

Lookout™ Direct

Developer's Manual



License Agreement

“Valued Technology”

This License Agreement is your proof of license. Please treat it as valuable property. This is a legal agreement between you (either an individual or entity) the end-user of this product, and Automation Direct. If you do not agree to the terms of this Agreement, promptly return the package and any accompanying items to Automation Direct for a full refund.

LookoutDirect SOFTWARE LICENSE

1. **GRANT OF LICENSE** — This LookoutDirect License Agreement (“License”) permits you to use one copy of the specified version of the LookoutDirect software product (“Software”) on any single computer, provided the Software is in use on only one computer at any time. If you have multiple Licenses for the Software, then at any time you may have as many copies of the Software in use as you have Licenses. The Software is deemed “in use” on a computer when it is loaded into the temporary memory (i.e. RAM) or installed into the permanent memory (i.e. hard disk, CD--ROM or other storage device) of that computer, except that a copy may be installed on a network server for the sole purpose of distribution to other computers that are not currently “in use”. If the anticipated number of Software users will exceed the number of applicable Licenses, then you must have a reasonable mechanism or process in place to assure that the number of persons using the Software concurrently does not exceed the number of Licenses. If the Software is permanently installed on the hard disk or other storage device of a computer (other than a network server) and one person uses that computer more than 80% of the time it is in use, then that person may also use the software on a single portable computer or a single computer at home.
2. **COPYRIGHT** — The Software is owned by Automation Direct or its suppliers and is protected by United States copyright laws and international treaty provisions. Therefore, you must treat the Software like any other copyrighted material (e.g., a book or musical recording) except that you may:
 - a. Make one backup copy of the Software solely for backup or archival purposes.
 - b. Transfer the Software to a single hard disk provided you keep the original and the backup solely for archival purposes. You may not copy any written materials that may accompany the Software.
3. **OTHER RESTRICTIONS** — This LookoutDirect License Agreement is your proof of license to exercise the rights granted herein and must be retained by you. You may not rent or lease the Software, but you may transfer your rights under this LookoutDirect License Agreement on a permanent basis provided that you transfer this License Agreement, the Software, and all accompanying written materials and retain no copies. The recipient must also agree to unequivocally abide by the terms contained in this License Agreement. You may not reverse engineer, decompile, or disassemble the Software. Any transfer of the Software must include the most recent update and all prior versions. If the Software is acquired within the United States, you may not export the Software outside of the United States without first complying with all applicable US export laws and regulations. You acknowledge that the Software will not function without a certain hardware key. This hardware key will be furnished to you by Automation Direct and you agree that such hardware key is to be used solely with the Software provided.

DISTRIBUTION LIMITATIONS

If you have acquired the development/runtime package, you may distribute process files created with the Software, provided that:

1. each recipient of your process file has a valid license for a separate runtime copy of the Software;
2. you include the following copyright notice, either on--screen in your process file’s About Box or written documentation, with each distributed copy of your

3. process file: "Copyright E [year] Automation Direct by Koyo, Inc. (Based on materials of National Instruments Corporation). All Rights Reserved";
4. you do not use Automation Direct's or National Instruments' ("NI") names, logos, or trademarks to market your process file without written permission; and
5. you agree to indemnify, hold harmless, and defend Automation Direct and NI (including their officers, directors, employees, and agents) and their suppliers from and against any claims.

LIMITED WARRANTY

1. Automation Direct warrants the Software will perform substantially in accordance with the written materials for a period of ninety (90) days from the date of receipt.

2. Automation Direct warrants that any hardware accompanying the Software will be free from defects in materials and workmanship under normal use and service for a period of one (1) year from the date of receipt.

NO OTHER WARRANTIES. EXCEPT AS EXPRESSLY SET FORTH ABOVE, THE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, AND NO OTHER WARRANTIES, EITHER EXPRESSED OR IMPLIED ARE MADE WITH RESPECT TO THE SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT, OR ANY OTHER WARRANTIES THAT MAY ARISE FROM USAGE OR TRADE OR COURSE OF DEALING. Automation Direct AND ITS SUPPLIERS DO NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE OF OR THE RESULTS OF THE USE OF THE SOFTWARE IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE AND DO NOT WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE

UNINTERRUPTED OR ERROR FREE. Automation Direct AND ITS SUPPLIERS EXPRESSLY DISCLAIM ANY WARRANTIES NOT STATED HEREIN.

Customer Remedies — Automation Direct's entire liability and your exclusive remedy shall be at Automation Direct's option. Automation Direct will either refund the price paid, or repair or replace the Software or hardware that does not meet Automation Direct's Limited Warranty. You must return the product to Automation Direct with a copy of your purchase receipt. This Limited Warranty is void if failure of the Software or hardware has resulted from accident, abuse, or misapplication. Any replacement Software will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

No Other Warranties — Automation Direct DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE, AND ANY ACCOMPANYING WRITTEN MATERIALS OR HARDWARE. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE TO STATE.

No Liability for Consequential Damages — In no event shall Automation Direct or its suppliers be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use of or inability to use this Automation Direct product, even if Automation Direct or its suppliers have been advised of possibility of such damages.

U.S. GOVERNMENT RESTRICTED RIGHTS

The Software and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c) (1) (2) of the Commercial Computer Software -- Restricted Rights at 48 CFR

52.227--19 as applicable. Contractor/manufacturer is Automation Direct by Koyo, Inc. / 3505 Hutchinson Road, Cumming, GA 30040 (under license from National Instruments Corporation, 11500 N. Mopac Expressway, Austin, Texas 78759--3504).

This agreement is governed by the laws of the State of Georgia.

WARNING

- (1) THE SOFTWARE IS NOT DESIGNED WITH COMPONENTS AND TESTING FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR INCONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.
- (2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTEROPERATINGSYSTEMSOFTWARE FITNESS, FITNESS OF COMPILERS ANDDEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED "SYSTEM FAILURES"). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE RELIANT SOLELY UPON ONE FORM OF ELECTRONIC SYSTEM DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK--UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END--USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM UTILIZED TESTING PLATFORMS AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE THE SOFTWARE IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY PLCDIRECT OR ITS SUPPLIERS, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF THE SOFTWARE WHENEVER THE SOFTWARE IS INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

Lookout is a registered trademark of National Instruments Corporation.

Lookout™Direct

Developer's Manual



Please include the Manual Name and the Manual Issue, both shown below, when communicating with Technical Support regarding this publication.

Manual Name: LookoutDirect Developer's Manual
Issue: First Edition, Rev. A
Issue Date: 11/02

Publication History

Issue	Date	Description of Changes
First Edition	8/01	Orginal
Rev. A	11/02	Deleted Chapter 11 and updated index

Contents

About This Manual

How to Use This Manual Set.....	vii
Document Conventions.....	vii
Technical Support	viii
Phone	viii
World Wide Web.....	viii

Chapter 1 Expressions

Creating Expressions	1-3
Data Polymorphism	1-3
Path Names in LookoutDirect	1-6
Expressions on Control Panels	1-8
Expression Objects	1-10
Expressions as Parameters.....	1-11
Expressions as Connections	1-12
Edit/Insert Expression Dialog Box	1-12
Expression Syntax.....	1-14
White Space.....	1-14
Arithmetic Operators.....	1-15
Text Operator.....	1-15
Comparison Operators.....	1-15
Expression Functions	1-18
Logical Functions	1-18
Lookup Functions.....	1-20
Mathematical Functions	1-21
Statistical Functions.....	1-24
Text Functions	1-26
Trigonometric Functions	1-29
Date/Time Functions	1-31
Quality Functions	1-31

Chapter 2 Graphics

Setting Snap to Grid.....	2-1
Static Graphics.....	2-2
Displaying Text, Plates, Insets, Rectangles, and Lines	2-2
Displaying Static Custom Graphics.....	2-4

Contents

Dynamic Graphics.....	.2-6
Displaying Dynamic Logical Signals2-7
Displaying Dynamic Numeric Signals2-10
Displaying Dynamic Text Signals2-12
Finding a Lost Graphics File.....	.2-13
Creating Custom Graphics2-14
Creating Custom Graphics Example2-14
Creating the Graphic2-14
.....	.2-16
Save or Export the Graphic and Place in LookoutDirect2-16
Testing the Graphic in LookoutDirect2-16
Graphic File Types.....	.2-18
Bitmaps2-18
Metafiles2-18
Bitmaps or Metafiles?.....	.2-18
Memory Considerations.....	.2-19

Chapter 3

Serial Port Communication Service

Introduction to Driver Objects3-1
Understanding the Communication Service3-2
Defining Serial Port Settings.....	.3-3
Selecting the Serial Port.....	.3-3
Setting Receive Gap3-3
Selecting the Serial Connection.....	.3-4
Hardwired Settings3-4
RTS/CTS Handshaking Settings3-4
Dial-Up Modem Settings3-5
Serial Port Hangup.....	.3-7
Serial Port Diagnostics.....	.3-7

Chapter 4

Networking

Registering Computers.....	.4-2
Logging Data.....	.4-3
Time Synchronization.....	.4-3
Checking the Network Connection between Two Computers.....	.4-6

Chapter 5

Dynamic Data Exchange

Linking LookoutDirect to Other Applications.....	.5-2
DDE Server Example5-2

DDE Client Example	5-3
DDE Peer-to-Peer Example.....	5-4
DDE Alarms	5-5

Chapter 6 Security

Logging On	6-2
User Manager.....	6-4
Creating User Accounts.....	6-5
Creating Groups.....	6-6
Modifying Users and Groups	6-7
Special Users and Groups.....	6-8
Control Security	6-9
Viewing Security	6-10
Control Panels	6-10
Controllable Objects.....	6-10
System Security Settings	6-11
Network Security	6-12
Configuring Security for Processes and Objects.....	6-12
Permissions	6-12
Advanced Security	6-15
Keeping Security Precedence Simple	6-18
Process File Edit Security	6-19
Action Verification	6-20
Importing Old Security Files into LookoutDirect 4.....	6-21

Chapter 7 Logging Data and Events

Spreadsheet Logger.....	7-1
Data Location	7-2
CSV Files.....	7-3
File and Disk Errors.....	7-3
Concurrent File Access.....	7-4
Information Overload	7-4
Citadel Historical Database	7-5
Default Data Location	7-5
Process Data Location	7-6
Creating a Historical Database	7-7
Logging Criteria.....	7-9
Alarm Information Overload.....	7-9
Event Logger.....	7-10
Event Data Location	7-10
Event Information Overload.....	7-10

Contents

Report Generation	7-11
Control Panel Reports.....	7-11
Third-Party Reports	7-13

Chapter 8 **Structured Query Language**

Introduction.....	8-1
What is ODBC?	8-1
What is SQL?.....	8-1
Creating a Citadel ODBC Data Source	8-2
Accessing Citadel Data	8-5
Traces Table.....	8-5
Points Table	8-6
Data Transforms	8-6
SQL Examples	8-7
Accessing Citadel Data with Microsoft Query	8-8
Accessing Citadel Data with Microsoft Excel.....	8-13
Accessing Citadel Data with Microsoft Access.....	8-13
Accessing Citadel Data with Visual Basic	8-17

Chapter 9 **Alarms and Events**

Defining Alarm Conditions.....	9-1
Database-Generated Alarms	9-1
Alarm Objects	9-2
The Alarm Subsystem	9-4
Selecting Processes to Monitor for Alarms	9-4
Alarm Areas.....	9-4
Alarm Priorities	9-6
Alarm Window	9-6
Alarm Display Options	9-7
Alarm Filters.....	9-8
Alarm Print	9-10
Alarm Acknowledgment.....	9-11
Acknowledging Alarms Programatically	9-13

Chapter 10 **Redundancy**

Standby Basics	10-2
Failover Scenarios	10-3
Configuring Standby	10-4
Enabling File Sharing in Windows 98/95 and Windows NT	10-4

Time Synchronization with Standby Computers.....	10-5
Implementing Redundancy	10-6
Redundancy Overview	10-6
Setting up the Primary Process and Computer	10-6
Setting up the Standby Process and Computer.....	10-9
Configuring Clients for Standby Operation	10-11

Chapter 11

Editing Object Databases

Editing Database Parameters	11-1
Numeric Member Parameters.....	11-4
Logical Member Parameters.....	11-7
Text Member Parameters	11-8
Importing and Exporting Object Databases.....	11-9
Exporting an Object Database	11-9
Creating a Database Spreadsheet.....	11-11
Importing an Object Database	11-13
Copying an Object Database	11-15

Glossary

Appendix A

Networking With DDE

NetDDE Networking Considerations	A-2
Multilink NetDDE Networking	A-3
Linking Controllable Objects	A-3
Linking Controllable Objects Together Across a Network	A-3
Linking Non-Controllable Objects	A-4
To Access Real-Time Data at Another Computer	A-4
Table Networking	A-6
Hardware Networking.....	A-10
Multilink and Table Networking Comparison	A-11
Setting up Networking with DDE.....	A-12
Running NETDDE.EXE Automatically.....	A-12
Windows 98/95	A-12
Windows NT	A-12
Adding a Trusted DDE Share.....	A-13

Appendix B

CBL Compiler

CBL Compiler Error Messages.....	B-2
----------------------------------	-----

Contents

Appendix C Lookout.INI File

Index

About This Manual

How to Use This Manual Set

The *Getting Started* guide contains installation instructions and a basic introduction to LookoutDirect features and functionality. It includes general information to help acquaint you with the important elements of LookoutDirect along with tutorial exercises to introduce you to the basics of building a LookoutDirect process.

The *Object Reference Manual* is in Portable Document Format (PDF) and describes the LookoutDirect object set. It is available on the LookoutDirect CD in the documentation directory. The LookoutDirect installation gives you the option of copying the PDF files into the LookoutDirect/documentation folder on your hard drive. To view these files, you must have Adobe Acrobat Reader 3.0 or later installed. If you do not have Adobe Acrobat Reader installed, you can install it from the LookoutDirect/documentation directory or from the Adobe web site at www.adobe.com.

This *Developer's Manual* explains how to create control processes for your HMI/SCADA applications. This manual includes detailed explanations of various LookoutDirect features, functions, and services.

Document Conventions

The following document conventions are used in this manual.

	Indicate the path for nested menu and command selections. Example: Start » Programs » xxx
	Indicates a tip that provides helpful information related to the current procedure or topic.
	Indicates a important supplementary information pertinent to the current procedure or topic.
	Denotes a caution statement. Failure to follow the guidance provide in the caution statement could result in a loss of data or an interruption to a critical process being controlled or monitored by LookoutDirect.

bold face text	Denotes the name of dialog boxes, property sheet items, application features, and menu commands that you select as part of a procedure.
<i>italic</i>	Denotes references to dialog boxes, property sheet items, application features, and menu commands that are not part of the current procedure, or key concepts.
monospace	Denotes characters that you enter from the keyboard, code/syntax examples. The monospace font is also used to express proper names of disk drives, paths, directories, programs, subprograms, subroutines, device names, functions, operations, variables, filenames and extensions.

Technical Support

Phone

Contact Automation Direct's technical support department toll free at 1-770-844-4200. Technical support is available weekdays from 9:00 a.m. through 6:00 p.m. EST.

World Wide Web

Visit the Automation Direct website at www.automationdirect.com and click on *Technical Support* to access a wide variety of technical support assets including print-on-demand documentation and software downloads.

Expressions

This chapter explains the features and uses of Lookout*Direct* expressions, which can act as variables, capable of flexible, real-time math statements, condition testing, and other complex operations.

Lookout*Direct* expressions can use spreadsheet-style formulas with a mixture of constants and variable signals from objects. They can be short and simple, or extremely complicated with several signal inputs, function calls, and multiple levels of parentheses.

A single expression can incorporate any number of numeric, logical, and text signals within its calculation. However, the *result* of the expression can be only one of three types: numeric, logical, or text. The outermost function or operator in the expression returns a variable type that determines the overall signal type of the expression.

While an expression produces a single type of data, most Lookout*Direct* objects interpret data according to the data type that object needs for a particular data member. You can connect a logical output to a numeric input, and the arriving data will be interpreted as a 0 or a 1. Refer to the *Data Polymorphism* section of this chapter for detailed information on how Lookout*Direct* interprets data without respect to the original type.

Simple expressions consist of a single value. Simple expressions do not contain any modification, manipulation, math, or logic (for example, a single object with a name like `Pot1`). When you add any functionality to a simple expression, you create a complex expression (for example, `Pot1 > 33`).

The following examples are typical expressions. Depending on your system requirements, your expressions might be much more involved.

True

Always returns the value true (on).

Pot1

Returns the current value [the (`intrinsic`) data member] of Pot1.

879.03

Always returns the value 879.03.

1:04:33

Returns the value 0.0448264. If formatted in hours, minutes, and seconds, it is displayed as 1:04:33.

Pressure * 2.31

Multiplies Pressure by 2.31.

(Switch1 or Switch2) and TankLevel > 65.2

If Switch1 or Switch2 is true, and TankLevel is greater than 65.2, return true. Otherwise, return false.

Pressure >= Setpoint and !ReliefValve and TimeOccuring > 0:30

If Pressure is greater than or equal to Setpoint, and ReliefValve is false, and TimeOccuring is greater than 30 seconds, return true. Otherwise, return false.

if(PLC1.AutoPos, AutoTemp, if(PLC1.ManPos, ManualTemp,0))

If the alias member AutoPos of PLC1 is true, return the value of AutoTemp. If the alias member ManPos of PLC1 is true, return the value of ManualTemp. Otherwise, return the value 0.

tif(HOA=1,"Hand",tif(HOA=2,"Off","Auto"))

`tif` is the text version of the `if` function. If the HOA switch is at position 1, return `Hand`. If the HOA switch is at position 2, return `Off`. Otherwise, return `Auto`.

You can accomplish much the same thing with the following `tchoose` expression as you can with the previous `tif` expression.

tchoose(HOA,"hand","Off","Auto")

If the HOA switch is at position 1, return `Hand`. If the HOA switch is at position 2, return `Off`. If the HOA switch is at position 3, return `Auto`.

Make sure that your expressions do not attempt to calculate illegal operations, such as dividing by zero or finding the arc cosine of a number greater than 1. If a signal can assume a value that could cause `LookoutDirect` to attempt an illegal mathematical operation, you should specifically test for that condition in the expression. Expressions trap illegal mathematical operations and generate alarms in the Math alarm area. The alarms do not reset until you correct the expression. The following list includes some of the illegal conditions that `LookoutDirect` traps:

- attempting to divide by zero

- taking the square root of a negative number
- numeric underflow
- numeric overflow



Note If any value referenced in an expression changes when an event occurs, the expression automatically recalculates. This is the same event-driven concept that objects implement.

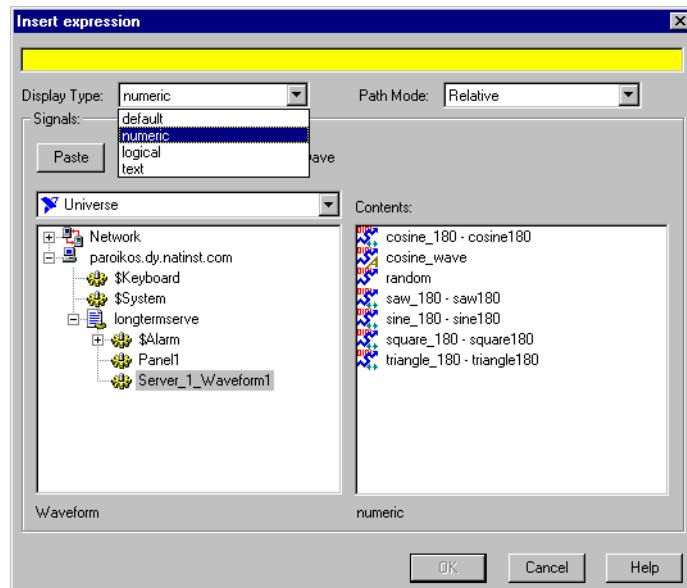
Creating Expressions

Because Lookout*Direct* uses expressions in many places, you can create expressions as independent objects, parameters in objects, connections to object data members, or you can insert expressions on control panels.

Through dialog boxes, you create expressions during process development. There are three types of data entry fields in Lookout*Direct* dialog boxes—white, yellow, and green. White data entry fields accept only constant values, yellow data entry fields accept expressions, and green fields accept URLs.

Data Polymorphism

Lookout*Direct* data is not strongly typed, which means that it does not need to be interpreted as the type it was originally cast in. Data polymorphism in Lookout*Direct* means that data of one type is interpreted appropriately when connected to an input of another type. You can select the data type you want an expression to be displayed as in the **Display Type** field, shown in the following dialog box.



The following table describes how data types are used by inputs of a different type.

Table 1-1. Data Type Conversion

Data Type	Interpreted as Numeric	Interpreted as Text	Interpreted as Logical
Logical	0 or 1	ON or OFF	—

Table 1-1. Data Type Conversion (Continued)

Data Type	Interpreted as Numeric	Interpreted as Text	Interpreted as Logical
Numeric	—	digits of the number	0 displays as OFF Any value other than 0 displays as ON
Text	<p>Appear as digits if the string consists only of digits in a valid LookoutDirect display format, such as a decimal number or a scientific expression.</p> <p>In a time format, such as 10:05:30, the text is interpreted as a number (the fraction of one day represented by the time quantity) in scientific notation.</p> <p>If the text string does not consist of digits in a valid LookoutDirect format, the text is interpreted as a 0, with the exception of ON or TRUE, which display as 1.</p> <p>The text strings OFF or FALSE are interpreted as 0, by default.</p>	—	<p>Interpreted as a 0 or as OFF, except when the text string consists of a 1, On, or True, all of which are interpreted as ON.</p> <p>By default, the text strings 0, Off, or False are interpreted as OFF.</p>

After you have placed an expression on your panel, you cannot change the data type in that expression. You have to delete that expression and place a new one if you want to change data types.

Expressions used as parameters in Lookout*Direct* objects do not permit you to select the data type because the object interprets input according to the data type required by that object. Making Lookout*Direct* data polymorphic permits you to use output of a data type different from that required by an object, as long as the output can be interpreted meaningfully by the object.

Polymorphic data types affect the DataTable object. In versions earlier than Lookout*Direct* 4, the DataTable used data members such as A1.logical to set the type of data in a particular cell.



Note The Lookout*Direct* 3.xx DataTable data members are preserved in Lookout*Direct* 4 for the sake of compatibility. It is not necessary to use the data-typed data members to simulate polymorphic data. To achieve this compatibility, however, Lookout*Direct* DataTable data is still strongly typed. You must input numeric data to DataTable numeric data members, logical data to DataTable logical data members, and so on.

