background color.



Example of Transparent Text

Configure Menu

Control Engine(s)—Selects which ioControl strategy (or strategies) are used for this project. The ioDisplay Configurator uses the information from the strategy to connect the appropriate ioControl data item to the dynamic attribute of a dynamic object. The control engine Properties dialog box prompts for the ioControl strategy. If an ioControl strategy is not configured for this project, dynamic attributes cannot be assigned to any dynamic objects.

Refresh Times—Changes the scan time for a refresh time group. The Refresh Time dialog box prompts you for the new times. See Chapter 6, "Working with Graphics," for more information about configuring refresh times.

Alarm Points—Displays all configured alarm points by user-specified names. This dialog box also allows you to add, modify, or delete alarm points. See Chapter 9, "Configuring Trigger-Based Events," for more information about alarm points.

Alarming Setup—Configures alarming features. The Options page sets up various Runtime options, the Logging page sets up file and printer logging, and the Sound page sets up sound functions. See Chapter 9, "Configuring Trigger-Based Events," for more information.

Historic Data Log—Creates historic logs. A Historic Log List dialog box is displayed and lists the historic log files that have been created. This command allows you to modify which points are recorded and how frequently data is logged to the files. See Chapter 9, "Configuring Trigger-Based Events," for more information about historic data logs.

Event Log—Records a message caused by an event to a disk file. You can change parameters such as the number of messages saved, the delimiter used between messages, and the file rollover period. File name extensions are .msg. The number of files retained on disk for an event log is also set within the Event Log File Configuration dialog box. When the limit is reached during Runtime, the file with the oldest time stamp is deleted. See Chapter 10, "Using ioDisplay Runtime," for more information about event logs.

SuperTrend Remote Logging—If the same ioDisplay project is running on more than one computer, this menu item selects the local or networked computer that will save SuperTrend data. See "Working with SuperTrends" on page 8-7 for more information about configuring SuperTrends and collecting data from them.

Applications—Adds or modifies application managers for use in the project. The Application Manager List dialog box displays available application managers for the project. The Application Manager Configuration dialog box lets you select the program file, working directory, launch options, initial display view, and trigger for the application manager. See Chapter 9, "Configuring Trigger-Based Events," for more information about launching applications based on a trigger.

Sounds—Selects sounds and assigns their trigger for use in the project. The Sounds dialog box lists the available sounds for the project. The Sound Configuration dialog box lets you configure start and stop triggers with the Trigger dialog box. The Sound Configuration dialog box also prompts you for the sound file. See Chapter 9, "Configuring Trigger-Based Events," for more information about triggering sounds.

Window State—Adds or makes changes to existing window managers. The Window Manager List dialog box displays all currently configured window managers and allows access to the Window Manager Configuration dialog box. The Window Manager Configuration dialog box allows you to change triggers with the Window Manager Start Trigger dialog box and control the draw window visual state with the Pop Window dialog box. See Chapter 9, "Configuring Trigger-Based Events," for more information about trigger-based window states.

Recipes—Configures download or upload of recipes to a control engine by a trigger. This method of recipe management does not require a graphic to be selected during Runtime for the recipe action to occur. See Chapter 9, "Configuring Trigger-Based Events," for more information about trigger-based recipe uploads and downloads.

Scanner Location—Sets the local or remote computer used to run ioDisplay's scanner engine.

Runtime—Defines the initial setup of the draw windows at Runtime. For example, you may want to have certain draw windows pop up and have others iconified when the project starts. This command can also be used to prevent the user from exiting ioDisplay Runtime when this

project is loaded. See Chapter 10, "Using ioDisplay Runtime," for more information about configuring the Runtime session.

Tools Menu

AutoCorrect Tags—Run this tool when you first configure a strategy for your project. The tool verifies tagnames from a strategy for compatibility with ioDisplay and changes the tagnames where necessary. The tool creates a text file with the file extension .$$$ that summarizes the changes that were made by the tool to any tagnames. Refer to "Correcting Tags from a Strategy" on page 5-18 for more information about this command.

Window Menu

ioDisplay Configurator allows you to configure several draw windows per project. The Window menu items control the number and properties of each draw window in a project. You can create draw windows, delete draw windows, copy draw windows, and change properties of existing draw windows.

New—Creates a new draw window and adds it to your project. You can specify the new window's name, size, behavior, and other attributes. You must provide a unique name for each draw window.

Open—Opens draw windows that are configured but are currently closed. Select which window to open from a list of all draw windows that are closed. Draw windows that are open when a project is saved are open when the project is started at Runtime. This can be modified using the Configure?Runtime menu command.

Close—Closes draw windows that are currently open. Select which window to close from a list of all draw windows that are open. Draw windows that are closed when a project is saved are closed when the project is started at Runtime. This can be modified by using the Configure➞Runtime menu command.

Copy—Duplicates the active draw window. You must enter a unique name for the duplicate and specify any properties you wish to change. All graphics and their connections in the copied window remain the same in the new copy.