Path Names in Lookout*Direct*

In versions of Lookout*Direct* prior to Lookout*Direct* 4, you could only have one process running in any instance of Lookout*Direct*, and there were no folders in a process to organize objects. Networking was done through DDE, which used specific paths. In Lookout*Direct* 4, the relationship between objects running in different processes on different computers is far more flexible and potentially complicated.

When you insert an expression on a Lookout*Direct* panel or make a connection to or between data members, you might be displaying a value that comes from the process containing that panel. You might also need to display a value that comes from another process running on the same computer or from a process running on some other computer in your network.

A path can be *absolute*, starting with a computer's fully qualified network name and leading through nested directories and subdirectories to a single object; or *relative*, instructing the computer to search for an object in some directory defined not from the computer down, but relative to the currently active folder.

To provide maximum programming flexibility, you can use a number of different levels of relativity, or *path relativity modes*, in setting the path to the data when you create an expression. These modes operate very much like other relative paths used in DOS operations, spreadsheet cells, and Web page addressing.

You can set a path for each individual object or connection, using a different level of relativity. These path relativity modes include

- Relative (the Lookout*Direct* default)
- Process Relative
- Computer Relative
- Absolute

The level of path relativity you choose makes a difference in how your process operates when copied or moved to another computer. For instance, suppose you have a client process running on computer Alfred that refers to objects in a server process also running on computer Alfred. If you run that

client process on computer Bert, will it refer to objects on computer Alfred or computer Bert?

Setting the level of path relativity is how you can control which objects your processes refer to. The key point to remember about path relativity modes in LookoutDirect is that the modes are distinguished by the path prefix.

In *Relative* mode, the form of the path is

object

or

folder\object

without prefix.

Relative mode is the default mode in LookoutDirect 4. All paths are relative to the folder or process that contains the expression you are creating or the object you are connecting to. Use Relative mode when you connect objects that should be grouped together and that should be copied or moved as a group.

Connections between objects in a folder should use Relative mode so that they can be copied and moved elsewhere while maintaining the same relationship with each other.

Process Relative mode is indicated by a single prefixed backslash (\) followed by a folder or object name, as in

\folder\...folder\object

or

\object

Use Process Relative mode when you want to make a connection between two objects in different folders in the same process.

Computer Relative mode is indicated by a prefix consisting of two backslashes, a period, and a backslash (\.\.) followed by a process name, as in

\.\process\folder\...folder\object

Use Computer Relative mode when you want to reference other processes running on the same computer.

For instance, if you had a process called `Station_Control` that called a number of other processes by name on computer Alfred, you could use that

same Station_Control process on computer Bert to refer to the identically named processes running on computer Bert.

Absolute mode is indicated by two backslashes (\\\) followed by a complete network path, as in

```
\computer.place.com\process\folder\...folder\object
```

Absolute mode is, in effect, a URL. Use Absolute mode when you want to refer to a particular process on a particular computer. Any references to a PLC, for instance, should be in Absolute mode so as to always refer to the computer physically connected to that PLC—no matter where the process making the reference is running.



Note Though Relative mode is the Lookout*Direct* default, when you use an object outside the immediate location of your expression, the path becomes more specific. The addition of two periods at the beginning of a path, for instance, can be used to specify the parent of the first folder listed, as in ..\folder\object.

You should manually change a path mode to a level beyond that strictly necessary under your current circumstances if you want to ensure that your process will still work properly if you relocate it, completely or partially.

Expressions on Control Panels

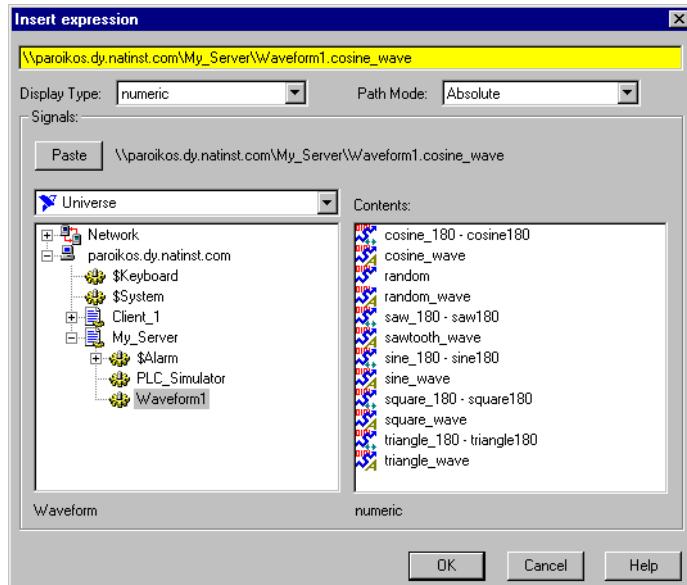
With the **Insert»Expression** command, you graphically display the result of an expression on a control panel. However, this method does not create an object; therefore, it does not create an output signal that other expressions or objects can use. (There is no name associated with the expression.)

You can insert this same kind of expression by dragging and dropping the data member you want to display from the Lookout*Direct* Object Explorer.



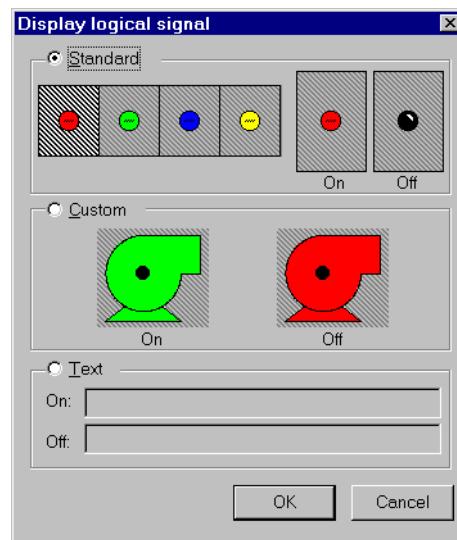
Note When you insert an expression through the menu, you can explicitly choose the path relativity before you create the expression. If you insert an expression by dragging a data point from the Lookout*Direct* Object Explorer, the path will reflect the node you dragged the expression from. If you drag and drop an expression from the local node of Lookout*Direct* running on your computer, you will insert an expression with a relative path. If you drag an expression from a process under the network nodes for Lookout*Direct* processes (including the network node for your local computer), the expression will have an absolute path. Refer to the *Path Names in Lookout*Direct** section for more information on Lookout*Direct* path modes.

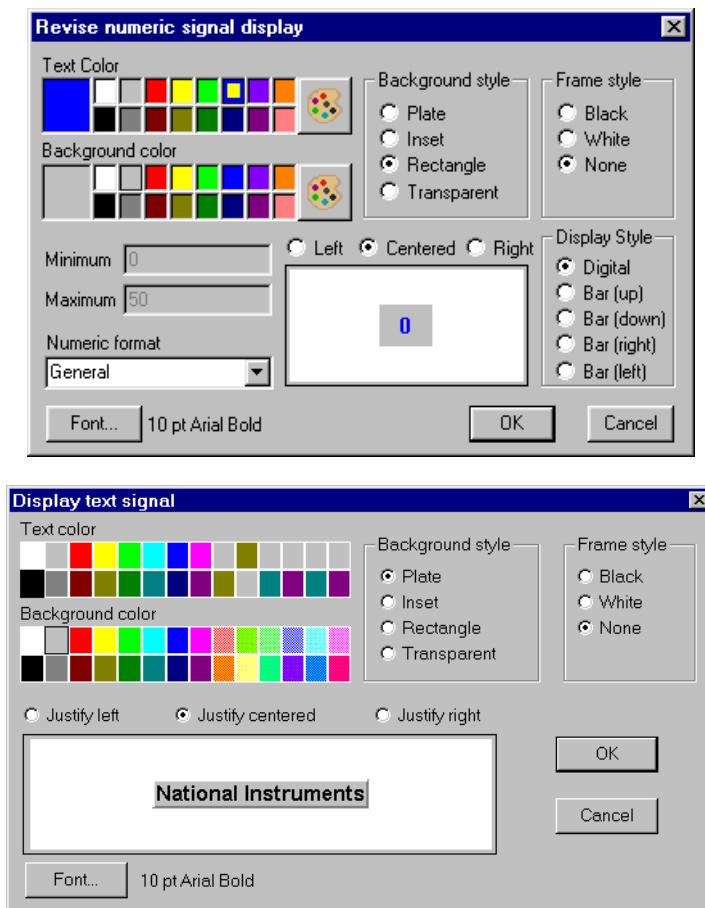
The following illustration shows the **Insert Expression** dialog box you use to create or modify the path for a typical Lookout*Direct* expression.



Notice that you can set the data type for the expression to display. After you have created an expression, you can no longer modify the data type, so be sure you select the correct type for your display.

The following illustrations show typical Lookout *Direct* dialog boxes to select the display properties for expressions.

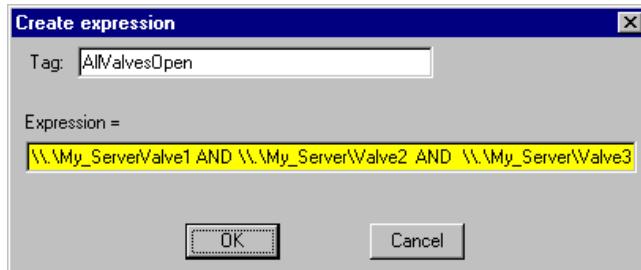




Expression Objects

With the **Object»Create** menu command, you can create an Expression object, or *named expression*. Like other object classes, the global (expression) object class requires a unique name.

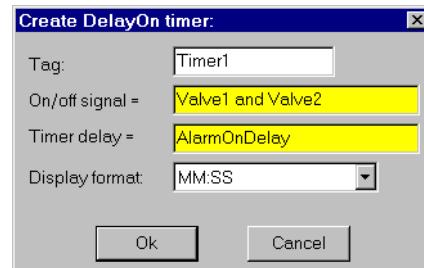
Other expressions and/or objects can reference the output of an (expression) object. When you need to define a unique condition that your process uses multiple times, use an (expression) object. Instead of defining the same expression in many places, you can create it one time and use its name wherever the condition applies. For example, the name AllValvesOpen in the following **Create expression** dialog box is easier to reference than the long expression.



You can also use the expression editor to assemble the long expression by right-clicking in the yellow field.

Expressions as Parameters

Many object classes accept expressions as parameters. When they do, the parameter expects the expression to return a certain signal type—numeric, logical, or text. For more information about object parameters and data members, refer to the specific object class definition in the online help.



The parameter **On/off signal** expects a logical expression and **Timer delay** requires a numeric expression. Both `Valve1` and `Valve2` are names for switches. Connecting them with `and` produces a logical result and satisfies the type condition of the first parameter. `AlarmOnDelay` is the name of a potentiometer used to adjust the timer delay setpoint, which creates a numeric signal, satisfying the type condition of the second parameter.

Because the **Timer delay** parameter is an expression, we have many configuration possibilities, including the following:

- Enter a constant. In this case, the delay never changes.
- Enter the name of an output signal from another object such as a Pot. In this case, the operator adjusts.
- Enter a complex expression that automatically calculates the delay based on multiple inputs.

Similar configuration solutions exist for the **On/off signal** parameter.

When an expression parameter fields is yellow, you can get help building your expressions. For example, assume that you cannot remember whether the valve name in the previous example was `Valve1`, `Valve_1`, or `ValveOne`. If you position the cursor over the yellow expression field and right-click, the **Expression Editor** dialog box appears. Using this dialog box, you can select the proper name (`Valve1`) and paste it directly into the expression field. Click on **OK** and *LookoutDirect* writes the expression into the targeted parameter field.

Expressions as Connections

You can connect object data members with expressions. Writable data members accept expressions as inputs, much like parameters. To connect an expression to a data member, use the **Object»Edit Connections** menu command, or right-click on the object you want to connect to in the Object Explorer, and select **Edit Connections**. See Chapter 4, *Using LookoutDirect*, of the *Getting Started With LookoutDirect* manual for detailed information on connecting expressions to data members.

Edit/Insert Expression Dialog Box

You can insert an expression as a display element on a control panel, use it as a parameter for a *LookoutDirect* object, or use it as an input data member for a *LookoutDirect* object. When you use an expression as a display object, you can create or modify the expression using the **Insert Expression** dialog box. When you use an expression as a parameter or input to a data member, you can create or modify the expression using the **Edit Expression** dialog box.

These dialog boxes are almost identical, the only difference being that the **Insert Expression** dialog box contains a **Display Type** field that you use to select how the data expression will appear on your control panel. This field does not exist in the **Edit Expression** dialog box because an expression being used as a parameter or as an input to an object data member will be interpreted as the data type required for that input.



Note After you insert an expression for display, you cannot change the display type of the expression. The **Display Type** dialog box will be disabled if you open the expression to edit it. To change the data type, you have to insert a new expression.

You can access the **Insert Expression** dialog box in any of the following ways:

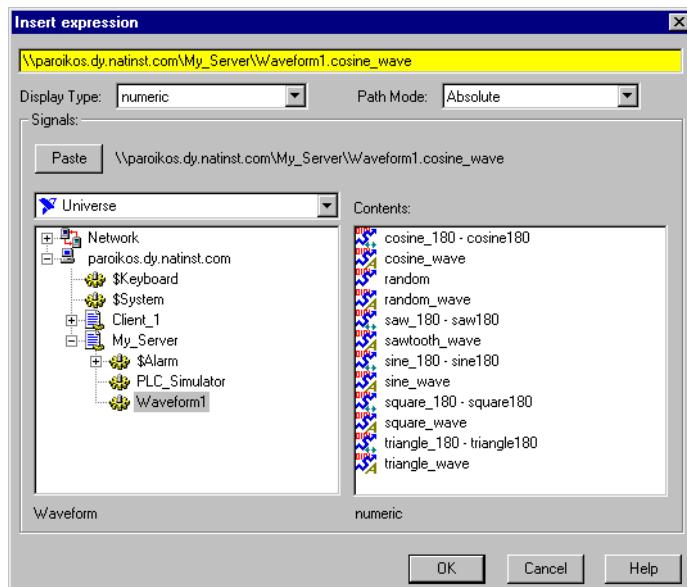
- Select **Insert»Expression** from the *LookoutDirect* menu.

- Click-drag an object from the Object Explorer.
- Right-click on an expression display on a control example and select **Object Properties**.
- Right-click in a control panel and select **Insert Expression**.

You can access the **Edit Expression** dialog box by right-clicking in any yellow expression field in any *LookoutDirect* dialog box.

Except for setting a display data type in the **Insert Expression** dialog box, you use the two dialog boxes in the same way. The following instructions and comments apply to both dialog boxes.

Figure 1-4 shows the **Insert expression** dialog box. You can enter your expression in the yellow expression field at the top of this dialog box.



Just below the yellow expression field are the **Display Type** and **Path Mode** selection boxes. The **Display Type** box selects the data type for your expression to be displayed with (if relevant), and the **Path Mode** box selects how detailed the path name of your expression will be.

The lower-left field lists all objects that generate readable signals under the root shown in the root window (under the **Paste** button). The lower-right field lists the readable data members (**Contents**) for the currently selected object. Notice that the dialog box indicates the *object class* selected in the object list (Waveform) and the *signal type* selected in the **Contents** list (numeric).

You can enter an expression directly or use the **Paste** button to select and insert object names in the expression field. As you select different objects from the listbox, the name to the right of the **Paste** button changes accordingly.

Some objects have multiple readable data members, such as a Modbus object. Lookout*Direct* concatenates the name, followed by a period and the selected data member.

After you combine the desired name and data member, click on the **Paste** button. Lookout*Direct* pastes the name into the expression field. Clicking on the **Paste** button a second time copies another instance of the name to the expression window—this time at the cursor.



Tip Instead of selecting a data member and then clicking on the **Paste** button, double-click on the data member. Lookout*Direct* automatically pastes the name into the expression window.

You can manually type or modify a name, a data member, or a mathematical function in the expression field at any time. Because you might have many hard-to-remember, defined objects, the navigation and selection listboxes serve as a quick reference for all of your previously defined objects.

Expression Syntax

In an expression, operators are instructions to perform an operation on a value or to combine values to form a new value. For example, a simple operator is the / symbol. It divides one value by another; for example, the formula (5 / 2) reads *five divided by two* and produces a result of 2.5. The five and two in the preceding example are operands (the / operator requires numeric operands).

White Space

You can use tabs and spaces between operators, functions, and function parameters in Lookout*Direct* to make expressions easier to read. This manual uses spaces between operators for added clarity.

Spaces and tabs are called *white space* characters because they provide space between parameters. This practice is used for the same reasons that spaces are used between words and paragraphs in a book—to achieve greater organization and clarity. Because Lookout*Direct* ignores white space characters, you can use white space characters to separate object names in an expression. However, you cannot embed white space characters within names or alias names.

Arithmetic Operators

The following operators perform basic arithmetic operations. They require numeric operands and produce numeric results.

Table 1-2. Arithmetic Operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Percentage (divides preceding value by 100)
^	Exponentiation
-	Negation (additive inverse of the following value)

Text Operator

The text operator is the ampersand character (&). An ampersand joins two text strings. For instance, the expression "Call me" & "Ishmael" produces a text signal of Call me Ishmael. Typical uses for the text operator include

- imbedding a numeric value within a text string
- using it in an action verification expression
- using it in an alarm message
- sending out to a remote PLC display panel

The expression "Flow rate is" & TEXT(Flow, "0.00")& "gpm" produces a text signal of Flow rate is 141.23 gpm, assuming Flow is a numeric signal whose value rounds to 141.23.

Comparison Operators

Comparison operators compare two numeric values and produce a logical result of either TRUE or FALSE (on or off).

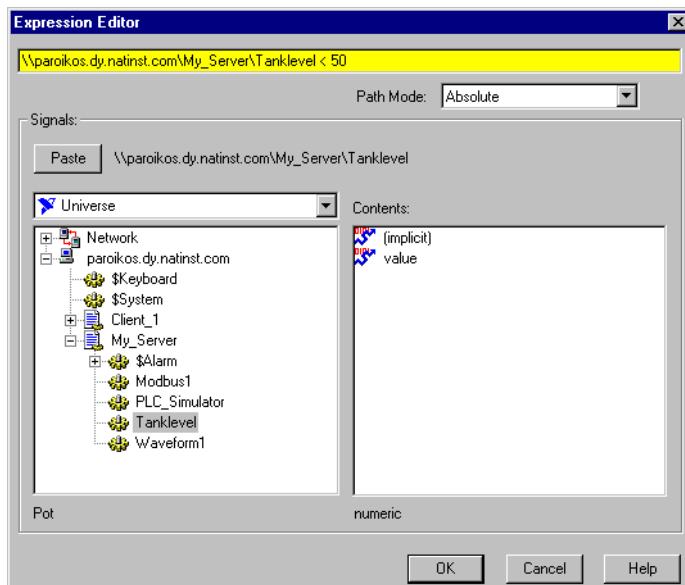
Table 1-3. Comparison Operators

<	Is less than
>	Is greater than
<=	Is less than or equal to
>=	Is greater than or equal to

Table 1-3. Comparison Operators (Continued)

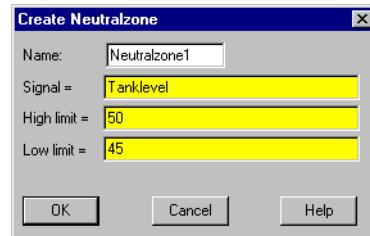
=	Is equal to
<>	Is not equal to

When setting up process control strategies, you often want to compare two numeric values. For example, you can use the result of the following expression to control a tank fill valve: TankLevel<50 is TRUE while TankLevel is less than 50 and FALSE while TankLevel is greater than or equal to 50. The use of the < operator makes this a logical test, returning true or false depending on how the expression evaluates.

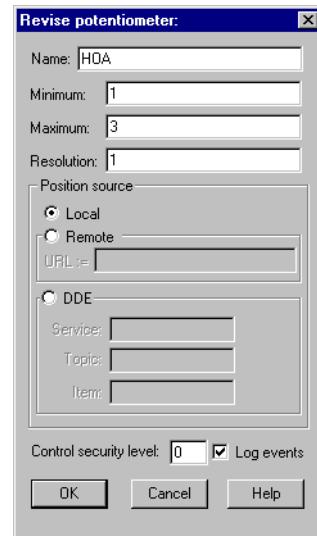


You can then connect this logical result to a PLC or RTU to control the fill valve so that the valve opens when the tank level drops below 50 and closes when the tank level rises above 50. Notice, however, that this method can cause the valve to fluctuate between open and closed too often if the tank level hovers around 50.

Neutralzone is an object class with built-in dead band that is more appropriate for controlling the tank fill valve. The following dialog box shows a better way to control a valve than the expression in the previous figure.

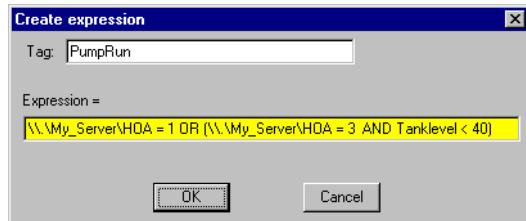


Avoid using the *is equal to* ($=$) and *is not equal to* (\neq) comparison operators to compare an analog value from a PLC or the result of mathematical calculations in an expression. Because numeric (floating point) values have about 17 significant digits, $TankLevel = 40$ might never be exactly true. If you know the signals are integer values, such as the numeric signal from the potentiometer in the following dialog box, use the $=$ and \neq operators.



In the following **Create expression** dialog box, a three-position switch is created with the Pot object class, where 1=Hand, 2=Off, and 3=Auto.

Next, an expression object is created that turns the pump on in two conditions: if the HOA switch is in the Hand position, or if the switch is in the Auto position and the TankLevel is less than 40.



Notice that the HOA signal is exactly 1, 2, or 3.



Tip You could also use the *LookoutDirect* Radiobutton object instead of creating an HOA Pot.

Expression Functions

There are over 50 built-in expression functions, generally classified as follows:

- logical functions
- lookup functions
- mathematical functions
- statistical functions
- text functions
- trigonometric functions
- date/time functions
- quality functions

The remainder of this chapter describes each function, specifies the syntax, and provides an example on how to use the function.

Logical Functions

AND	Syntax	<i>logical1 AND logical2 AND logical3...</i>
	Example	Switch1 AND Pot1 > 50.0
	Description	If <i>Switch1</i> is on and <i>Pot1</i> is greater than 50.0, return TRUE, otherwise return FALSE. Returns TRUE if all logical parameters are TRUE.

FALSE	Syntax	<code>FALSE</code>
	Example	<code>False</code>
	Description	Returns the logical value FALSE. Accepts no parameters.
IF	Syntax	<code>IF (logical, result1, result2)</code>
	Example	<code>IF(Switch1, 99.9, Pot1)</code>
	Description	If <i>logical</i> is TRUE, return <i>result1</i> , otherwise return <i>result2</i> . In the example, if Switch1 is on, it outputs the value 99.9. Otherwise, it outputs the value of Pot1.
LIF	Syntax	<code>LIF (logical, logical result1, logical result2)</code>
	Example	<code>LIF(Switch1, False, Pot1 > 50.0)</code>
	Description	Included for compatibility with early versions of LookoutDirect. If <i>logical</i> is TRUE, return <i>logical result1</i> , otherwise return <i>logical result2</i> . In the example, if Switch1 is on, it outputs the logical value False. Otherwise, it returns the logical result of <code>Pot1 > 50.0</code> .
NIF	Syntax	<code>NIF (logical, numeric result1, numeric result2)</code>
	Example	<code>NIF(Switch1, 99.9, Pot1)</code>
	Description	Included for compatibility with early versions of LookoutDirect. If <i>logical</i> is TRUE, return <i>numeric result1</i> , otherwise return <i>numeric result2</i> . In the example, if Switch1 is on, it outputs the numeric value 99.9. Otherwise it outputs the value of Pot1.
NOT	Syntax	<code>NOT (logical) or !logical</code>
	Example	<code>NOT(Switch1) or !Switch1</code>
	Description	If <i>logical</i> is TRUE, return FALSE, else return TRUE. In the example, if Switch1 is on, return FALSE, but if Switch1 is off, return TRUE.
OR	Syntax	<code>logical1 OR logical2 OR logical3...</code>
	Example	<code>Switch1 OR Pot1 > 50.0</code>
	Description	Returns TRUE if at least one logical parameter is TRUE. In the example, if either Switch1 is on or Pot1 is greater than 50.0, or both, the expression returns TRUE. If neither condition is true, it returns FALSE.

TIF	Syntax	<code>TIF (logical, text result1, text result2)</code>
	Example	<code>TIF(Switch1, "Text Message", TextInput1)</code>
	Description	Included for compatibility with early versions of <code>LookoutDirect</code> . If <i>logical</i> is TRUE, return <i>text result1</i> , otherwise return <i>text result2</i> . Notice the use of quotation marks. In the example, if <code>Switch1</code> is on, it outputs <code>Text Message</code> ; otherwise, it outputs the current text value of <code>TextInput1</code> .
TRUE	Syntax	<code>TRUE</code>
	Example	<code>TRUE</code>
	Description	Returns the logical value TRUE. Accepts no parameters.
XOR	Syntax	<code>logical1 XOR logical2 XOR logical3...</code>
	Example	<code>Switch1 XOR Pot1 > 50.0</code>
	Description	Returns TRUE if <i>only one</i> logical parameter is TRUE. In the example, if <code>Pot1</code> is greater than 50 and <code>Switch1</code> is on, it outputs FALSE because both logical parameters are TRUE. If <code>Pot1</code> is less than 50 and <code>Switch1</code> is on, it outputs TRUE because only one logical parameter is TRUE.

Lookup Functions

CHOOSE	Syntax	<code>CHOOSE (numeric, value1, value2, value3,...)</code>
	Example	<code>CHOOSE(Pot1, "Switch1", "TRUE", "Pb1")</code>
	Description	Returns the value parameter corresponding to the integer portion of the numeric parameter. If the <i>numeric</i> parameter is less than 2.0, <code>CHOOSE</code> returns <i>value1</i> . If the <i>numeric</i> parameter is greater than the number of value parameters, <code>CHOOSE</code> returns the last logical parameter listed. In the example, if the value of <code>Pot1</code> is .5, 1, or 1.4, it returns the value of <code>Switch1</code> . If <code>Pot1</code> is 2.0 or 2.8, <code>CHOOSE</code> returns the logical value TRUE. If <code>Pot1</code> is 3.0 or higher, <code>CHOOSE</code> returns the value of <code>Pb1</code> .

LCHOOSE	Syntax	<code>LCHOOSE (numeric, logical1, logical2, logical3,...)</code>
	Example	<code>LCHOOSE(Pot1, Switch1, TRUE, Pb1)</code>
	Description	Included for compatibility with early versions of <i>LookoutDirect</i> . Returns the logical parameter corresponding to the integer portion of the <i>numeric</i> parameter. If the <i>numeric</i> parameter is less than 2.0, LCHOOSE returns <i>logical1</i> . If the <i>numeric</i> parameter is greater than the number of logical parameters, LCHOOSE returns the last logical parameter listed.
NCHOOSE	Syntax	<code>NCHOOSE (numeric, numeric1, numeric2, numeric3,...)</code>
	Example	<code>NCHOOSE(Pot1, Pot2, Pot3, 14.3)</code>
	Description	Included for compatibility with early versions of <i>LookoutDirect</i> . Returns the <i>numericx</i> parameter corresponding to the integer portion of the <i>numeric</i> parameter. If the <i>numeric</i> parameter is less than 2.0, NCHOOSE returns <i>numeric1</i> . If the <i>numeric</i> parameter is greater than the number of <i>numericx</i> parameters, NCHOOSE returns the last <i>numericx</i> parameter listed.
TCHOOSE	Syntax	<code>TCHOOSE (numeric, text1, text2, text3,...)</code>
	Example	<code>TCHOOSE(Pot1, "Auto", "Manual", "Local", "Locked")</code>
	Description	Included for compatibility with early versions of <i>LookoutDirect</i> . Returns the text parameter corresponding to the integer portion of the <i>numeric</i> parameter. If the <i>numeric</i> parameter is less than 2.0, TCHOOSE returns <i>text1</i> . If the <i>numeric</i> parameter is greater than the number of text parameters, TCHOOSE returns the last logical parameter listed.

Mathematical Functions

ABS	Syntax	<code>ABS(numeric)</code>
	Example	<code>ABS(Pot1 - 50.0)</code>
	Description	Returns the absolute value of <i>numeric</i> . In the example, if <i>Pot1</i> is zero, the function returns 50.0.

EXP	Syntax	<code>EXP(numeric)</code>
	Example	<code>EXP(Pot1)</code>
	Description	Returns the base of the natural logarithm, e (2.71828182), raised to the power of <i>numeric</i> . In the example, if <code>Pot1</code> is 2.0, the function returns 2.71828182^2 or approximately 7.38906. EXP is the inverse of the function, LN. To calculate the values of other powers, use the exponentiation operator ($^$).
FACT	Syntax	<code>FACT(numeric)</code>
	Example	<code>FACT(4 . 2)</code>
	Description	Returns the factorial of the integer portion of <i>numeric</i> . In the example, the function returns the factorial of 4, or $1*2*3*4$, or 24. The factorial of zero, FACT(0), or any number less than zero is one (1).
INT	Syntax	<code>INT(numeric)</code>
	Example	<code>INT(4 . 2)</code>
	Description	Rounds <i>numeric</i> down to the nearest integer. In the example, the INT function returns 4. Notice also that <code>INT(-8.5) = -9</code> . See also TRUNC.
LN	Syntax	<code>LN(numeric)</code>
	Example	<code>LN(PLC.AI1)</code>
	Description	Returns the natural logarithm of <i>numeric</i> . In the example, if <code>PLC.AI1 = 1000.0</code> , the function returns 6.90776.
LOG	Syntax	<code>LOG(numeric1, numeric2)</code>
	Example	<code>LOG(Pot1, 2)</code>
	Description	Returns the logarithm of <i>numeric1</i> to the <i>numeric2</i> base, where <i>numeric1</i> is a positive real number. In the example, if <code>Pot1 = 8.0</code> , the function returns 3.0.
LOG10	Syntax	<code>LOG10(numeric)</code>
	Example	<code>LOG10(Pot1)</code>
	Description	Returns the base-10 logarithm of <i>numeric</i> , where <i>numeric</i> is a positive real number. In the example, if <code>Pot1 = 100.0</code> , the function returns 2.0.

MOD	Syntax	<code>MOD(numeric1, numeric2)</code>
	Example	<code>MOD(50,8)</code>
	Description	Returns the modulus (remainder) of <i>numeric1</i> divided by <i>numeric2</i> , where <i>numeric2</i> is not equal to zero. In the example, 50 divided by 8 equals 6 with a remainder of 2. Therefore, the function returns 2.
PI	Syntax	<code>PI()</code>
	Example	<code>PI()</code>
	Description	Returns an approximation of PI: 3.1415927. Accepts no parameters.
PRODUCT	Syntax	<code>PRODUCT(numeric1, numeric2, numeric3, ...)</code>
	Example	<code>PRODUCT(2,4,6,10)</code>
	Description	Returns the product of all the numerics. In the example, the function returns $2*4*6*10$ or 480.
RAND	Syntax	<code>RAND()</code>
	Example	<code>RAND() * Pot1</code>
	Description	Generates a new random number between zero and one every time the formula that this function is a part of is recalculated. Accepts no parameters. In the example, every time the value of <i>Pot1</i> changes, the function generates a new random number and multiplies it by the value of <i>Pot1</i> .
ROUND	Syntax	<code>ROUND(numeric1, numeric2)</code>
	Example	<code>ROUND(Pot1,2)</code>
	Description	Rounds the value of <i>numeric1</i> to <i>numeric2</i> decimal places. In the example, if <i>Pot1</i> equals 15.745, the function returns 15.75. If <i>numeric2</i> equals zero, the function returns an integer. If <i>numeric2</i> is less than zero, the function returns zero.
SIGN	Syntax	<code>SIGN(numeric)</code>
	Example	<code>SIGN(Pot1)</code>
	Description	Returns 1 if <i>numeric</i> is positive, 0 if <i>numeric</i> is 0, -1 if <i>numeric</i> is negative. In the example, if <i>Pot1</i> is -0.00001, the function returns -1.

SQRT	Syntax	<code>SQRT(numeric)</code>
	Example	<code>SQRT(ABS(Pot1))</code>
	Description	Returns the square root of <i>numeric</i> , where <i>numeric</i> is a positive real number. In the example, if Pot1 is -25.0, the absolute function first converts the Pot value to positive 25 and the square root function calculates $\sqrt{25} = 5$.
TRUNC	Syntax	<code>TRUNC(numeric)</code>
	Example	<code>TRUNC(8.9)</code>
	Description	Truncates <i>numeric</i> to its integer component by removing its fractional part. In the example, the TRUNC function returns 8. Notice also that <code>TRUNC(-8.9) = -8</code> . See also <i>INT</i> .

Statistical Functions

AVG	Syntax	<code>AVG(numeric1, numeric2, numeric3, ...)</code>
	Example	<code>AVG(2, 4, -6, 12)</code>
	Description	Returns the arithmetic mean (average) of all the numerics listed. This function requires at least two numeric parameters. In the example, the function returns $(2+4-6+12)/4$ or 3.0.
MAX	Syntax	<code>MAX(numeric1, numeric2, numeric3, ...)</code>
	Example	<code>MAX(2, 4, -6, 12)</code>
	Description	Returns the highest value of all the numeric values listed. This function requires at least two numeric values. In the example, the function returns 12.
MIN	Syntax	<code>MIN(numeric1, numeric2, numeric3, ...)</code>
	Example	<code>MIN(2, 4, -6, 12)</code>
	Description	Returns the lowest value of all the numeric values listed. This function requires at least two numeric values. In the example, the function returns -6.

STDEV	Syntax	<code>STDEV(numeric1, numeric2, numeric3,...)</code>
	Example	<code>STDEV(2.9,4.5,5.0,4.3,3.8)</code>
	Description	Returns the sample standard deviation of the numeric values listed. (Standard deviation is a measure of dispersion, calculated as the positive square root of the variance.) This function calculates the standard deviation using the non-biased or $n-1$ method. This function requires at least two numeric values. In the example, the function returns 0.796869.
STDEVP	Syntax	<code>STDEVP(numeric1, numeric2, numeric3,...)</code>
	Example	<code>STDEVP(2.9,4.5,5.0,4.3,3.8)</code>
	Description	Returns the standard deviation of a full population of numeric values. This function calculates the standard deviation using the biased or n method. This function requires at least two numeric values. In the example, the function returns 0.712741.
SUM	Syntax	<code>SUM(numeric1, numeric2, numeric3,...)</code>
	Example	<code>SUM(2,4,-6,12)</code>
	Description	Returns the sum of all the numeric values listed. This function requires at least two numeric values. In the example, the function returns 12.
VAR	Syntax	<code>VAR(numeric1, numeric2, numeric3,...)</code>
	Example	<code>VAR(2.9,4.5,5.0,4.3,3.8)</code>
	Description	Returns the sample variance of the numeric values listed. (Variance is a measure of dispersion.) This function calculates the variance using the non-biased or $n-1$ method. This function requires at least two numeric values. In the example, the function returns 0.635.
VARP	Syntax	<code>VARP(numeric1, numeric2, numeric3,...)</code>
	Example	<code>VARP(2.9,4.5,5.0,4.3,3.8)</code>
	Description	Returns the population variance of the numeric values listed. This function calculates the variance using the biased or n method. This function requires at least two numeric values. In the example, the function returns 0.508.

Text Functions

EXACT	Syntax	<code>EXACT(text1, text2)</code>
	Example	<code>EXACT(TextEntry1, "batch a")</code>
	Description	Returns TRUE if <i>text1</i> exactly matches <i>text2</i> , otherwise returns FALSE. This function is case sensitive. In the example, the function returns TRUE if the text result of TextEntry1 exactly matches batch a (notice that the entry in this example is all lowercase).
FIND	Syntax	<code>FIND(text1, text2, numeric)</code>
	Example	<code>FIND("ON", "Reactor A is ON", 1)</code>
	Description	Searches for <i>text1</i> within <i>text2</i> starting after <i>numeric</i> characters, and returns the position of the first character where the match begins. This function is case sensitive. The output of this function is numeric. It returns 0 if no match is found. In the example, the function searches the entire string for the word, ON and returns 14, indicating the position of the first character of the word. Also see <i>SEARCH</i> .
FIXED	Syntax	<code>FIXED(numeric1, numeric2)</code>
	Example	<code>"The flow is" & FIXED(Pot1,2)</code>
	Description	Rounds the value of <i>numeric1</i> to <i>numeric2</i> decimal places and then converts <i>numeric1</i> to a text string. In the example, if the value of Pot1 equals 123.456, the FIXED function returns the text value 123.46. Thus, the entire text string would read The flow is 123.46. If <i>numeric2</i> is negative or omitted, <i>numeric1</i> is rounded to the nearest whole number. For example, <code>FIXED(123.588)</code> returns 124. See also <i>TEXT</i> .
LEFT	Syntax	<code>LEFT(text, numeric)</code>
	Example	<code>LEFT("Reactor A is ON", 9)</code>
	Description	Retrieves the specified number of characters from the lefthand end of <i>text</i> and outputs it as a text value. In the example, the function returns Reactor A.

LEN	Syntax	<code>LEN(text)</code>
	Example	<code>LEN("Reactor A is ON")</code>
	Description	Returns the length of the text string (the number of characters in <i>text</i>). In the example, the function returns the numeric value 15.
LOWER	Syntax	<code>LOWER(text)</code>
	Example	<code>EXACT(LOWER(TextEntry1), "batch a")</code>
	Description	Converts <i>text</i> to all lower case. In the example, the LOWER function ensures that a match is found whether TextEntry1 content reads Batch A, BATCH A, or batch A.
MID	Syntax	<code>MID(text, numeric1, numeric2)</code>
	Example	<code>MID("Reactor A is ON", 9, 7)</code>
	Description	Retrieves <i>numeric2</i> characters from <i>text</i> , beginning at character <i>numeric1</i> . In the example, the function returns the text value A is ON.
PROPER	Syntax	<code>PROPER(text)</code>
	Example	<code>PROPER("reactor A is ON")</code>
	Description	Capitalizes the first character of each word in <i>text</i> . In the example, the function returns the text value Reactor A Is On.
REPLACE	Syntax	<code>REPLACE(text1, numeric1, numeric2, text2)</code>
	Example	<code>REPLACE("Reactor A is ON", 9, 1, "B")</code>
	Description	Replaces <i>numeric2</i> characters with <i>text2</i> beginning with character <i>numeric1</i> in <i>text1</i> . In the example, the function returns the text value Reactor B is ON.
REPT	Syntax	<code>REPT(text, numeric)</code>
	Example	<code>REPT("LookoutDirect", Pot1)</code>
	Description	Repeats <i>text</i> <i>numeric</i> times. In the example, if the value of Pot1 is 8, the function returns the text value, LookoutDirect LookoutDirect LookoutDirect LookoutDirect LookoutDirect LookoutDirect LookoutDirect LookoutDirect.

RIGHT	Syntax	<code>RIGHT(text, numeric)</code>
	Example	<code>RIGHT("Reactor A is ON", 7)</code>
	Description	Retrieves the specified number of characters from the righthand end of <i>text</i> and outputs it as a text value. In the example, the function returns the text value <i>A is ON</i> .
SEARCH	Syntax	<code>SEARCH(text1, text2, numeric)</code>
	Example	<code>SEARCH("on", "Reactor A is ON", 1)</code>
	Description	Finds <i>text1</i> within <i>text2</i> beginning at <i>numeric</i> character, and returns the position of the first character where the match begins. This function is not case sensitive. The output is numeric. It returns 0 if no match is found. In the example, the function searches the entire string for the word <i>ON</i> , <i>on</i> , <i>On</i> , or <i>oN</i> and returns 14, indicating the position of the first character of the word. Refer also to <i>FIND</i> .
TEXT	Syntax	<code>TEXT(numeric, text)</code>
	Example	<code>TEXT(Pot1, "0.00")</code>
	Description	Included for compatibility with early versions of <i>LookoutDirect</i> . With polymorphic data, this function becomes unnecessary. Like <i>FIXED</i> , the <i>TEXT</i> function converts <i>numeric</i> to a textual value. While <i>FIXED</i> allows you to specify the number of decimal points, <i>TEXT</i> allows you specify a desired numeric format in the <i>text</i> parameter. If <i>Pot1</i> equals 12.3456, the function would return the textual value 12.35. See <i>Numeric Formats</i> in Chapter 5, <i>Developer Tour</i> , in your <i>Getting Started with LookoutDirect</i> manual. See also <i>FIXED</i> .
TRIM	Syntax	<code>TRIM(text)</code>
	Example	<code>TRIM("Reactor A is ON")</code>
	Description	Trims multiple spaces from between words. In the example, there are multiple spaces on either side of the word, <i>A</i> . This function returns <i>Reactor A is ON</i> , eliminating all repeated spaces.

UPPER	Syntax	<code>UPPER(text)</code>
	Example	<code>UPPER(text)</code>
	Description	Converts <i>text</i> to all capital letters (upper case). In the example, the UPPER function ensures that a match is found whether TextEntry1 content reads Batch A, BATCH A, or batch A.

Trigonometric Functions

ACOS	Syntax	<code>ACOS(numeric)</code>
	Example	<code>ACOS(Pot1)</code>
	Description	Returns the arccosine of <i>numeric</i> (that is, it returns the angle of the cosine you specify as <i>numeric</i>). <i>Numeric</i> must range between -1 and 1. The result is output in radians and ranges from 0 to π . In the example, if <i>Pot1</i> = 0.5, the function returns 1.0472 radians (60 degrees). You can express the arccosine in degrees by multiplying the result by $180/\pi$.
ASIN	Syntax	<code>ASIN(numeric)</code>
	Example	<code>ASIN(Pot1)</code>
	Description	Returns the arcsine of <i>numeric</i> (that is, it returns the angle of the sine you specify as <i>numeric</i>). <i>Numeric</i> is the sine of the angle and must range between -1 and 1. The resulting value is given in radians and ranges from $-\pi/2$ to $\pi/2$. In the example, if <i>Pot1</i> = -1, the function returns -1.5708.
ATAN	Syntax	<code>ATAN(numeric)</code>
	Example	<code>ATAN(Pot1)</code>
	Description	Returns the arctangent of <i>numeric</i> (that is, it returns the angle of the tangent that you specify as <i>numeric</i>). The resulting value is given in radians and ranges from $-\pi/2$ to $\pi/2$. In the example, if <i>Pot1</i> = 180, the function returns 1.56524 radians (about 90 degrees).

ATAN2	Syntax	<code>ATAN2(numericX, numericY)</code>
	Example	<code>ATAN2(5, 5)</code>
	Description	Returns the arctangent of the specified x- and y-coordinates (that is, it returns the angle of a line extending from the origin (0,0) to a point specified by the <i>numericX</i> , <i>numericY</i> coordinate pair that you specify). The resulting value is given in radians and ranges from greater than $-\pi$ to π . In the example, the function returns 0.785398 radians (45 degrees).
COS	Syntax	<code>COS(numeric)</code>
	Example	<code>COS(Pot1)</code>
	Description	Returns the cosine of <i>numeric</i> where <i>numeric</i> is the angle in radians. In the example, if <i>Pot1</i> = 1.047, the function returns 0.500171. You convert degrees to radians by multiplying by $\pi/180$.
SIN	Syntax	<code>SIN(numeric)</code>
	Example	<code>SIN(Pot1)</code>
	Description	Returns the sine of <i>numeric</i> where <i>numeric</i> is the angle in radians. In the example, if <i>Pot1</i> = 3.14159, the function returns 1.2246E-16 (effectively zero). You convert degrees to radians by multiplying by $\pi/180$.
TAN	Syntax	<code>TAN(numeric)</code>
	Example	<code>TAN(Pot1*PI() /180)</code>
	Description	Returns the tangent of <i>numeric</i> where <i>numeric</i> is the angle in radians. In the example, the value of <i>Pot1</i> is 45, but it is in degrees, not radians. Convert it to radians by multiplying it by $\pi/180$. In this example, the function returns 1.

Date/Time Functions

NOW	Syntax	<code>NOW(<i>logical</i>)</code>
	Example	<code>NOW(\$Keyboard.F1)</code>
	Description	Returns a numeric value representing the current system date and time when any signal within the parentheses changes. The result is a floating point number in which the integer represents the date and the fraction represents the time of day. In the example, when you press the function key <F1>, the date and time are output as a single number, like 34738.3. If you change the Numeric Format of the number to mm/dd/yy hh:mm:ss, the same number would be shown as 02/08/95 07:49:02.
TODAY	Syntax	<code>TODAY(<i>logical</i>)</code>
	Example	<code>TODAY(\$Keyboard.F1)</code>
	Description	Returns a numeric value representing the current system date when any signal within the parentheses changes. The result is an integer that represents the number of days that have passed since Jan. 1, 1900. In the example, when you press the function key F1, the date is output as a single number, such as 34738. If you change the Numeric Format of the number to mm/dd/yy, the same number would be shown as 02/08/95.



Note If you want to display the current time only, subtract the TODAY function from the NOW function.

Example: `NOW($Keyboard.F1) - TODAY($Keyboard.F1)`

In this example, if you want the result to update itself every second, replace `$Keyboard.F1` with a one second pulse timer.