Delete—Removes the active draw window from the project. All graphics and their connections in the active draw window are also deleted. Use caution since deleted draw windows cannot be recovered.

Properties—Modifies the properties of the active draw window. You can change the window's name, size, behavior, position, color, and other attributes.

Export/Import Window—An ioDisplay window can be exported from one project, saved as a file, and then imported into another ioDisplay project. The exported window file contains all the objects and tags that were in the original window. Exporting and importing draw windows is a convenient way to reuse the same window in different ioDisplay projects.

(Open Window List)—Displays the names of up to nine currently opened or iconified windows. Select a draw window's name from this list to display that draw window and bring it to the front.

If more than nine windows are open, a menu item named More Windows is added, which you can use to select a window's name from a list to display it. A window must be open or iconified to be listed.

Help Menu

Contents and Index—Starts Help and displays help topics for ioDisplay Configurator.

Manuals—Opens the online version of the printed *ioDisplay User's Guide*. This document is in Adobe Acrobat format, and the Acrobat Reader application is required to view it.

Opto 22 on the Web—Lists useful links to information on the Opto 22 Web site. Your PC must have an installed Web browser and be connected to the Internet to access these links.

About ioDisplay Configurator—Displays version information about ioDisplay Configurator.

ioDisplay Runtime Menus

Runtime menus provide access to the Runtime commands. These commands allow you to open and close projects, view the event log, and view the control engine configurations.

File Menu

Open Project—Loads an existing project created in ioDisplay Configurator. You must navigate to and select the project you want to open. Scanning and animation begin immediately once the project is loaded.

Project Path—Displays the project's full directory path. You can also see the project's path displayed in the title bar, but if it's too long to fit there, you can use this command.

Printer Setup—Selects an available printer and sets its attributes.

Print—Prints the ioDisplay Runtime window. You can specify the number of copies to be printed and also set up the options available in the Printer Setup command.

Exit ioDisplay Runtime—Stops the scanner, closes all Runtime windows, and exits the ioDisplay Runtime.

View Menu

Hide Menu Bar—Hides the menu bar. The ESC key toggles the menu bar on and off.

Control Engine(s)—Displays a list of all of the control engines that are configured for the project. See Chapter 10, "Using ioDisplay Runtime," for more information about Runtime control engine status.

OptoCom—Displays the OptoCom.DLL version and path.

Event Log—Displays the Runtime system event log. This log contains system errors and messages received during Runtime. The list box contains the most recent system event

messages generated by ioDisplay. Each message consists of a date and time stamp, and message text. The message text describes events such as communications and I/O errors. Use the scroll bar to view prior messages. If the text of the message is too wide to completely fit in the list box area, you can double-click the message to display it all.

Alarm Menu

Modify Alarm Points—Allows the operator to change parameters for alarm points in the ioDisplay project if it has been configured to let the operator do so. See "Setting up Runtime" on page 10-2 for more information about configuring Runtime options for the operator. Also see "Configuring Alarm Points" on page 9-28 to learn more about configuring alarm points.

Alarm point settings can be changed in each one of the five alarm ranges available: HiHi, Hi, Normal, Lo, and LoLo. Each alarm point state has a value that defines its range.

Alarms Enabled—Disables alarming, including all alarm graphics, sound, and logging. This can be useful when starting or stopping a process during which alarm conditions may be expected to happen. This menu item can be initially enabled or disabled through the Alarming Setup dialog box in the Configurator. It can also be permanently grayed out and made inaccessible by unchecking the Alarms Enabled menu item option in the Alarming Setup dialog box.

Priority Filter—The priority filter level menu items can be used to accept only those alarm points with a certain priority level. For example, during a startup or shutdown procedure, you may wish to receive only the most serious alarms. The priority for each alarm point is configured in the Alarm Point dialog box in the Configurator.

Window Menu

Open—Opens any Runtime window that is currently closed. An Open Window dialog box displays a list of closed windows from which you can select the window to open.

Close—Closes any currently open or iconified Runtime window. A Close Window dialog box displays a list of currently open windows from which you can select the window to close.

Switch Control Engines—Use to connect to different control engines running the same ioControl strategy. Select one or more project windows and then select a control engine. All objects with dynamic attributes will now use tag values from that control engine. See "Working with Control Engines" on page 10-15 for more information.

Open Window List—Displays currently open or iconified windows. Up to nine window names are displayed. Select a window's name from this list to display that draw window and bring it to the front. If more than nine windows are open, a menu item named More Windows is added; use More Windows to select a window's name from a list to display it. A window must be open or iconified for it to be listed.

Help Menu

Contents and Index—Starts Help and displays help topics for ioDisplay Runtime.

Manuals—Opens the online version of the printed *ioDisplay User's Guide*. This document is in Adobe Acrobat format, and the Acrobat Reader application is required to view it.

Opto 22 on the Web—Lists useful links to information on the Opto 22 Web site. Your PC must have an installed Web browser and be connected to the Internet to access these links.

About ioDisplay Runtime—Displays version information about ioDisplay Runtime.

ioDisplay Index

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T