Quality Functions

Lookout*Direct* monitors data quality and informs you if the data quality has gone bad. When you display data on a control panel directly through an expression, or when an object with a displayable element is connected to some other object and receives bad data, the quality problem is indicated by a red X that Lookout*Direct* superimposes over the display.

Some data members might not be displayed or may not connect to an object on some particular panel, but you may still need to monitor the quality of that data. You can use the `qgood` and `qbad` functions for a quick check of quality.

Also, although a red X tells you there is a data quality problem, you might need more specific information about what has actually gone wrong with the data. You can use the `qtext` function to report the specifics of what has gone wrong.

qtext	Syntax	<code>qtext(<i>expression</i>, [<i>delimiter</i>])</code>
	Example	<code>qtext(modbus.40001, "AND")</code>
	Description	Returns a text string specifying what elements of data quality have gone bad.
qgood	Syntax	<code>qgood(<i>expression</i>)</code>
	Example	<code>qgood(modbus.40001)</code>
	Description	Returns TRUE while the expression is good.
qbad	Syntax	<code>qbad(<i>expression</i>)</code>
	Example	<code>qbad(modbus.40001)</code>
	Description	Returns TRUE while the expression is bad.

Graphics

This chapter describes how to add static and dynamic graphics to a control panel and how to create and use custom graphics.

Any visible item on a *LookoutDirect* control panel is a graphic. All graphics are either static or dynamic. *Static* graphics never change state, but *dynamic* graphics change state to represent process variations. *LookoutDirect* provides an extensive graphics library. These graphics range from switches, potentiometers, and pushbuttons to bar graphs, valves, tanks, pumps, plates, insets, scales, and more. However, there might be times when the standard *LookoutDirect* graphics do not exactly fit your needs. You can use any other drawing software to create your own custom graphics.

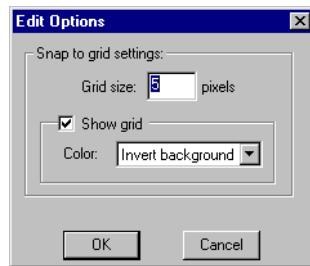


Note Consider screen resolution when creating display panels (VGA versus Super VGA). The same panel appears differently on computers using different resolution display drivers. Read about screen resolutions in the description of *Panel* objects in the online help or the online PDF *LookoutDirect Object Reference Manual* before you design your panels.

Setting Snap to Grid

You can activate the snap to grid feature in *LookoutDirect* by selecting **Edit»Snap to grid**. This feature aligns objects you are positioning to a grid on the *LookoutDirect* panel.

You can adjust this grid by selecting **Edit»Options**. The **Edit Options** dialog box appears.



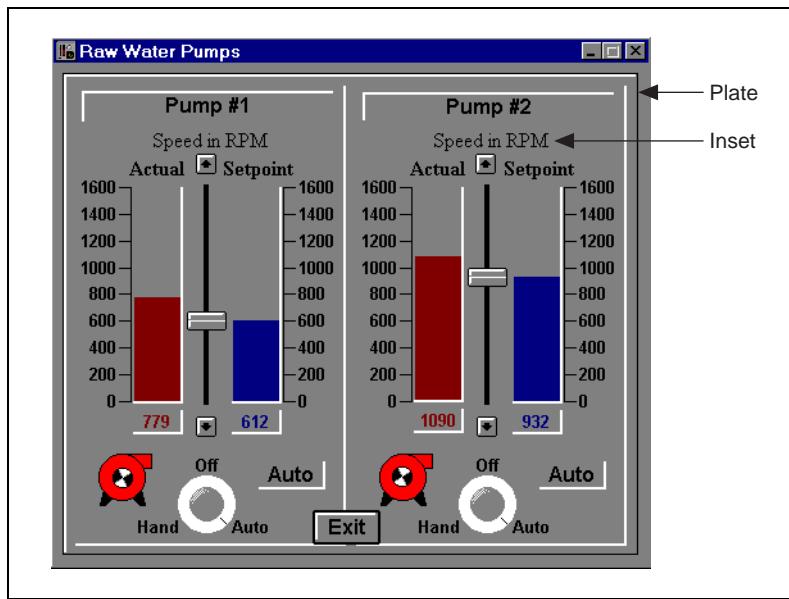
Use this dialog box to set the distance (in pixels) between grid points, show or hide the grid, and select different color options for the grid display.

Static Graphics

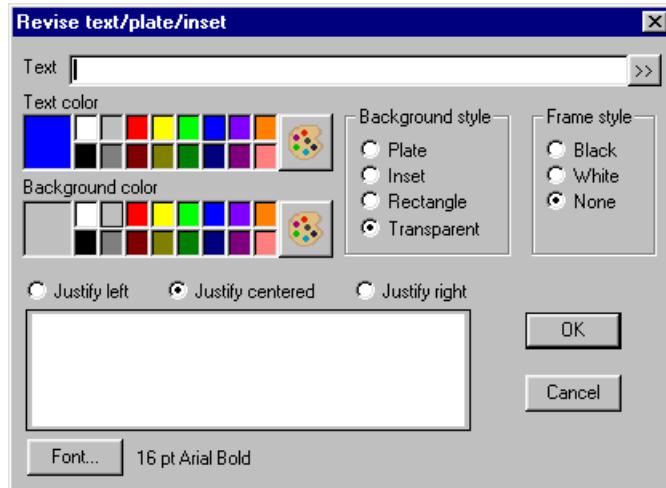
Static graphics never change state. They exist on a control panel much the same way a picture hangs on your wall, never changing or moving. Static graphics range from text labels, plates, insets, and scales to complex schematic overviews and scanned photographic images.

Displaying Text, Plates, Insets, Rectangles, and Lines

Effective use of text, plates, and insets make control panels intuitive and easy to use. The example below demonstrates how plates and insets organize information about two pumps.



To create static text, plates, insets, rectangles, or lines in Lookout*Direct*, select **Insert»Text/plate/inset**. The **Insert text/plate/inset** dialog box appears.



If you want to create a rectangle, line, plate, or inset, leave the **Text** field blank and choose a **Background style**. The preview window displays the element as it appears on a control panel.

To create a vertical or horizontal line, choose **Rectangle**. After clicking on **OK**, size the rectangle to the desired length, and reduce the width to the thickness you want your line to be. If you need the rectangle an exact size, use the yellow status bar, which indicates dimensions.



Note Lookout*Direct* uses color grids in many dialog boxes. The grids make selecting colors quick and visually helpful. However, they might look slightly different from computer to computer.

Some computer display adapters can display only 16 solid colors on the screen. The standard VGA adapter in conjunction with the Windows VGA driver supports only 16 colors directly. Windows uses a technique called *dithering* to simulate the display of colors not directly supported as solid colors. For example, Lookout*Direct* might display orange as an alternating pixel pattern of red and yellow bits on the screen.

Some objects in Lookout*Direct* require solid colors, so if you specify a dithered color, Lookout*Direct* uses the nearest solid color instead. For example, a Trend object requires a solid color for its background and solid colors for the trend lines. Furthermore, Lookout*Direct* always displays text in a solid color. If you specify pastel green for a trend line color, you might get yellow instead. Bar graphs and control panel backgrounds can display dithered colors.

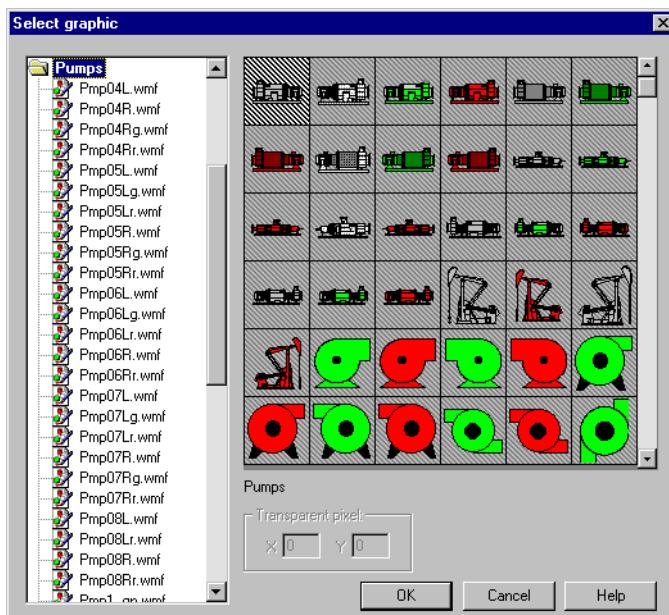


Note Some display adapters support 256 or even 16 million colors on the screen at one time. Even though Lookout*Direct* allows you to specify only 28 colors, Lookout*Direct* displays custom graphic files on control panels using all available colors.

Displaying Static Custom Graphics

Lookout*Direct* supports two graphical file types—Windows Device-Independent Bitmap (.BMP) and Windows Metafile (.WMF).

To display static bitmap and metafile graphic files on a control panel, select the **Insert»Graphic** command. The **Select graphic** dialog box appears.



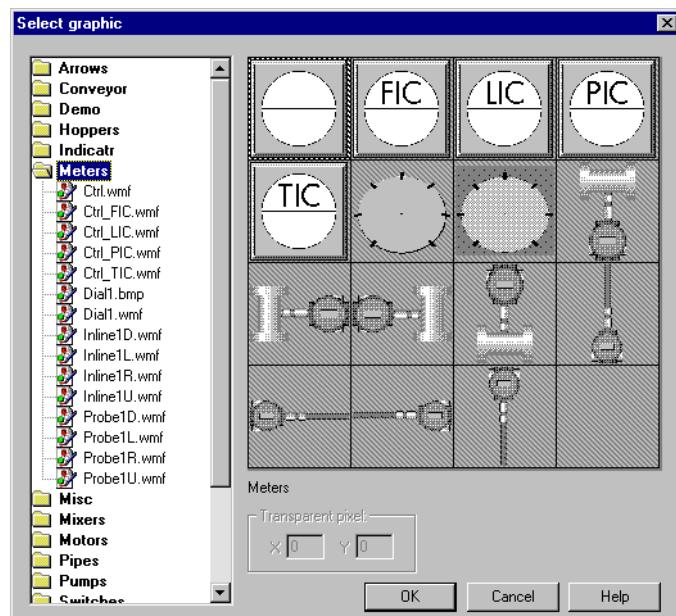
From the **Select graphic** dialog box, you can scroll through various directories containing a variety of graphic files. The list box includes all bitmaps and metafiles located in the GRAPHICS subdirectory under the root LOOKOUTDIRECT directory. As you select each category, you see thumbnail sketches displayed in the preview window to the right of the list box. Because graphics might stretch or shrink to fit in the preview window, they might not appear exactly as they do on a control panel.



Note If you know the name of the file, type the first letter of the filename. The list box automatically scrolls to the first file beginning with that letter.

If you choose a bitmap (.BMP) file, you can use the **Transparent pixel** data fields to specify which color pixels in that graphic you want to be

transparent in LookoutDirect. Imagine the viewing area as an X-Y coordinate plane with (0,0) being the top left corner of the graphic (not the entire viewing area). You can enter any **X** and **Y** coordinates, and the corresponding pixels become transparent, as do all other pixels that color. The **Transparent pixel** fields do not have any default values. If you leave the **X** and **Y** fields blank, no pixels become transparent.



You can use a multicolor bitmap as a type of mask. For this to work, some part of the bitmap (usually the interior) must be transparent and the rest of the graphic opaque—masking the underlying part of the control panel. In the case of the interface selected above (the second from the left in the second row), the pixels to be made transparent are gray. You can insert another Lookout*Direct* element, such as a bar graph, to furnish a convenient visual warning, such as a rising color level.

You could guess the X,Y coordinates of any grey pixel on the graphic. It helps to know that the center of the graphic is (-1,-1). In the example above, (-1,-1) is a black pixel; to select a gray pixel, you would have to offset your choice to (-1,0). Because this is a gray pixel, all other gray pixels become transparent when inserted on a control panel.



Note Windows metafiles are normally easier to use and manipulate than bitmap images. Unlike bitmaps, metafiles can be resized in Lookout*Direct*. Because of their inherent structure, you can also use metafiles as masks without specifying transparent pixels. Had the

example above used a .WMF graphic, the area in gray would have appeared in the *LookoutDirect* window as a crosshatched area.

Dynamic Graphics

Many process control panels require some type of animation, such as a pump changing colors to represent on and off. Table 2-1 lists the *LookoutDirect* elements that use animation or change states.

Table 2-1. Tools for Displaying Dynamic Graphics

LookoutDirect Component	Description
Animator object class	An Animator object provides full graphical animation including horizontal and vertical motion, dynamic resizing and visibility, dynamic sequencing, and color changing.
Multistate object class	A Multistate object displays up to six different custom graphics based on advanced if-then-else logic.
Pipe object class	A Pipe object makes a rectangle or line (of any dimension) change colors or blink, based on advanced if-then-else logic.
DialGauge object class	A DialGauge object displays a numeric signal as a sweeping needle on an analog gauge or dial.
Gauge object class	A Gauge object makes a numeric expression (digital number or barchart) change colors or blink based on advanced logic.
Spinner object class	A Spinner object is a small rotating disk. It can be turned on and off with a logical signal, and its rotation speed and direction are controlled by a numeric value.
Switch object class	A Switch object can represent its two positions using standard switch symbols or custom graphics.
Pushbutton object class	A Pushbutton object looks like a button that changes when depressed, but you can also make it transparent. You might use transparent pushbuttons over custom graphics.
Pot object class	A Pot object can be displayed as a knob, vertical slider, horizontal slider, increment and decrement buttons, or a digital number.
Logical expression	When you create an expression that results in a logical value, you can represent that value using standard lights, text, or custom graphics.

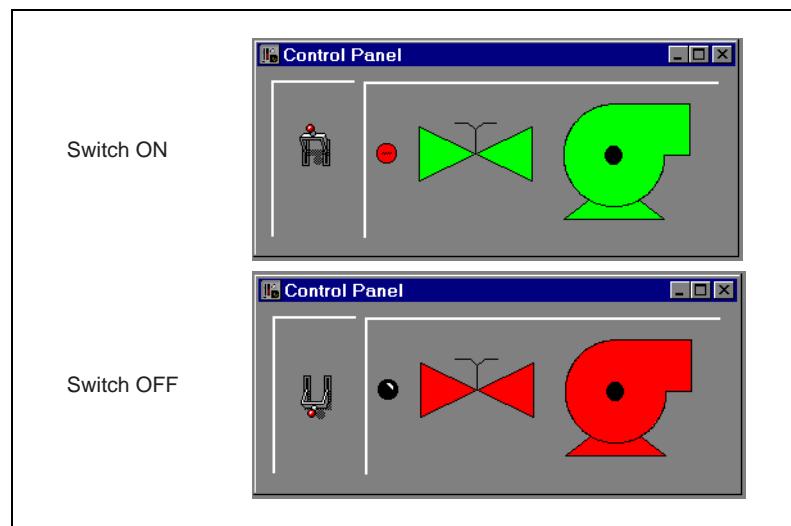
Table 2-1. Tools for Displaying Dynamic Graphics (Continued)

LookoutDirect Component	Description
Numeric expression	When you create an expression that results in a numeric value, you can represent that value using a digital number or a vertical or horizontal bar chart.
Text expression	When you create an expression that results in a text value, you can represent that value using any style and size font loaded on your computer.

For information on a specific object class, refer to the online help or the online PDF *LookoutDirect Object Reference Manual*. For information on expressions, refer to Chapter 1, *Expressions*.

Displaying Dynamic Logical Signals

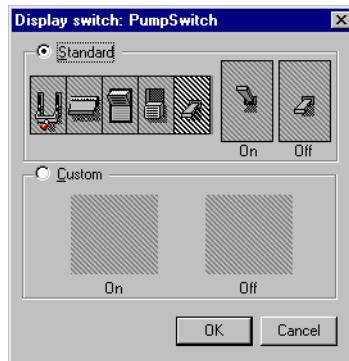
Logical expressions and Switch objects are two commonly used graphical animation tools. You can display the signal a Switch object generates in a variety of ways.



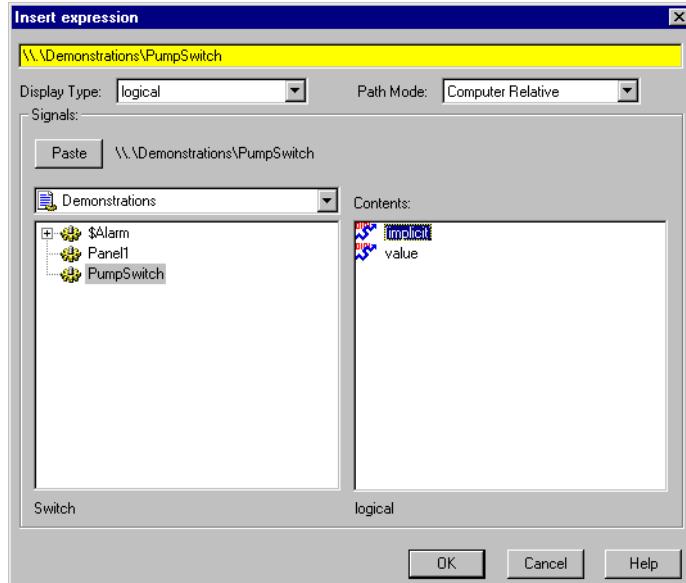
The Switch is shown in the On and Off positions. The graphics on the right side of the control panel show three ways of graphically displaying the logical signal generated by the Switch. The graphics representing the Switch signal are expressions of the (*implicit*) value of the switch, created through the **Insert»Expression** menu command.

To create a Switch object, first choose the logical signal that you want to represent with dynamic graphics (for example, a logical input from a PLC). Then, select an object to display the logical signal. For this example, use a Switch to simulate a contact from a motor starter relay.

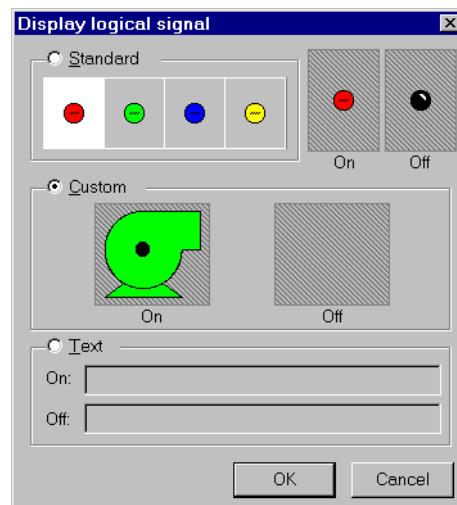
Create a Switch and call it PumpSwitch. After you define the Switch parameters, LookoutDirect presents the display parameters dialog box for the Switch object (as shown in the following figure). You can choose to represent PumpSwitch with one of the standard graphics, or you can use custom graphics. For this example, select a standard graphic and click on **OK**.



To insert an expression representing the signal generated by the Switch object, use the **Insert>Expression** command and select PumpSwitch, or drag and drop the signal from the LookoutDirect Object Explorer.



When you click on **OK**, the **Display logical signal** dialog box appears. From the **Display logical signal** dialog box, select display characteristics for a logical signal expression.



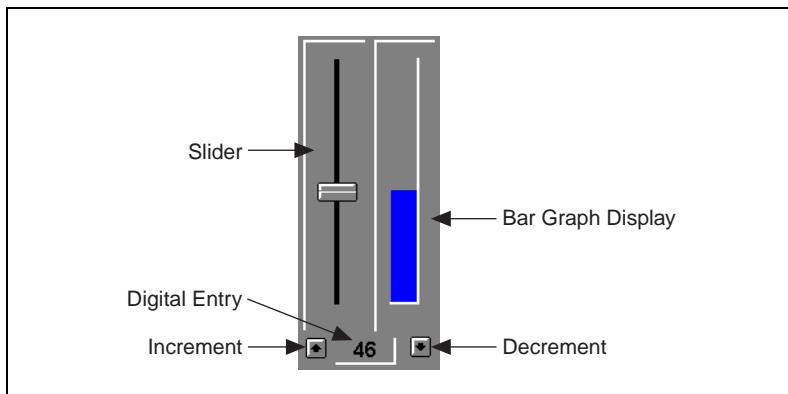
Click on the **Custom** selection, and then on the **On** and **Off** list boxes, scrolling through the choices until you find the appropriate graphic. Your **On** and **Off** graphic selections appear as thumbnail sketches in the two preview windows to confirm and verify your selection.

Click on **OK** and test your example by flipping the switch. The graphics should change according to your selections in the **On** and **Off** preview windows. To customize your switch, experiment with the **Text** selection in the **Display logical signal** dialog box.

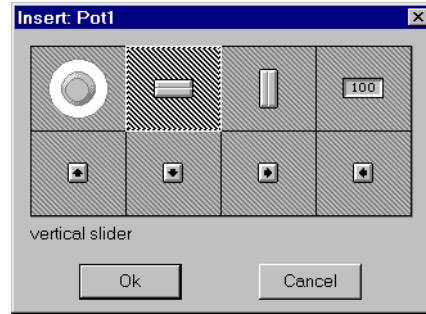
Use this same basic method to display any dynamic signal on a control panel. If you want to represent more than two conditions with dynamic graphics, use a Multistate object. Similarly, you can use a Pipe object to make a line or rectangle dynamically change colors and blink.

Displaying Dynamic Numeric Signals

You can use a Pot object to display dynamic numeric signals. The following illustration shows the Pot as a slider, a digital entry, and increment and decrement buttons—all are graphical representations of the same Pot object.



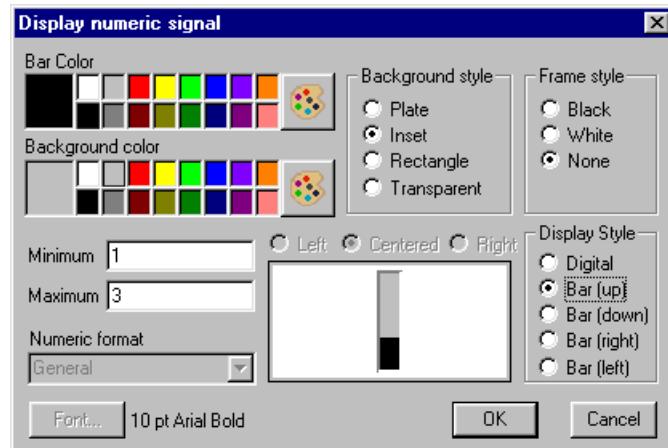
To use a Pot object to display a signal, first choose the numeric signal that you want to represent with dynamic graphics, such as an analog input from a PLC. Then, select an object to graphically display the dynamic numeric signal. For this example, create a Pot object and name it PumpSpeed. After you define the Pot parameters, a display parameters dialog box appears. In this example, the dialog box is named **Insert: Pot1**. You can choose to represent the Pot with any of the standard graphics.



Use the **Insert»Expression** command (or click and drag) and select PumpSpeed to insert an expression that represents the signal generated by the Pot object.

When you click on **OK**, the **Display numeric signal** dialog box appears. From this dialog box, select the display characteristics for a numeric signal expression.

If you choose **Digital** display style, you can use the **Font** button to select the desired font style and size. You can also specify a **Numeric format** for the value. For more information on numeric formats, refer to the *LookoutDirect Getting Started Guide, Chapter 3, Getting Started with LookoutDirect*



Click on **OK** and test your example by adjusting the Pot. The graphics should change according to your selections.

If you want to represent the numeric signal with moving custom graphics, use the Animator object class.

Displaying Dynamic Text Signals

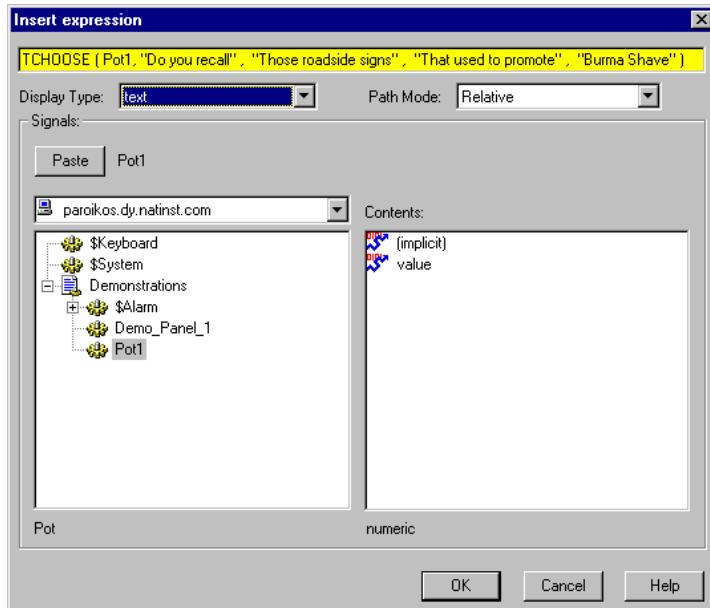
You can display dynamic text messages on a control panel with text expressions. You can easily display up to two separate text messages using logical signals, but there are times when you need to display three or more separate text messages in a single statement. If you have more than two separate messages, a numeric signal might determine which message to display.

To graphically display a dynamic text signal, first choose the numeric signal that you want to use to control which message is displayed, perhaps an analog input from a PLC. Then, select a graphical object to display that signal. This example uses a Pot object.

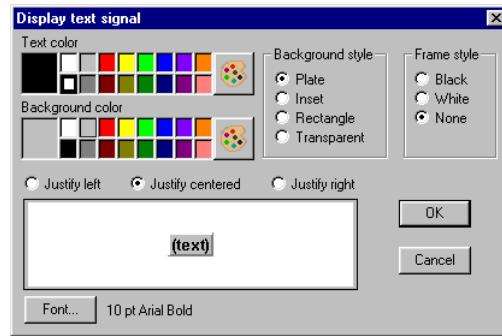
Create a Pot object and call it Pot1. Define the Pot minimum, maximum, and resolution parameters as 1, 4, and 1 respectively. Select the **Insert»Expression** command and enter the following expression in the dialog box that appears.

```
TCHOOSE(Pot1, "Do you recall", "Those roadside signs",
"That used to promote", "Burma Shave").
```

See Chapter 1, *Expressions*, for information on the *TCHOOSE* function.



When you click on **OK**, the **Display text signal** dialog box appears. From this dialog box, select display characteristics for a text signal expression. Use **Plate** as your background style.



Click on **OK** and test your example by adjusting the Pot. The text should change according to your expression.

Finding a Lost Graphics File

LookoutDirect uses a standard missing graphics bitmap to mark a location on a panel where a graphic should be present but cannot be found in the location *LookoutDirect* expects. The status bar at the bottom of the *LookoutDirect* screen displays the path and filename of the missing graphic.

Right-clicking on the missing graphic bitmap displays a standard screen that also tells you the name of the missing graphics file and the previous path.

To restore the look of your panel, either put a new copy of the graphic in the location in which *LookoutDirect* expects to find it, or delete the missing graphic bitmap and place a new copy of the graphic from its current location.

Creating Custom Graphics

There might be times when none of the Lookout*Direct* standard graphics exactly fits your needs. You can use any drawing software to create your own custom graphics. Lookout*Direct* supports both Windows Device-Independent Bitmaps (.BMP) and Windows Metafiles (.WMF).

After you create a custom graphic file, copy it to the GRAPHICS subdirectory of your choice in your Lookout*Direct* root directory. Because Lookout*Direct* can use only graphics located in the GRAPHICS directory and subdirectories, keep all standard and custom graphics in the GRAPHICS subdirectories. Lookout*Direct* can access your custom graphic file any number of times in the same or different process files from the GRAPHICS subdirectories.

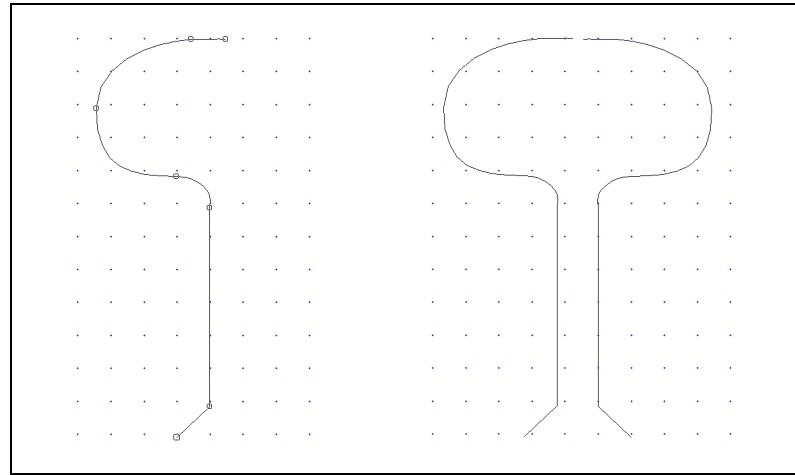
Creating Custom Graphics Example

This section describes how to create custom graphics for use in Lookout*Direct*. This example sketches out the creation of an elevated tank graphic you can use to show water level in a real tank. You export that tank graphic to Lookout*Direct*, implement the display in Lookout*Direct*, and test it. The example might only parallel what you would have to do with your own graphics creation program, but should serve as an illustration of the important points.

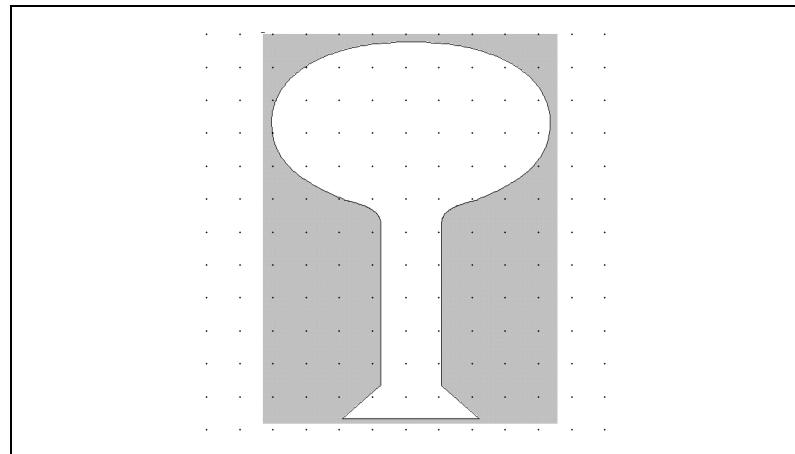
If you have not already mastered using a drawing program, you should allow yourself time to become accustomed to using the drawing application of your choice. The illustration below is from a third-party drawing application.

Creating the Graphic

1. Draw half of the tank.
2. Copy the tank half and mirror the image to create the other half.
3. Connect and refine the halves to create an enclosed tank object.



4. Refine your image until it looks the way you want.
5. Draw a rectangle around the tank and connect both the rectangle and tank into a single object.
6. To match the background color of the mask to the control panel, select the object group and click on the gray color bar selection.
7. Select invisible for the line style.

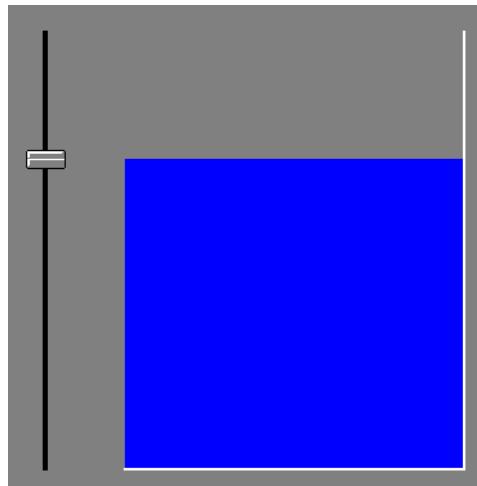


Save or Export the Graphic and Place in Lookout*Direct*

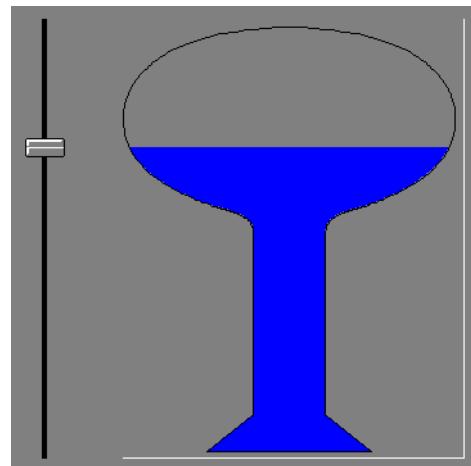
1. Save or export your new graphic from the drawing program as a Windows Metafile (.WMF) in the directory C:\LOOKOUT\GRAPHICS\TANKS and name your new graphic ELEV TANK.WMF.
2. Export.

Testing the Graphic in Lookout*Direct*

1. Launch Lookout*Direct*.
2. Create a Pot object using the **Object»Create** command.
3. Select **Insert»Expression**, choose the Pot object, and click on **OK**.
4. In the **Display numeric signal** dialog box, select **Bar (up)** for the **Display style**, **Rectangle** for the **Background Style**, and click on **OK**.



5. Use the **Insert»Graphic** command to insert your new graphic, ELEV TANK.WMF, over the bar graph.
6. Position the graphic over the bar graph and stretch it to size.
7. Toggle out of edit mode and test your example by moving the slider up and down.



Graphic File Types

LookoutDirect accepts both Windows Device-Independent Bitmaps (.BMP) and Windows Metafiles (.WMF).

Bitmaps

Bitmaps are raster files, made up of differently colored pixels. Because each pixel in the bitmap takes up one pixel on the screen, bitmap images are always rectangular and never resizable. You can display raster images on control panels that are not rectangular in *LookoutDirect* by using transparent pixels for the parts of the rectangle you do not want to show.

Bitmap files typically have a .BMP file extension. The Paint program that comes with Windows can read .PCX bitmap files (another very common bitmap format) and convert those files to Windows bitmaps.

Metafiles

Metafiles are vector files, consisting of coordinates that are connected by lines and curves, as well as area-fill commands. A vector file can consist solely of two sets of coordinates connected by a line or a complex set of colored area fills and colored lines and curves to create line art images. Metafile images are not necessarily rectangular.

Because metafiles contain a set of coordinates, they can be resized and stretched to any size or aspect ratio. Microsoft does not add information in the basic metafile file format for metafile size information. The Aldus Corporation (authors of PageMaker) created a metafile header that contains metafile size information. Most software packages that generate metafiles also add this header information. Without the header, *LookoutDirect* cannot maintain the correct aspect ratio (width to height ratio) for a metafile. To determine if a metafile has this information, resize the graphic on a control panel while holding down <Ctrl>. If aspect information is available, *LookoutDirect* does not stretch the metafile drawing out of proportion.

Bitmaps or Metafiles?

Which format is better: bitmap or metafile? Both have strengths and weaknesses. You might want to use a combination of metafiles and bitmaps. Bitmaps effectively handle large background schematics or system overviews, and metafiles work well for individual pumps, valves, lamps, and other miscellaneous graphics.

Bitmaps usually display faster than metafiles. In fact, a complex drawing rendered as a bitmap can display 100 times faster than the corresponding metafile. If you want to display scanned photographic quality images with hundreds of colors, use bitmaps.

Metafiles are resizable—you can stretch them to any size or aspect ratio. Metafiles are typically smaller than bitmap files, so they take up less disk space and consume less memory than bitmap files. You can use one file that contains a metafile drawing of a pump to display several pumps of various sizes on the screen. With most drawing programs, you can save your line art images as metafiles. You can also copy the image on screen to the Windows clipboard and paste it into a paint program for bitmap conversion.

Memory Considerations

LookoutDirect loads each graphic into computer memory the first time it is displayed. The image remains in memory until *LookoutDirect* or another application needs more memory than is available. When more memory is needed, the graphic is discarded from memory and reloaded from disk the next time it is displayed, so that you can display more bitmaps and metafiles on the screen than can be held in memory at one time.

If you are running Windows in enhanced mode on a 386 or 486 computer, you have virtual memory. Windows uses virtual memory to swap memory images between RAM and disk, giving applications the appearance of more memory. If available memory becomes low and Windows must use virtual memory to handle applications and data between disk and memory, you might notice a slower application speed. For more information on virtual memory, refer to your Windows user guide. If your computer disk drive light flashes every time you pull up a new control panel in *LookoutDirect*, consider purchasing more RAM.

3

Serial Port Communication Service

This chapter describes serial communications and describes how to define settings for three different serial connections: hardwired, radio (RTS/CTS), and dial-up.

Introduction to Driver Objects

Certain object classes represent and communicate with external physical devices such as PLCs, RTUs, and controllers. A few examples include Modbus, Tiway, AB_PLC5, and Optomux. We use the generic term *driver* to refer to these types of object classes. The functionality built into driver objects enables them to communicate with the physical devices that they represent. *LookoutDirect* communicates with the outside world primarily through driver objects.

In traditional systems, drivers are separate applications that run independently of the operator interface. Driver programs compete for CPU time with applications such as database managers, HMIs, and historical data loggers, necessitating multitasking and increased CPU power. In contrast, *LookoutDirect* drivers are not separate applications. *LookoutDirect* driver objects work as any other object in the *LookoutDirect* event-driven environment, except that they communicate with external devices.

With traditional systems, you assign a particular driver to a specific serial port. In such configurations, multiple drivers cannot share a single serial port. *LookoutDirect* does not associate baud rate, data bits, parity, or stop bits with a particular serial port. In this way, drivers that implement different protocols and baud rates can use the same port and the same modem or radio frequency.

This capability allows you to mix and match RTUs, PLCs, and other devices over a single radio frequency without communication conflicts or special hardware. For example, you can use a single two-way radio connected to a serial port to communicate with several different brands of RTUs out in the field, each one using a different protocol. You can have seventy-five remote PLCs share a set of five dial-up modems.

All this is possible because of the Lookout*Direct* port communication service. Objects use the communication service, an environment service, to gain access to serial ports in an orderly and timely fashion.



Note Some Lookout*Direct* driver objects communicate with physical devices through dedicated hardware. These driver objects do not use serial ports but instead rely on their own proprietary network cards to communicate with the outside world. A few examples include Modbus Plus (SA85 card), Data Highway (KT card), and DeltaTau (PMAC card). You do not need to configure serial ports for these objects classes. Refer to the appropriate object class documentation in you online help or in the PDF *Lookout*Direct* Object Reference Manual* to verify if a particular object class uses a serial port.

Understanding the Communication Service

The Lookout*Direct* serial communication service allocates serial port usage between driver objects. At the frequency of the object **Poll Rate**, a driver object notifies the communication service that it needs to use a specific serial port to poll a device. If the requested serial port is not in use, Lookout*Direct* allocates the serial port to the driver object. When the driver object takes control of the serial port, it defines port communication parameters such as baud rate and protocol and polls its device. When polling is complete, the driver object releases the port so the communication service can allocate it to other driver objects.

You can uniquely configure each serial port for hardwired, radio, or dial-up communications through the **Serial Port Settings** dialog box. Refer to the *Defining Serial Port Settings* section for more detailed information.



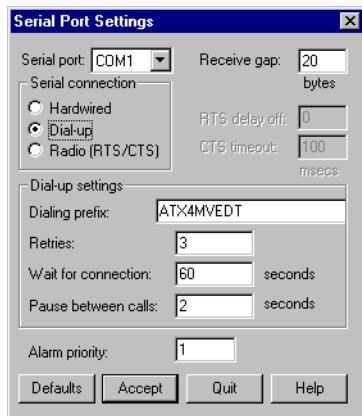
Note You must define serial port communication settings on every copy of Lookout*Direct*. If you have more than one instance of Lookout*Direct* running on the same computer, each instance must use a different serial port.

Because multiple Lookout*Direct* applications cannot share the same serial port, if it is necessary for two processes to access the same serial port, they need to be run in the same Lookout*Direct* application. If you are designing a system with multiple server process files, and those server files all access the same serial port on a single computer, they need to be run in the same Lookout*Direct* application.

Defining Serial Port Settings

This section walks you through the steps to configure serial port settings for hardwired, radio, and dial-up communications.

1. From the Lookout*Direct* menu bar, select **Options>Serial Ports**. The **Serial Port Settings** dialog box appears.



2. In the **Serial port** data field, select the communication port you are defining.
3. Define the serial port parameters for the appropriate communication port. The rest of this section contains complete descriptions of the parameters.
4. Click on **Accept** to save the parameter changes for the serial port.
5. Click on **Quit** to exit the dialog box.

Selecting the Serial Port

Use the **Serial port** field to select the communication port you are defining. Microsoft Windows supports up to nine serial ports; however, most computers support only two serial ports without additional hardware.

Setting Receive Gap

The **Receive gap** setting is available for all serial connection types. This number specifies the number of empty bytes (or amount of time) a driver receives from a controller before the driver recognizes the end of a message frame and asks for another message. Normally you should leave this at the default setting of 5. However, if you are experiencing garbled communication alarms, you might try increasing this number to allow more dead time before

Lookout*Direct* decides it has received a complete message. For example, with a slow baud rate of 1200, you might have to increase the **Receive gap** setting to approximately 30.

Selecting the Serial Connection

Hardwired Settings

Hardwired serial connections require no hardware handshaking for line control. Use this setting for all serial communication types except dial-up telephone and remote radio transceivers. You should also use this setting when directly connecting Lookout*Direct* to the master repeater on a radio system or through a leased-line modem. Because a master repeater is a full-duplex device that does not require keying and unkeying of the frequency, it acts much like a physically hardwired network. Other hardwired connection types include RS-232, RS-422, RS-485, and leased telephone lines.

RTS/CTS Handshaking Settings

RTS/CTS is a local hardware handshaking mechanism between the local computer and the local communication device. Use the **Radio (RTS/CTS)** serial connection when you connect the serial port to a device that requires RTS/CTS hardware handshaking, such as a radio transceiver that must be keyed up during data transmission and unkeyed during data reception. Other half-duplex communication media, such as RS-485, might require RTS/CTS hardware handshaking. Although the RTS/CTS scheme works identically for other RTS/CTS communication schemes, this example assumes that you are communicating through radio.

When you select RTS/CTS hardware handshaking, Lookout*Direct* controls the RTS, or request-to-send pin, and monitors the CTS, or clear-to-send pin, during data transmission (pins 4 and 5 on a 25-pin RS-232 connector). Therefore, you must have at least the RTS pin (pin 4) wired *straight through* on your RS-232 cable. The CTS pin (pin 5) is optional.

Lookout*Direct* initiates a serial transmission on an RTS/CTS port by first asserting RTS to key the radio. Lookout*Direct* then begins monitoring the state of the CTS pin. When the radio transmitter is fully keyed and ready to transmit, the radio asserts CTS and Lookout*Direct* immediately begins data transmission. If the radio does not assert CTS within the **CTS timeout** setting (default is 100 ms), Lookout*Direct* assumes the radio is ready to transmit and transmits anyway.

The **CTS timeout** setting is the maximum amount of time that *LookoutDirect* waits after asserting RTS for CTS before transmitting. Most radios typically take between 10 and 80 milliseconds to key up. Consult your radio specifications and DIP switch settings to determine the key-up delay on your radio.

If your radio can assert CTS when it is ready to transmit, add about 50 milliseconds to the radio key-up delay specification and use this total value for the **CTS timeout**. If your radio does not assert CTS, you should begin by adding about 20 milliseconds to your radio key-up time. Then, increase this value in 10 millisecond increments until the remote radio begins to correctly receive the first bytes of the message.

Some radios might assert CTS before they are actually ready to transmit. In this case, disconnect the CTS line (pin 5 on a 25-pin RS-232 connector) and set the **CTS timeout** to a value high enough to let the radio fully key before transmission.

After it transmits the last byte of data, *LookoutDirect* continues to assert RTS, keeping the radio keyed until the **RTS delay off** time period expires. You should set this value to the default of zero milliseconds so that *LookoutDirect* unkeys the radio as soon as possible to prepare to receive the response.

When unkeyed, most radios generate an audible squelch tail that the remote device might decode as unexpected garbage bytes. Some remote devices reject the entire message instead of just decoding the valid data and ignoring the extra garbage bytes. In this case, keep the radio keyed for several milliseconds using the **RTS delay off** setting. This time period delays the squelch tail long enough for the remote device to recognize the last data frame as valid before receiving garbage bytes caused by the squelch tail.

If you set the **RTS delay off** setting too high, the remote device begins transmitting its response before the local radio is unkeyed, causing a communication alarm in *LookoutDirect*.

Dial-Up Modem Settings

Use the **Dial-up** serial connection when you use a modem in conjunction with a switched telephone line (not leased line). You can customize the dial-up settings for your particular modem and phone line.

The default **Dialing prefix** settings are based on the Hayes Corporation AT command set, which is an industry standard for data modems. The following table explains the *LookoutDirect* default settings. For additional commands, refer to your modem operation documentation.

Table 3-1. Dialing Prefix

AT	Attention code that must precede all commands
D	Dial phone number with these modifiers: P for pulse; T for tone
En	Local echo mode: E for no echo
Mn	Speaker on or off: M for speaker always off
Vn	Verbal or numeric result codes: v for numeric result codes
Xn	Result code and dialing options: x4 waits for dial tone before dialing, and recognizes busy signal

When you use an external dial-up modem with Lookout*Direct*, the DTR line in your cable between the modem and the computer must be wired straight through. This line is pin 20 on a 25-pin RS-232 connector and pin 4 on a 9-pin connector. Lookout*Direct* uses the DTR line to command the modem to disconnect (hang up) and return to the command mode.

Some factory modems are not configured to respond to the DTR line. After Lookout*Direct* first successfully dials out to a remote modem and finishes the polling cycle, it drops the DTR line but the modem remains connected. If the modem does not respond after several seconds of Lookout*Direct* attempting to raise and drop the DTR line, Lookout*Direct* generates an alarm stating that the modem is not responding. If you receive this alarm message, your modem is not configured to monitor the DTR line.

The Hayes Corporation standard command for configuring the modem to hang up and enter command mode upon loss of DTR is &D2. You can use a terminal program to make this setting permanent on most modems by entering the modem command AT&D2&W to store the setting permanently in nonvolatile modem memory. Or you can just add &D2 into the **Dialing prefix**. The default **Dialing prefix** is ATX4MVDET, so you might change it to AT&D2X4MVDET.

Retries specifies the number of times Lookout*Direct* dials the specified phone number and attempts to connect to the modem at the other end of the line. If Lookout*Direct* fails to connect after the specified **Retries**, it generates an alarm and moves on to the next phone number in the polling queue (if a queue has formed).

Wait for connection specifies the length of time Lookout*Direct* waits to receive a connect signal back from the modem it is calling. The time period begins when Lookout*Direct* first sends the local modem the dialing prefix command. The time should be long enough for the local modem to receive a

dial tone, dial the phone number, allow the remote modem to pick up the line, and send back a connect message. If the specified time is too short, your system could be operating correctly but never make a connection.

Pause between calls is the length of time *LookoutDirect* waits after hanging up before it sends the local modem the next dialing prefix signal. If the specified time is too brief, your system might not hang up the existing call but still attempt to call the next number.



Note Your specific modems, radios, and local phone lines might operate faster or slower than the default settings. You might need to use a trial-and-error approach to find the best settings for your system.

Serial Port Hangup

You can configure your serial port to use +++ATH hangup as well as DTR hangup.

Every serial port you have configured will have a configuration section in the *LookoutDirect.ini* file under the port name, such as [COM1]. Add the following entry to the file to set your hangup mode:

```
DTR_Hangup=N
```

When $N=1$ (default), that port uses DTR hangup. When $N=0$, the port uses +++ATH hangup.

Serial Port Diagnostics

You can create serial port diagnostic files to help solve serial port problems. Open the *LookoutDirect.ini* file (located in your *LookoutDirect* directory) with a text processor. Every serial port you have configured has a configuration section in this file under the port name, such as [COM1]. Add the following entry to the file to create a diagnostic file

```
Diagnosticfile=N:\completepath\filename
```

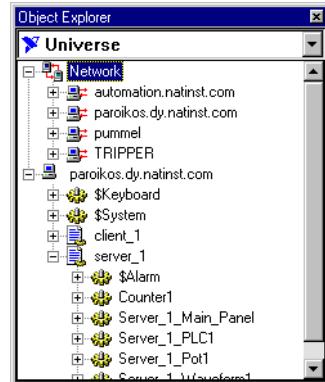
where N is the drive letter, followed by the complete path to the file (including the file name) you want to hold your diagnostic information. After editing the *Lookout.ini* file, reload your *LookoutDirect* process file to force *LookoutDirect* to reread the .ini file.

4

Networking

With *LookoutDirect*, you can monitor and control your process from any workstation (node) on the network.

LookoutDirect is designed to make networking easy. When you want to display an expression or make a connection from one process to another, you use the following dialog box, which appears in one form or another in many *LookoutDirect* dialog boxes, as well as in the *LookoutDirect* Object Explorer.



Each computer registered as running *LookoutDirect* appears in this dialog box. Click on the computer to open the full display and reveal the processes currently running on that computer. Click on a particular process to reveal all the objects in that process that you can access.

After you navigate to the computer and process you want to make a connection to, you work as you would with any local *LookoutDirect* object.

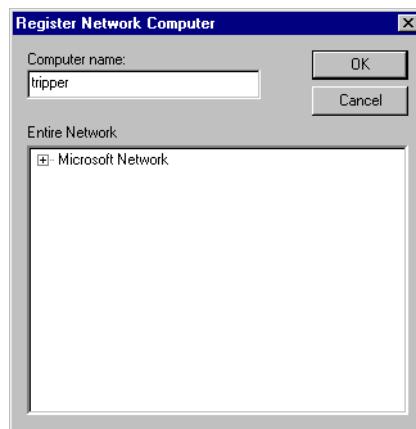
Registering Computers

Use the Lookout*Direct* Connection Browser or Lookout*Direct* Object Explorer to register the computers running Lookout*Direct* processes on your network.

In addition to registering computers, both the Object Explorer and the Connection Browser perform a number of useful functions in Lookout*Direct*. These other functions are detailed in Chapter 3, *Lookout*Direct* Basics: Windows, Tools, and Files*, of the *Getting Started With Lookout*Direct** manual.

To register the computers using Lookout*Direct* on your network, open a process. Select **Object»Connection Browser** or **Object»Object Explorer**.

Right-click on Network, and select **Register Network Computer**. The following dialog box appears.

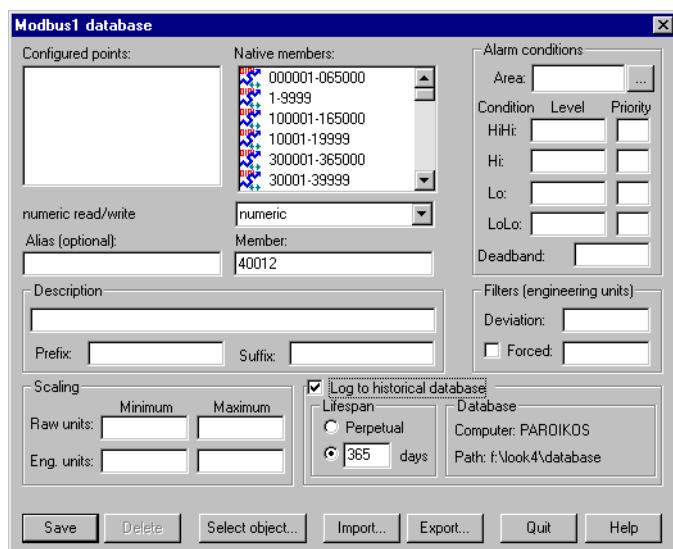


Type the name of the computer in the **Computer name** field. If you are unsure of the spelling, you can browse the network for computers by clicking on the network node in the **Entire Network** field.

To remove a computer from your list of registered computers, select **Object»Connection Browser** or **Object»Object Explorer** and double-click on Network. Highlight the name of the computer you want to remove, and select **Unregister Network Computer**.

Logging Data

Before you can log data across the network, you must select the **Log to historical database** option for the data member you want to log. Choose **Object»Edit Database** to select this option. The dialog box you see might vary from the one shown here, depending on what object you edit the database for. For more information, refer to Chapter 3, *Getting Started with LookoutDirect* in the *LookoutDirect Getting Started Guide*.



Time Synchronization

To keep your data properly time stamped, you must make sure the times on your computers are properly synchronized. The *LookoutDirect* time synchronization service is installed as a service in Windows NT that runs every time you run your computer. Time synchronization runs as a background process in Windows 98/95.

To configure time synchronization, select **Options»Time Synchronization**. The following dialog box appears.



Any computer that is running the time synchronization service can serve as a time server or a time client. The primary time server is the first computer listed in the **Time Server Search Order** field.



Note You must make sure that the order of search for time servers is the same for all the computers running *LookoutDirect* on your network, including the primary time synchronization server.

You do not include a computer running *LookoutDirect* in its own list of time synchronization services.

Suppose you have four computers you need to have synchronized. If one fails, the others look for the next in line to synchronize to as time servers. For computers A, B, C and D, you would use the following time server search order in each computer.

Table 4-1. Time Synchronization Order

Computer A	Computer B	Computer C	Computer D
None listed	A	A	A
		B	B
			C

As the primary time server, Computer A would have no other servers listed. As long as Computer A is running, it should synchronize to itself. Computer B should synchronize to Computer A as long as A is running. If A is not running, B should synchronize to itself. Computer C should synchronize to Computer A if it is running, Computer B if A is not running, and to itself if neither A nor B is running. This pattern should be used for all the computers you want in one synchronized set.

To change the order in which your computers search for a time synchronization server, select the computer name and click on the **Up** or **Down** buttons.

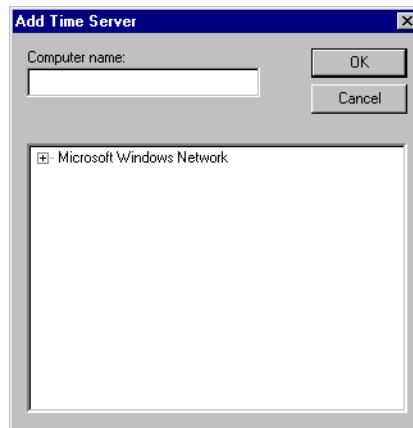
Use **Sleep Time (seconds)** to set how long each computer waits between each synchronization. You should set the primary time synchronization server sleep time to 60 seconds.

If your primary server is off-line for some reason, a computer scheduled to synchronize automatically seeks out the second computer on the synchronization server list. At the time of the next synchronization, the computer first looks for the primary server before seeking a secondary synchronization server.



Note If you have some computers running Windows 98/95 and other computers running Windows NT in your network, you should list your Windows NT machines first in the server search list. Time synchronization works better between Windows 98/95 and Windows NT systems when the Windows NT computer is the server.

To add a computer to the **Time Server Search Order** field, click on the **Add** button. The following dialog box appears.



If you know the name of the computer you want to add, you can type it into the **Computer name** field. If you do not know the exact name of the computer, you can browse for it in the network tree contained in the second field.

To remove a computer from the **Time Server Search Order** field, highlight the computer name and click on the **Remove** button.

If no computer is set as a primary time server, your computer synchronizes to itself.

Checking the Network Connection between Two Computers

LookoutDirect networking, like the ping program, is based on TCP/IP and assumes that computer network addresses can be resolved by name. Use the ping program to see if two computers are properly networked before trying to run *LookoutDirect*.

Run ping by opening a DOS window and entering
`ping server`
from the client and
`ping client`
from the server.

If you cannot successfully run ping from both computers, *LookoutDirect* networking will not work.

When ping runs successfully, it produces output similar to the following:

```
>ping plato
Pinging plato.natinst.com [123.45.67.89] with 32 bytes of
data:
Reply from 123.45.67.89: bytes=32 time<10ms TTL=128
```

If ping succeeds, but only after a long time, *LookoutDirect* will also take a long time to make connections.

One possible cause of a slow response is an incorrect domain suffix search order. Check this in your **DNS configuration** dialog box, and correct the search order if necessary. To access DNS configuration in Windows NT 4.0, select **Start»Settings»Control Panel»Network»Protocols**. Click on **TCP/IP Protocol** to open the **TCP/IP properties** dialog box. Select the **DNS** tab to check the search order.

To access DNS configuration in Windows 98/95, select **Start»Settings»Control Panel»Network**. Select the **Configuration** tab, and click on **TCP/IP** in the list of installed components. The **TCP/IP properties** box appears. Select the **DNS Configuration** tab to check DNS search order.

Under Windows NT 4.0, open a DOS shell and enter the command
`nslookup computernname`

to see if DNS is working and to find out how long it takes to do the name lookup.

If `ping` does not succeed, try accessing the Network Troubleshooter by selecting **Start»Help»Troubleshooting**. This Windows utility can sometimes diagnose a problem. In some cases, it might be necessary to use the machine's fully qualified name, such as `plato.dy.natinst.com`.

If the Network Troubleshooter does not resolve your problems, you might need to ask for help from a network administrator.

Chapter 4 *Networking*

5

Dynamic Data Exchange

This chapter explains how to use Dynamic Data Exchange (DDE) with Lookout*Direct*. DDE is the Microsoft message-based protocol used by applications like Microsoft Excel and Lookout*Direct* to link to data in other applications.

When the data in a source application changes, it dynamically updates all linked data (in real-time). With DDE, you can dynamically link other Windows applications to Lookout*Direct*.

There are several DDE protocol formats. Lookout*Direct* supports the standard Microsoft formats, XITable and CF-TEXT. XITable is often referred to as the Fast table format; CF-TEXT is often called text format. Lookout*Direct* also supports hot DDE links and NetDDE.

Any two applications participating in dynamic data exchange are engaging in a DDE conversation. In such a conversation, Lookout*Direct* acts as either the client application or the server application (or both, in a peer-to-peer configuration). If Lookout*Direct* is getting data from another application, Lookout*Direct* is the client. But if another application is getting data from Lookout*Direct*, then Lookout*Direct* is the server.

The client application is responsible for establishing a DDE link with the server. When Lookout*Direct* is a client, it first tries to establish an XITable DDE connection (because this is the most efficient). If the server application does not support this format, Lookout*Direct* uses the CF-TEXT DDE format.

To establish a DDE link, the client application must identify the location of the desired data. A three-tier address identifies the location of the data: Service, Topic, and Item. Look in the application documentation to determine its service, topic, and item.

Service specifies the name of the server application the client is linking to. Each application that supports DDE has a unique service name. For example, Lookout*Direct* is the service name of Lookout*Direct*, and EXCEL is the service name of Microsoft Excel.

Topic is the second level in the three-tier address. For many server applications like Excel and Lookout*Direct*, topic specifies a particular file.

In *LookoutDirect*, the topic is the process file name, minus the .LKP extension. For example, you would refer to a process file named PLANT.LKP as Plant when using it as the topic in a DDE link.

Item identifies the specific data or value being linked between the server and the client. A *LookoutDirect* item is the object name, followed by a specific data member (such as Name.datamember) if needed. See *Identifying Object Data Members* in Chapter 4, *Using LookoutDirect*, of your *Getting Started with LookoutDirect* manual for detailed information on selecting objects and data members. An item in a spreadsheet, such as cell B3 in Microsoft Excel, would be r3c2.

Linking *LookoutDirect* to Other Applications

LookoutDirect can act as a DDE client, DDE server, and both DDE client and server. Therefore, there are three basic ways to link *LookoutDirect* to another application using DDE:

- *LookoutDirect* as the server
- *LookoutDirect* as the client
- *LookoutDirect* as both client and server (peer-to-peer)



Note All readable numeric, logical, and text values in *LookoutDirect* are automatically available to any other application through DDE. No special setup is required.

Because Microsoft Excel is widely used and accepted, it is used in the *LookoutDirect* DDE examples.

DDE Server Example

In this example, you can send information from *LookoutDirect* to another application, making *LookoutDirect* the server. First, create a potentiometer in *LookoutDirect* so you can link its value in real-time to a cell in Excel. Any time the Pot is adjusted, the value in the spreadsheet cell automatically changes.

1. Make sure *LookoutDirect* is not in edit mode.
2. Hold down the <Ctrl> key and click on the object you want to link to. In this case, select the Pot object you just created.
3. *LookoutDirect* beeps when it successfully copies the object value to the clipboard. The object can be a slider, bar graph, switch, pushbutton, digital display, text entry object, knob or almost anything else in *LookoutDirect* that contains a value.
4. Start Excel and select the cell you want to link to.

5. If your version of Excel is 5.0 or later, select **Edit»Paste Special**, then click on **Paste Link**. If you have an older version of Excel, select **Edit»Paste Link**.

You have just created your first DDE link. Repeat this process as many times as you need. If you are linking large numbers of objects to Excel, you might want to use the Excel copy and edit tools to speed up the process.

Not all applications support the Windows clipboard shortcut method as described above. Therefore, you might have to manually enter the appropriate *LookoutDirect* service, topic, and item in the other application to create a DDE link to that package. The format in which you enter this information varies from one package to another. For this reason, you should refer to the documentation of the client application for instructions.

DDE Client Example

In this example, you import information from another application into *LookoutDirect*. For instance, you might want to use a value calculated inside a spreadsheet as a process control setpoint for a *LookoutDirect* application. In this kind of DDE link, *LookoutDirect* is the client and the spreadsheet application is the server. Because *LookoutDirect* is the client, it is responsible for establishing the link to the server data. Therefore, you must identify the service, topic and item in *LookoutDirect*. These are object parameters in the *DdeLink* object class.

1. Select **Object»Create** and select the *DdeLink* object class.



2. In **Service**, enter the name of the software package (Excel in this example).
3. In **Topic**, enter the name of the spreadsheet file.
4. In **Item**, enter the address of the cell you want to read a value from. Notice that the entered cell address is r1c1. This translates to row1/column1 (cell A1) in Excel. The Excel DDE structure requires this format.

5. Click on **OK**, and then select **OK** again when Lookout*Direct* prompts you to insert the expression `DDE1`. Finally, pick the desired display format and click on **OK**.

To test your link, enter a numeric value into cell A1 of your spreadsheet. Whatever value you enter into the spreadsheet is immediately written to the DDE expression on your panel. You can also connect the `DdeLink` object you just created to other Lookout*Direct* objects. Refer to the `DdeLink` definition in the online help for more information.

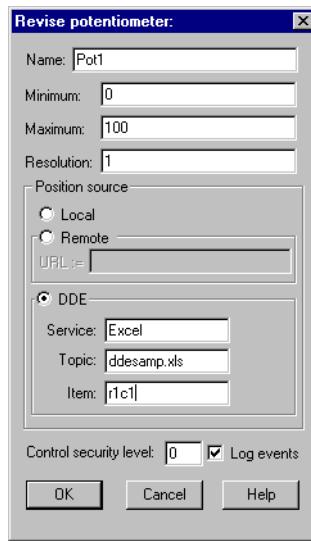


Note The `DdeTable` object is another way of linking data to Lookout*Direct* using DDE. This object class links large quantities of data though the more efficient XITable format. Refer to the `DdeTable` definition in the online help for more information.

DDE Peer-to-Peer Example

Assume you want to take the Lookout*Direct* as a DDE server example one step farther. Suppose you want to adjust the Pot to change a value in Excel and also be able to enter a different value in Excel to adjust the Pot. That is, you want to send data both ways through a DDE link. You can easily create such two-way links for user-controlled objects (that is, Switches, Pots, and Pushbuttons).

Select **Object»Create** and define a new Pot; or select **Object»Modify** and select the existing Pot object.



1. Change **Position source** from **Local** to **DDE**.

2. In **Service**, enter the name of the software package (in this case, Excel).
3. In **Topic**, enter the name of the data file.
4. In **Item**, enter the address of the cell you want to read a value from, such as r1c1 (for cell A1 in Excel).
5. Click on **OK** to create or modify the definition of the object.
6. If the object is new, insert its display member into the panel so you can test your link.

To test your link, enter a value into the spreadsheet cell you specified and watch the Pot. Then adjust the Pot and watch the spreadsheet cell. You should see the values within the two applications change in unison.



Note If you link to a Pot object, the linked value is numeric, so you enter a numeric value into the spreadsheet cell. But if you link to a Switch or Pushbutton object, the linked value is logical. Linked logical values are shown in spreadsheet cells as `true` or `false`. To change the value of a logical value in a spreadsheet cell, enter `true` or `false`, `0` or `1`, or `on` or `off`.

DDE Alarms

The following section explains alarms that might appear in the *LookoutDirect* alarm window.

Cannot establish DDE conversation with <service>, <topic>

This alarm occurs if a *LookoutDirect* client is unable to connect to the server corresponding to the given service and topic. The alarm also occurs if the server terminates the conversation (for example, if the server is shut down). The alarm is deactivated when the *LookoutDirect* client successfully connects to an item on the server.

Verify that the service and topic were typed correctly when you created the object that is using DDE. Verify that the server application is running. If the server is on another computer on the network, verify that the network is up. If the server is on a computer running Windows NT, verify that you are authorized to log on to that computer and that the current user logged onto the NT machine has trusted the DDE share to which you are trying to connect. For more information, refer to the section *Adding a Trusted DDE Share* in Appendix A, *Networking With DDE*.

DDE client error for <service>, <topic>, <item>: (received NACK for advise)

DDE client error for <service>, <topic>, <item>: (received NACK for request)

Verify that the named item exists on the server and that the server supports DDE links for the item. This alarm occurs when Lookout*Direct* is a client.

DDE client error for <service>, <topic>, <item>: (received NACK for poke)

This alarm occurs if you are using Lookout*Direct* as both client and server, and have made a remote connection to an item that is not writable in a DataTable, Pushbutton, Pot, Switch, or TextEntry. The only Lookout*Direct* objects that support writes (pokes) are DataTable, PushButton, Pot, Switch, and TextEntry. These support writes into their implicit data members only.

If the server is running Windows NT, it is possible that the DDE share on the computer is configured to support reads (advises) but not writes (pokes).

DDE client error for <service>, <topic>, <item>: (advise timed out)

DDE client error for <service>, <topic>, <item>: (request timed out)

DDE client error for <service>, <topic>, <item>: (poke timed out)

Verify that the server application is running. Verify that the item exists on the server. If the server is on another computer on the network, verify that the network is up.

**DDE client error for <service>, <topic>, <item>: (received invalid data)
DDE server: corrupt data block poked to item <item>, topic <topic>**

Either the server received a corrupt data block from the client, or the client received a corrupt data block from the server. This might be the result of network trouble. If the alarm is consistent and predictable, you might have discovered a bug. Call National Instruments technical support for further help.

DDE server: failed to post advise for item <item>, topic <topic>

Verify that the client application is running. Verify that the item still exists on the client. If the client is on another computer on the network, verify that the network is up.

6

Security

This chapter describes the two types of Lookout*Direct* operational security: network security and control security. Viewing security is primarily based in control security. You can use either or both security approaches to control who has access to different processes, control panels, individual controls, and data.

Lookout*Direct* processes can pass data and commands back and forth across a network. Lookout*Direct* security prevents or enables this communication based on who is logged in on each instance of Lookout*Direct* running on the networked computers. Lookout*Direct* network security works between different instances of Lookout*Direct* running on one computer or different computers on the same network.

Network security is based on user and group permissions configured for processes, collections of objects grouped in a folder, or individual objects.

Control security is based on security level parameters set in a given Lookout*Direct* object, usually a control such as a Pot or Switch. This security level is compared to the security level assigned to a user account or a group to prevent or enable access.

A *user account* identifies a single person authorized to log on to Lookout*Direct*. *Groups* consist of collections of users with similar duties and security levels.

Security information for a Lookout*Direct* process is kept in the .1ka file for that process. You must keep the .1ka file in the same directory as the .14p file for your security settings to work. If you misplace the .1ka file, all users will have complete access to all parts of the process.

The user and group account information for Lookout*Direct* 4 is kept in the Lookout.sec file, installed in your Windows SYSTEM directory.

If you want basic authentication to work between different computers running Lookout*Direct* on your network, you must have an identical Lookout.sec file installed on each computer. You can do this by creating a master security file on your main development computer and copying it to all the other computers running Lookout*Direct* on your network.

To preserve user and group account information from versions of *LookoutDirect* prior to *LookoutDirect* 4.0, import the old security file into your new security file using the **Import LookoutDirect 3.x Security File** option in the *LookoutDirect* User Manager. See the Importing Old Security Files into *LookoutDirect* 4 section for more information on importing old security files.

Permissions and security levels are cumulative in *LookoutDirect*. If you add a user account to a group that has a group security level or permissions different than that assigned to the user account, the user will have the higher of the two security levels or permissions.



Note While it is possible to assign a security level to a user account and then put that user into a group with higher (or lower) security levels, it is not a good practice. To minimize confusion, it is best to assign user accounts to groups with the same security level when possible. Refer to the *Keeping Security Precedence Simple* section of this chapter for information on how different security levels and group permissions interact.

Logging On

LookoutDirect requires operators to log on with a predefined name and corresponding password (if any). To log on, use the **File»Log on** command, press <Ctrl-L>, or click on the **Account name** box in the status bar. Collectively, the name and password are known as a *user account*, or account. Each account has a security level and can be included as a member of a group. Because *LookoutDirect* uses account names when logging events to disk and when operators acknowledge alarms, you can identify the operator logged on when an event occurs.

You can programmatically access the name and security level of the currently logged *LookoutDirect* user through the \$System object, using the `username` and `secllevel` data members.

All users must log on to *LookoutDirect*. When nobody has logged on to *LookoutDirect*, *LookoutDirect* shows (nobody) as being logged on in the status bar at the bottom of the main screen. The (nobody) account is built into *LookoutDirect* with a security level of 0 (zero), and cannot be edited.



Note If the (nobody) account is logged on, any functions of client or server processes running that require a security level greater than zero do not receive or report data until someone logs on using an account with a high enough security level.

The first time you start a development version of *LookoutDirect*, the Administrator user account has no password. Any time the Administrator

account has no password, *LookoutDirect* opens with Administrator logged in and active, without requiring any login. This is a convenience for you when creating your *LookoutDirect* processes, but you must be sure to assign a password to the Administrator account before allowing others who should not have Administrator privileges to use your copy of *LookoutDirect*.



Note Server and client run-time versions of *LookoutDirect* open with the (nobody) user account logged in, no matter what the password setting for the Administrator account.

After you have added a password to the Administrator account using the *LookoutDirect* User Manager, development versions of *LookoutDirect* open by presenting the **Welcome to LookoutDirect** dialog box which requires a *LookoutDirect* user to log in.



Each time you log on, enter your **User name** and **Password**.



Note If yours is the only account that is a member of the Administrators group, and you forget your password, there is no way to access the **System»User Manager** command, and there is no way to modify account settings. Contact National Instruments for assistance.

Domain—In current versions of *LookoutDirect*, you can only log on to your local domain.

Idle time—The amount of time the computer sits idle (no mouse movement or keyboard action) before *LookoutDirect* automatically logs off the operator. If you enter 0 (zero), idle time is disabled. For security reasons, you might want to use this feature to automatically log off high-level accounts if the computer is left unattended too long. After an account logs off, the account (nobody) is logged on. The (nobody) account has a security level of 0 (zero).

By default, *LookoutDirect* presents you with the login dialog box every time you open the program. To log off, select **File»Log off** or press <Ctrl-D>. No dialog box appears in response to either of these actions; *LookoutDirect* just logs you off. You can also log off though the login dialog box.

To log on after Lookout*Direct* has automatically logged you out, after someone else has logged off, or just to log in and replace the current user, select **File»Log on** or press <Ctrl-L> on the keyboard.

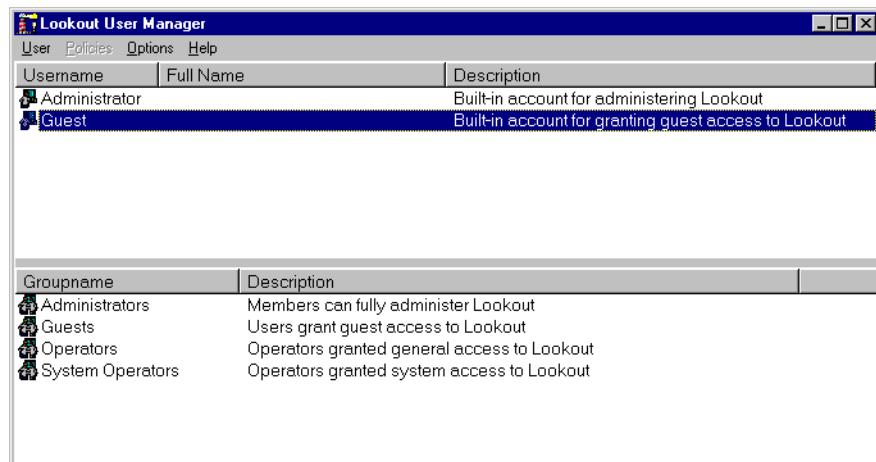
User Manager

You create individual accounts for operators and developers with the User Manager. Anyone whose account is a member of the Administrators group can create, revise, or delete system user accounts by selecting **Options»User Manager**. The Lookout*Direct* User Manager dialog box appears, as shown in the following illustration.



Note For your user accounts to work consistently across your network, you must use the same Lookout.sec file for all your installed copies of Lookout*Direct*. After you have created your Lookout.sec file, make a copy of it from your WINDOWS\SYSTEM directory. Place a copy of the file in the WINDOWS\SYSTEM directory of each of your Lookout*Direct* computers.

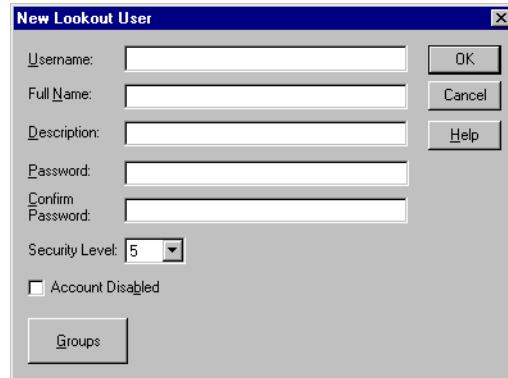
You should carefully consider users and the security level you assign each one. Assign level 10 access only to those people responsible for system security. Users with security levels of 8 and higher can close process files and exit Lookout*Direct*. Users with security level 9 or higher can edit process files in development versions of Lookout*Direct*.



From this dialog box, you can create and edit the properties of groups, create or edit the properties of user accounts, assign users to one or more groups, and otherwise manage security in Lookout*Direct*.

Creating User Accounts

To create a user account, select **User»New User**. The following dialog box appears.



Enter the new user's domain name in the **Username** field.

Enter the user's **Full Name**.

You can use the **Description** field for job titles or other relevant information.

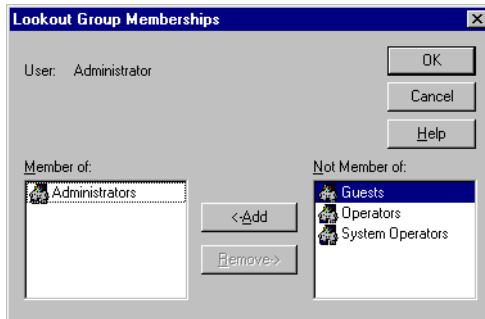
Enter the user's password in the **Password** field.

Enter the password a second time in the **Confirm Password** field to make sure there was no typing error in the first entry.

Set the new user's **Security Level**. Lookout*Direct* security levels range from 0 to 10, with 10 being the highest possible security authorization. Assign level 10 access only to those people responsible for system security. Users with security levels of 8 and higher can close process files and exit Lookout*Direct*. Users with security level 9 or higher can edit process files in development versions of Lookout*Direct*.

Select the **Account Disabled** checkbox if you want to disable a user account without removing the user from the system.

Click on the **Groups** button to add this user to various local security groups. The following dialog box appears.



The default groups in Lookout*Direct* are Administrators, Guests, Operators, System Operators, and Everyone. Any groups you have created are also shown.

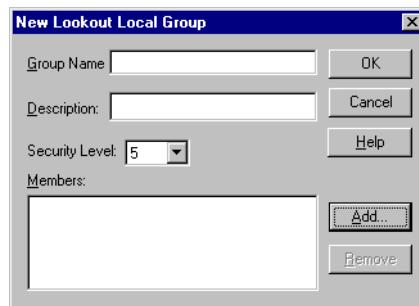
To enter a user in a group, highlight the group in the **Not Member of** field and click on the **Add** button. To remove a user from membership in a group, highlight a group in the **Member of** field and click on the **Remove** button.



Note When you add an individual user who has a security level different than that of the group, that user will have the higher of the security levels.

Creating Groups

To create a group, select **User»New Local Group**. The following dialog box appears.



Group Name assigns a name to your new group.

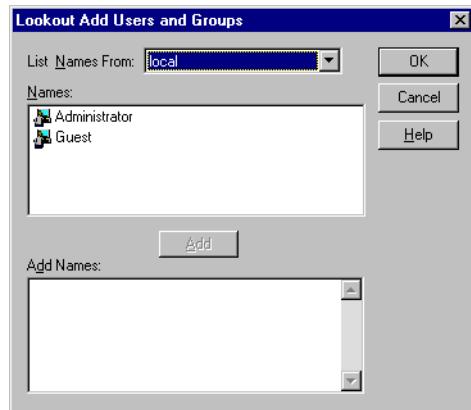
Enter a description of the group in the **Description** field.

Security Level assigns the security level for members of this group.



Note When you add an individual user whose individual account has a security level different than that of the group, that user will have the higher of the two security levels.

To add **Members**, click on the **Add** button. The following dialog box appears.

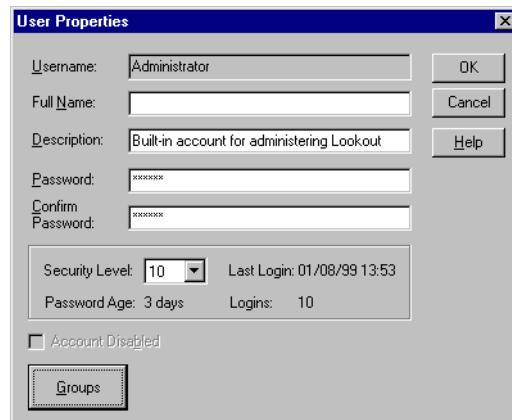


The **List Names From** listbox selects the domain to list user names from. In this version of Lookout*Direct*, you are restricted to your local domain.

Highlight the names you want to add in the **Names** field, and click on the **Add** button to add those users to your group.

Modifying Users and Groups

The dialog boxes for editing users and groups are essentially the same as those for creating users and groups. Open the User Manager, right-click on the user or group you want to edit, and select **Properties**. The following dialog box appears.



The **User Properties** dialog box displays information about user activity.

Special Users and Groups

LookoutDirect comes with several users accounts and groups built-in. The user accounts include Administrator, Guest, and (nobody). The built-in groups include Administrators, Everyone, Guests, Operators, and System Operators. You cannot delete any of these accounts, though you can edit the properties of some of them.

The Administrator account overrides all other security settings and has access to everything in *LookoutDirect*. This override extends to all accounts added to the Administrators group.

You cannot delete the Administrator account or change its security level. You can set the password and enter the name and a description of the Administrator. You can also add or remove user accounts.

The (nobody) account cannot be edited or deleted. This account is what *LookoutDirect* defaults to when no authorized user is logged on. It always has a security level of 0.

The Everyone group is built into *LookoutDirect*. You cannot edit or delete this group in the User Manager. When you first create a process in *LookoutDirect*, it is configured with this group allowed full read and write permissions. Because objects inherit their permission status from the process or folder in which they are created, all the objects you create have this same status until you change them manually, or change the permission status of the process or folder you create them in.

You can edit all the properties of the Guest user account and of the Guests, Operators, and System Operators groups.

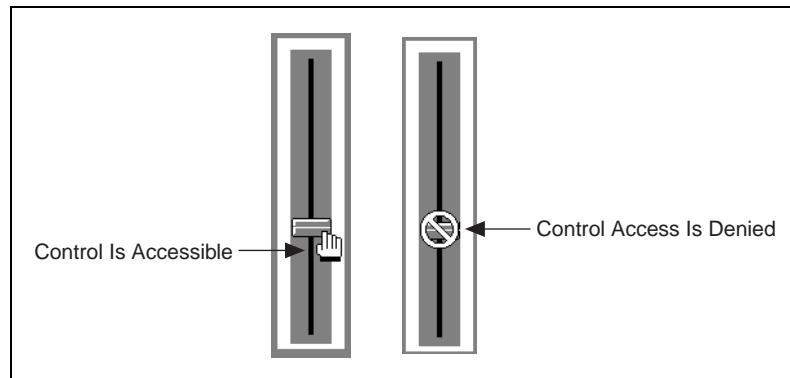
Control Security

Several object classes in Lookout*Direct* support control security, including such control objects as the Pot, Switch, Pushbutton, RadioButtons, and TextEntry, along with a few driver objects. Each class provides some type of control—Pots control numeric output signals, and Switches and Pushbuttons control logical output signals. Each class accepts the **control security level** parameter, which determines whether an operator can control the object. Refer to the online PDF *LookoutDirect Object Reference Manual* for additional information about control object properties.

With control security, Lookout*Direct* compares the security level control of an object to the security level of the currently logged-on account (the operator) to determine if an operator can control (write to) a particular object.

With network security, Lookout*Direct* checks the user account permissions configured for an object or process to determine if an operator can control (write to) a particular object. The user can adjust a control, but the process does not accept the input and the control will return to its original value.

Under control security, if the account security level is equal to or higher than that of the object, the mouse cursor changes into a hand when positioned over the object and the operator can adjust and control the object.



If the account security level is lower than that of the object, the cursor changes into the international symbol for forbidden, and the operator cannot control the object.

You can implement this feature on an object-by-object basis, either through the individual security level set in the object properties dialog box, or by assigning permissions. System integrators can secure high priority Switches,

Pots, and Pushbuttons from operators while still allowing operators to adjust lower-level security objects. Refer to the *Configuring Security for Processes and Objects* section for more information on assigning permissions.

Lookout*Direct* globally applies the Control Panel object security setting to all individual objects on that panel and assigns the higher security level (either the control panel or the individual object) when determining whether an operator can access an object. Refer to the discussion of the *Panel* object in the online PDF Lookout*Direct Object Reference Manual* or the online help for additional information.

Viewing Security

Lookout*Direct* provides viewing security for control panels, controllable objects, and system settings. With these security options, you can restrict access to control panels, objects, and Windows system resources.

Control Panels

A Control Panel object defines viewing security for the entire control panel. For example, if you set Viewing security to level 6 on a particular panel, operators with level 5 or lower cannot view that control panel and might not even know that panel exists. If a level 6 (or higher) operator logs on, the control panel instantly becomes available for display. This feature is useful for hiding panels that are rarely used or that contain sensitive information.

Controllable Objects

Controllable objects such as Pots, Switches, Pushbuttons, and so on have a writable data member called `visible`. When `visible` is true, you can see the object on a control panel. When `visible` is false, you cannot see or adjust the object. To ensure that the object is always visible when it is first created, `visible` defaults to true.

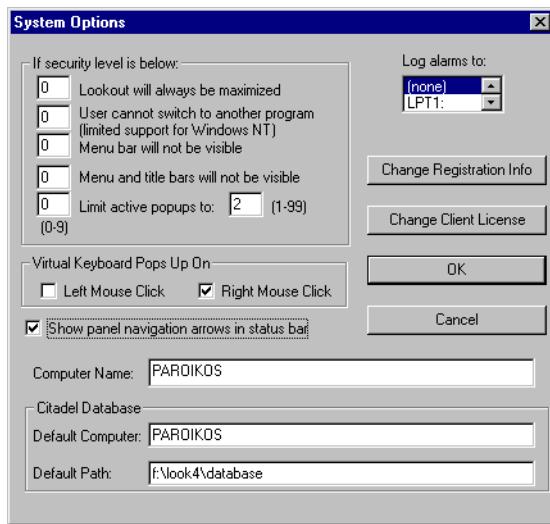
You can connect the `visible` data member of a controllable object (for example, a Pot object) to a controller mode indicator. When the controller is in computer control mode, the `visible` data member of the Pot might be true, allowing the operator to see the Pot and adjust the setpoint. But when the controller is not in computer control mode, the `visible` data member might be false, hiding the Pot from the operator and prohibiting operator control.

You can also use the `username` or `seclevel` data members of the `$System` object to control the visibility of a control object, depending on the name or security of the person logged on to Lookout*Direct* at any given time.

You can also configure network security permissions on these data members.

System Security Settings

With the **Options»System** menu command, you can define system options in the **System Options** dialog box to keep *LookoutDirect* maximized, the menu bar invisible, title bars invisible, and pop-ups to a minimum.



LookoutDirect will always be maximized—When you enter a security level, *LookoutDirect* prohibits users below that security level from closing *LookoutDirect*.

Users cannot switch to another program—This prevents an operator from using <Alt-Tab> to switch from *LookoutDirect* to some other program running on the computer. For this feature to work properly under Windows NT, you must install the *LookoutDirect* NT keyboard driver when you install *LookoutDirect*.

Menu bar (and title bars) will not be visible—When you enter a security level, users below that security level cannot view the menu bar or the title bar and, therefore, cannot change to a different Windows application. This feature is not completely supported under Windows NT 4.0. With Windows NT 4.0, you can still use <Ctrl-Esc> or the Windows key to activate the Windows **Start** menu or <Ctrl-Alt-Delete> to bring up the Task Manager.

Limit active popups to—This option requires two values: a security level and the number of pop-ups. Users below that security level can view up to the

specified number of pop-ups at one time. This feature keeps new users from becoming lost.

Network Security

Lookout*Direct* development security is modeled after Windows NT security but is supported for Lookout*Direct* processes running on Windows 98/95 as well. Users with the ability to access processes or elements within a process are organized into groups. You can limit group and user access to processes, folders, and objects.

Configuring Security for Processes and Objects

You can control access privileges by user or group, applying restrictions to processes, folders within a process, or individual objects. You can configure security in the tree views contained in the Lookout*Direct* Object Explorer, the Lookout*Direct* Connection Browser, the **Edit Connections** dialog box, or the **Insert Expression** dialog box.

You cannot configure security for a network node, for your local computer, or for any Lookout*Direct* global objects such as \$Keyboard or \$System. You must select a process, a folder within a process, or an object within a process or process folder to configure security.

Right-click on the process or object you want to configure security for and select **Configure Security**. The **Security Properties** dialog box appears.



From this box, you can either set **Permissions**, or do **Advanced** security configuration by clicking on the appropriate button.

Permissions

With permissions, you can set individual access privileges for a given process, a folder holding a collection of objects, or individual objects.

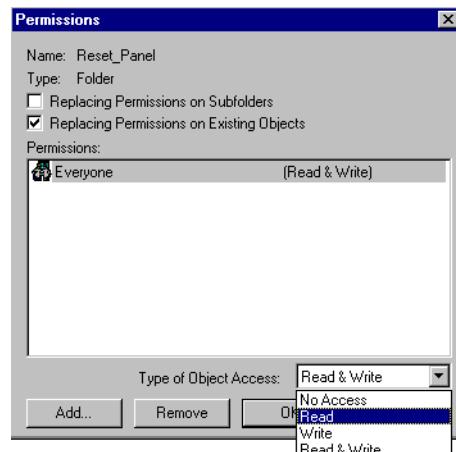
Configure security permissions using either the Lookout*Direct* Object Explorer or the Lookout*Direct* Connection Browser.

Lookout*Direct* objects inherit the permission status of the process or folder in which they are created. When you first create a process in Lookout*Direct*, it has full read and write permission granted to the Everyone group, by default. Any folder or object you create in the process has the same permission.

If you change the permission of the process or of one of the folders, any objects you create after the change have the permission status of the parent process or folder. Changing the permissions of a process or folder does not always change the permissions of an object or folder that already exists in that process, depending on how you set the **Permissions** dialog box options.

If a process has one set of permissions, and a folder under that process has a different set, the objects created under the folder will inherit the permission status of the folder only.

Select **Permission** from the **Security Properties** dialog box. The **Permissions** security properties dialog box appears.



The dialog box in the illustration above shows everyone with access to Lookout*Direct* having permission both to read and write all the controls in the Reset_Panel folder of the Server_1 process.

Your options are to substitute individual users or groups for the Everyone group, and give each user or group the appropriate permission. You can refuse access, permit reading or writing only, or allow both reading and writing.



Note Remember that permissions in LookoutDirect are cumulative. For your individual user and group permissions to have any effect, you must delete the Everyone group after you set your other permissions. Refer to the *Special Users and Groups* section for more information on the Everyone group.

Select the user or group you want to assign permissions for. Select the appropriate security level in the **Type of Object Access** list in the lower right section of the dialog box. When you are done, select **OK**.

By selecting or disabling **Replacing Permissions on Subfolders** and **Replacing Permissions on Existing Objects**, you can restrict that permission to the process, folder, or object you selected, or extend the permission application in different ways and to different degrees.

In the simplest case, if you have selected an individual object, you can only change the permissions on that object.

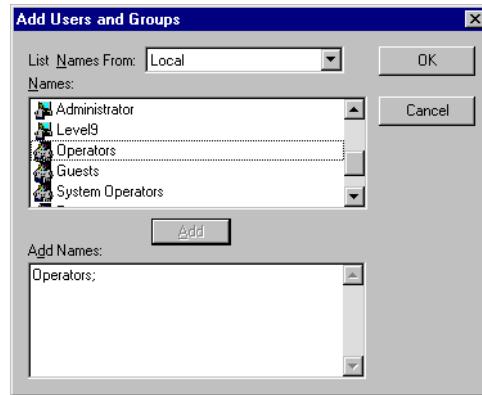
If you selected a process or folder, the options function as shown in the following table.

Table 6-1. Options for Propagating Changes in Security through a Process

Options Selected	Result
neither	Only the selected process, folder, or object has its security configuration changed.
Replacing permissions on subfolders	Changes permissions on the selected process or folder, all the folders under it, and any subfolders under them. This option is disabled when an individual object is selected.
Replacing permissions on existing objects	Changes permissions on all the individual objects contained immediately under the selected process or folder, but does not change permissions on any folder or subfolder, or any object in them. This option is disabled when an individual object is selected.
both	Changes permissions on all the individual objects contained immediately under the selected process or folder, as well as on any folder or subfolder, and all the objects in them.

To remove a user or group entirely from the permissions list, select the user or group and click on the **Remove** button.

To add a user or group, click on the **Add** button. The following dialog box appears.



Select a user or group account in the top window and click on the **Add** button. Click on **OK** when you have selected all the users and groups you want to add.

Configure the security permissions for the added groups as described in the beginning of this section.

Advanced Security

You can set a number of advanced network security options in *LookoutDirect*, but only at the process level. These options are not available on the folder or object level. Click on the **Advanced** button in the **Security Properties** dialog box. The **Advanced** security properties dialog box appears.



There are three advanced security options: **Basic authentication**, **IP setting**, and **Proxy user**.

The default *LookoutDirect* setting is to have both **Proxy user** and **Basic authentication** enabled and the other option turned off.

As with other Lookout*Direct* security settings, the effects of multiple selections in this dialog box are cumulative. In other words, if a user had permission to read under **Basic authentication** and to write under **Proxy user**, then if both options are enabled the user would be able both to read and write.

Basic Authentication

When you select this option, Lookout*Direct* checks the account information of a user logging in to that instance of Lookout*Direct*. Security responds to the security level, individual account, and group permissions of that user account.

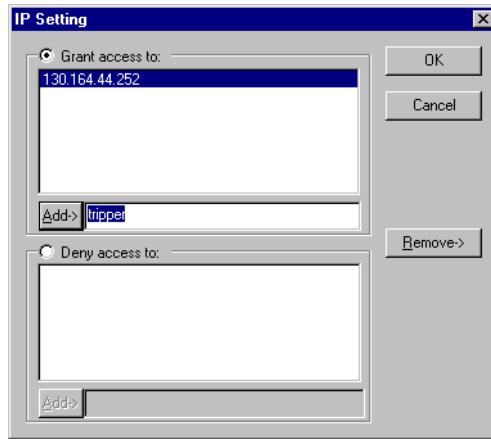
At this time, Lookout*Direct* cannot process the security status of a person logged on to another computer unless you install an identical Lookout.sec file on each computer running Lookout*Direct* on your network. Otherwise, if you have base authentication but not proxy access active on a server process, a person attempting to read from or write to a server from a remote computer cannot access the process unless you have configured the permissions for the Everyone group to have such access.

Activating the proxy access option in addition to the basic authentication option greatly increases your security options.

IP Setting

You can configure Lookout*Direct* to grant or deny access to any computer operating at a specific IP address. Click on the **IP setting** checkbox to enable this feature.

To grant or deny access by IP address, click on the **IP setting** button. The **IP Setting** dialog box appears.



Only one of the IP setting options can be active at one time. Select whether you want to grant access or deny access to a given set of computers.



Note The IP access option functions in a literal way. If you choose to grant access to one or more computers using the IP setting option, those will be the only computers able to access the process or object you have applied the restriction to. If you choose to deny access to one or more computers using the IP setting option, all other computers using *LookoutDirect* will be allowed to access the process or object you have applied the restriction to, subject to the other security settings in place.

Enter the IP address you want to add, and click on the **Add** button. You can enter either the IP number itself, or the simple name of the computer. Whether you use the IP number or the simple computer name, the IP address for the computer appears in the list after you accept the entry.

Proxy Access

You can, if you choose, designate a specific local security account whose security level applies for any user accessing processes in your local instance of *LookoutDirect* from another computer (or another instance of *LookoutDirect* running on the local computer).

In other words, no matter who is logged on to another instance of *LookoutDirect*, the proxy account determines external access rights to a process or object operating under this option. The security level of the operator logged into the external instance of *LookoutDirect* is ignored.

The Guest user is built into *LookoutDirect*, specifically provided for this purpose, as well as for providing a visitor with a user account. You can edit the Guest user account properties in the User Manager.

To enable the proxy option, select the **Proxy** checkbox. Enter the **Username** of the account you want to serve as the proxy security account.

You must enter the valid **Password** for the user account for the proxy option to function.

When a user attempts to access your Lookout*Direct* process or objects from another domain, the user's logon is recorded in the Lookout*Direct* events database. The user will be restricted to the security levels you have configured locally. You must configure any further access restrictions in the client process.

Keeping Security Precedence Simple

The Lookout*Direct* security system is flexible and designed for compatibility both with earlier versions of Lookout*Direct* and with planned future versions. This flexibility carries with it the risk of complexity.

It is not necessary to use every security feature in every Lookout*Direct* process. Keeping your security as simple as possible is the best approach. The following suggestions should help you keep your security simple.

- When converting a Lookout*Direct* process from Lookout*Direct* 3.8.xx or earlier, leave the control security in place when possible.
- In cases when you have both security parameters and network permissions set for an object, it is best to make sure that the security level parameters are consistent with permissions.

If you do find yourself with complex security setting interactions, the following principles should help you sort out how your interactions will work.

- User and group permissions are cumulative.

For example, if user_A is a member of both Operators and System Operators, with the Operators group having read access and the System Operators group having Write access, user_A has both Read and Write access.

- Permissions and control parameter settings are cumulative.

For example, if a control object has a security parameter set to 7, and user_A's user account has a security level of 5, user_A cannot access the control. But if user_A is also a member of the Operators group, and the Operators group has a security level of 7, user_A can access the control.

Additionally, if a control object has a security parameter set to 7, and user_A's user account has a security level of 5, user_A will nevertheless

have read permission if you use network security to grant read permission to user_A's user account.

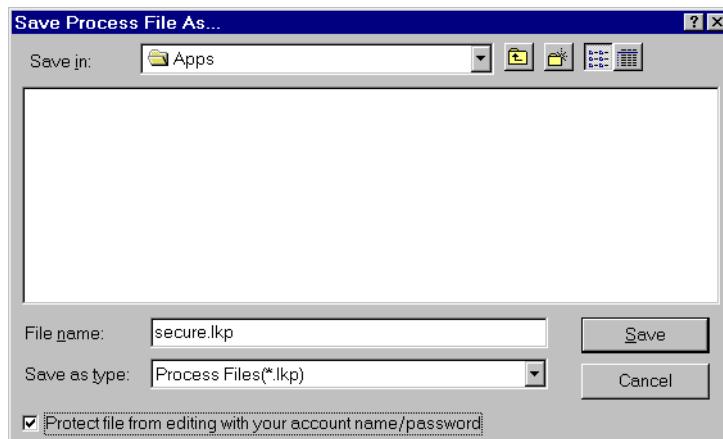
- The no access setting overrides all other permissions granted to a user or the user's group.

For example, if user_A is granted a read access, and user_A is member of Operators, if the Operators group has been assigned no access, user_A has no access either.

Additionally, if user_A is a member of both Operators and System Operators, and Operators group has been assigned no access, user_A has no access either—even if the System Operators group has read access.

Process File Edit Security

You can protect your process files from being edited by any other person without using the security accounts. Log in with an account name and a non-empty password, and select **File»Save As** from the menu. The following dialog box appears.



Check the **Protect file from editing with your account name/password** box at the bottom of the dialog box to save the file with your password as protection. To edit the file again, you have to log in under the same account with the same password.



Note You cannot open an encrypted file with an earlier version of Lookout*Direct*, even if you create an account with the same account name and password in that version.



Caution When you protect a process file, Lookout*Direct* does not save the .LKS file. Because the .LKS file serves as a backup file during application development, you should not use the encrypted-save feature until after you have completed your application and made a backup copy of both the .LKP and .LKS files on a separate archive disk.

Action Verification

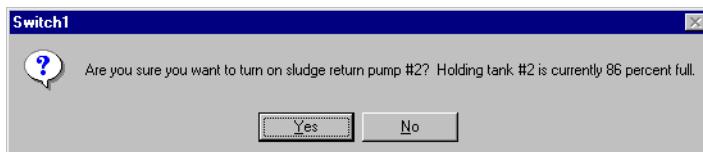
The Switch and Pushbutton object classes support action verification. When you define action verification for an object, Lookout*Direct* displays a message box stating your **Verify** message and prompts you to select either **Yes** or **Cancel**. If you click on **Yes**, Lookout*Direct* completes the previous operator command (for example, flips the switch or presses the pushbutton). If you **Cancel**, Lookout*Direct* ignores the previous operator command.

All action verification parameters accept text expressions, which can contain dynamic data. As an example, consider a switch that controls a pump responsible for filling a storage tank. However, that pump should not fill the tank if the water level is too high. You might enter an expression similar to the following for the switch **Verify On** parameter:

```
"Are you sure you want to turn on sludge return pump #2?  
Holding tank #2 is currently " & DATA_VARIABLE & " percent  
full."
```

Refer to Chapter 1, *Expressions*, for more information about creating expressions using variables.

The warning message appears every time you turn on the switch. Notice the water level is dynamic—it changes to reflect the value of *DATA_VARIABLE* when the switch is flipped.



When you turn off the switch, no warning message appears because the **Verify Off** parameter was not specified. If you want to disable the **Verify On** warning message, delete the entire expression from the data field.

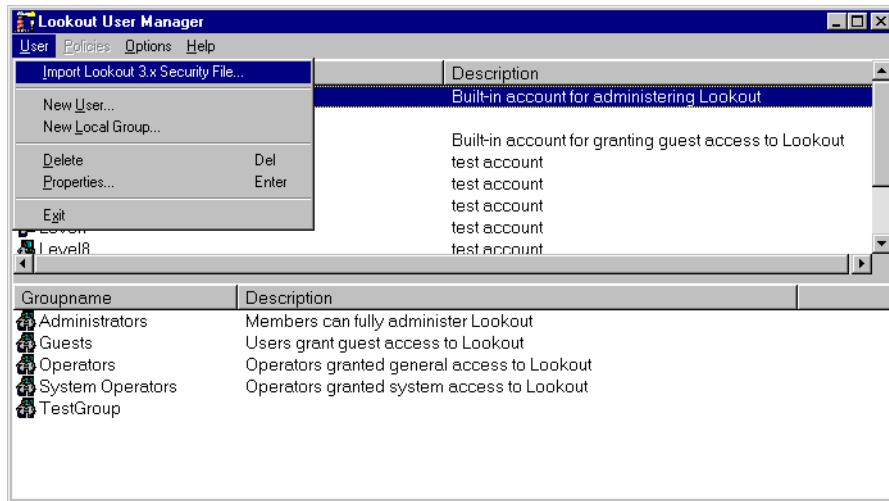


Caution Pushbutton verification works in much the same way. However, when you select **Yes**, the pushbutton creates only a momentary output signal. When action verification is enabled, it is impossible to hold the button down for any length of time.

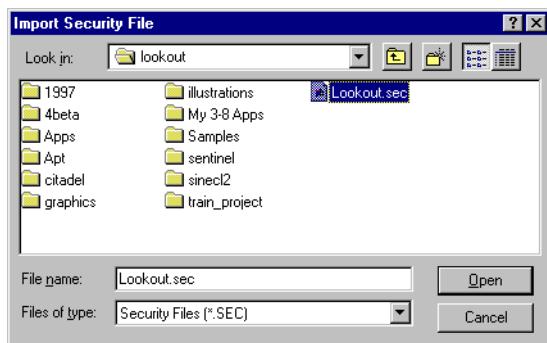
Importing Old Security Files into Lookout*Direct* 4

You can import the user account information from your Lookout*Direct* 3.8 processes into Lookout*Direct* 4 using the Lookout*Direct* User Manager.

1. Select **Options»User Manager** from the Lookout*Direct* menu. You must be in edit mode for the **User Manager** item to appear in the **Options** menu. The **User Manager** dialog box appears.



2. Select **User»Import Lookout*Direct* 3.x Security File** from the **User Manager** dialog box. The following dialog box appears.



3. Navigate to your old Lookout*Direct* 3.8 security file `Lookout.sec`, and select it. Lookout*Direct* 3.8 kept the `Lookout.sec` security file in the Lookout*Direct* directory.

4. Click on **Open**.
5. If you have already created any user accounts in Lookout*Direct* 4 that are the same as accounts you used in Lookout*Direct* 3.8, you will receive a message informing you that a user account with that name already exists. You may replace your recently created account, or choose not to use the old account information.
6. Exit the User Manager.



Note Unlike Lookout*Direct* 3.8, Lookout*Direct* 4 maintains the Lookout.sec security file in the Windows System directory. The Lookout*Direct* 4 User Manager creates a unique identification number for each user account. For Basic Authentication to work properly, you must use the same Lookout.sec file for each copy of Lookout*Direct* running on your network. Copy your Lookout.sec file to the Windows System directory in every computer on which you intend to run Lookout*Direct*.

Logging Data and Events

This chapter describes three Lookout*Direct* methods for logging real-time system data to disk—Spreadsheet Logger, Citadel Historical Database, and Event Logger—and report generation.

The Spreadsheet Logger creates standard ASCII text files in comma separated value (.CSV) file format. You can open and edit these files with common spreadsheet and database programs. The Citadel Historical Database creates a historical database that Lookout*Direct* HyperTrend objects access in real time. You can retrieve this data using Structured Query Language (SQL). The Event Logger creates a chronological audit trail of who did what and when.

You might implement all three methods on a single system. They are *not* mutually exclusive.

Refer to Chapter 9, *Alarms and Events*, for more information on the Alarm Logger, which logs alarms to disk.



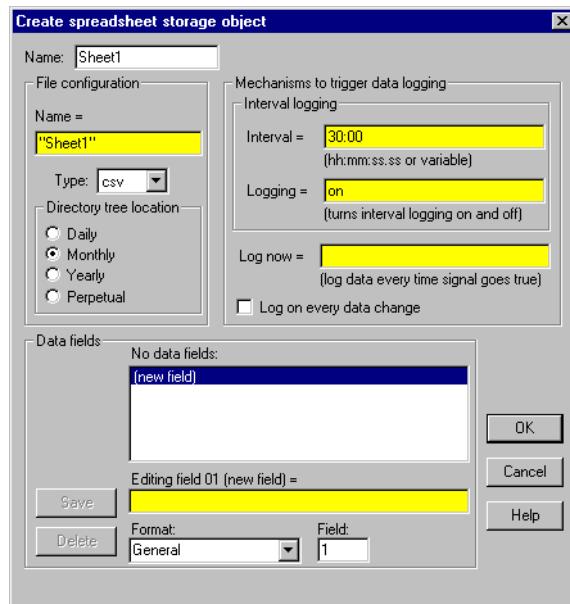
Note When your operating system switches in or out of Daylight Savings time, Lookout*Direct* corrects for the change relative to universal time so that there is no data discontinuity or loss in the Citadel database.

Spreadsheet Logger

When you want to create permanent ASCII files that you can later open with software packages such as Excel, Lotus, and Foxpro, use a Spreadsheet object to store real-time system data to disk.

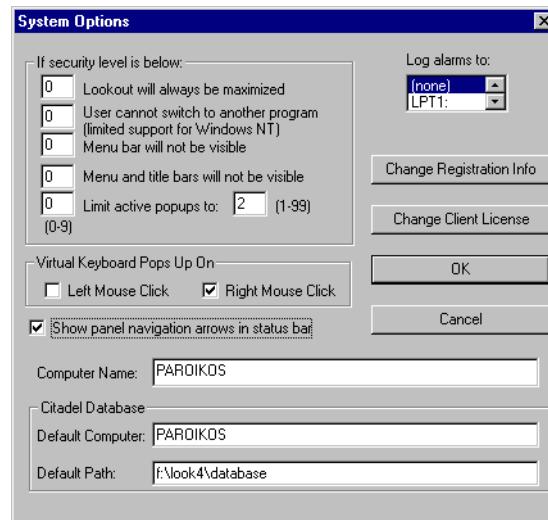
You can generate more than one spreadsheet for a single process if you use multiple Spreadsheet objects in one process file (a .LKP file). You can have different spreadsheet files in different subdirectories and each might have different data stored using different logging criteria. For more information on the Spreadsheet object, refer to the *Spreadsheet Logger* description in the online PDF *Object Reference Manual* or online help.

The following illustration shows the create object dialog box that you use to configure a spreadsheet object.



Data Location

LookoutDirect stores spreadsheet files in subdirectories specified by the **Citadel Database** section of the **System Options** dialog box illustrated below.



CSV Files

All LookoutDirect spreadsheet files use the comma-separated value (.csv) format. A .csv file contains text separated by commas. Each line of the file represents a row of spreadsheet data. The first row lists signal names and describes the data stored in each column of the file. The first column of the spreadsheet file specifies the date and time data in each row was logged. The following illustration shows a Lookout .csv file opened by Excel.

	A	B	C	D	E
1	Time	Presssure	TankLevel	ReliefValve	Temperature
2	9/29/97 15:44	46	11	0	78
3	9/29/97 15:45	37.64	31.37	0	76
4	9/29/97 15:45	30.63	50.92	0	77
5	9/29/97 15:45	23.99	58.3	0	78
6	9/29/97 15:45	19.19	64.94	1	79
7	9/29/97 15:45	15.5	55.35	0	78

You can view and edit spreadsheet files using a standard word processing program, but spreadsheet programs such as Microsoft Excel provide a more intuitive interface for viewing or editing data and printing custom reports.

File and Disk Errors

If LookoutDirect cannot find the data directory or spreadsheet files or cannot log data to disk, it generates a priority eight alarm with a descriptive error message. For example, if your default data location was on c:\lookout\database, and LookoutDirect was to encounter a full disk while writing to a file named data.csv, the alarm text would read

(Disk is full) "c:\lookout\database\data.csv".

When you correct the problem that caused the alarm, the alarm automatically resets.

Concurrent File Access

When logging data, *LookoutDirect* opens the file long enough to add a new row of data and then closes the file. *LookoutDirect* can log approximately ten new rows of data per second. (The time stamps associated with each row are rounded to the nearest second.) Because *LookoutDirect* runs under the Windows multitasking environment, you can open the spreadsheet or database while *LookoutDirect* is running. If you have the data file open in a spreadsheet program such as Excel, *LookoutDirect* cannot add new rows of data to the file. If *LookoutDirect* does not have write access to the file, it generates an alarm explaining the problem.

```
Error writing spreadsheet file: Permission denied  
[c:\lookout\database\data.csv]
```

If *LookoutDirect* cannot log data to the spreadsheet file, it creates a temporary buffer in memory to which it logs all the spreadsheet data. When you close the spreadsheet file, *LookoutDirect* updates it and resets the alarm.

Information Overload

If you create a *Spreadsheet* object that logs fifty data fields (object data members and/or expressions) at five-minute intervals, you generate 14,400 data points per day, or 432,000 data points per month. It can be difficult to determine which data are critical information.



Note Carefully consider the importance of each data point and the time interval between data points to avoid information overload for both you and your spreadsheet program.

If you have logged more than 16,384 rows of data, the current version of Excel cannot load the entire spreadsheet file. Spreadsheets created with one-minute intervals generate 44,641 rows of data if the directory tree location is set to **Monthly**. Spreadsheets created with two-minute intervals generate 22,321 rows of data if the directory tree location is set to **Monthly**. To decrease the number of data rows, try changing the directory tree location to **Daily**, which generates only 1,440 rows of data per day at one-minute logging intervals. You can also use a text editor to divide the larger file into smaller files with fewer data rows.

Besides limiting rows to 16,384, Excel limits spreadsheets to 256 columns of data. One column is reserved for the date and time.

Citadel Historical Database

When you need to view historical information using Lookout*Direct* HyperTrend objects or access historical data using Structured Query Language (SQL), use the Citadel Historical Database to store real-time numeric and logical data in a compressed format.

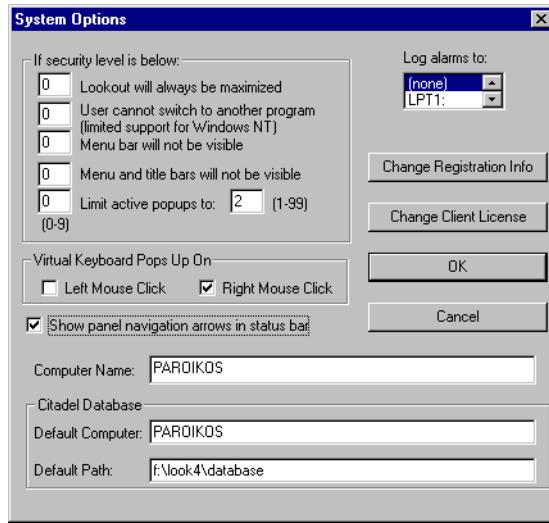
Lookout*Direct* logs data only when the value of a point changes, *not* at a timed interval. The logged data accurately reflects the actual behavior of the point being logged because the same value of a point is not recorded over and over. With Citadel, disk usage is exactly proportional to the amount of information recorded.

After the data is stored to disk, you can view it with a HyperTrend object in Lookout*Direct* or with SQL. HyperTrend objects serve as windows into your database. SQL, an industry-standard language supported by most database packages, enables other applications to directly retrieve data from Citadel.

Because Citadel uses Universal Coordinated Time (UCT) to time stamp the data, you do not need to compensate for data logged in different time zones. The HyperTrend object automatically converts from universal to local time before displaying a data history.

Default Data Location

The **Citadel Database** fields in the **System Options** dialog box specify the root directory under which all data logging is stored, including the database. Ordinarily, Citadel creates a special `database` subdirectory in the Lookout*Direct* directory. This subdirectory holds your historical database files.

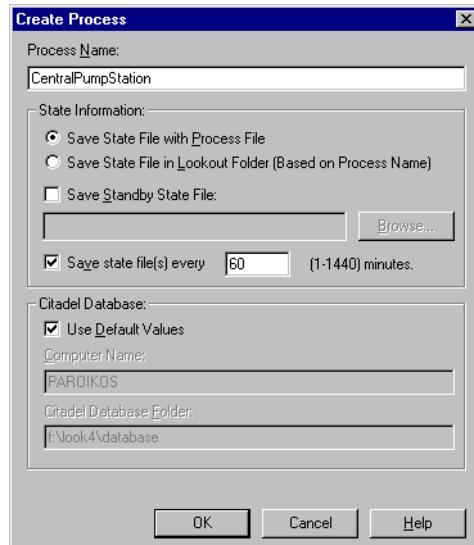


If you move or rename the subdirectory or files, you will be unable to access your data. If you change the default directory, the change does not take effect until you stop and restart *LookoutDirect*. This also interrupts data continuity, and you have to do separate SQL queries to both databases.

Unlike spreadsheet files, only one historical database per *LookoutDirect* process exists. If you want to develop multiple processes where each process file has its own unique historical database on a single computer, specify a different location for each process database, as explained in the *Process Data Location* section of this chapter.

Process Data Location

When you create a process in *LookoutDirect*, you have an opportunity to set the default data location for that process in the **Create Process** dialog box.



When this dialog box first appears, the **Use default values** checkbox is selected. This option directs *LookoutDirect* to log data from this process to the default location set in the **System Options** dialog box for the copy of *LookoutDirect* running the process. In this way, you can create a process that logs to the default data location of any copy of *LookoutDirect* running on any computer you move the process to.

You can also direct data to a location on a remote computer, as long as that computer is running the *LookoutDirect* Citadel service. Deselect the **Use default values** option and enter the computer and path you want data logged to.

Although you can direct data from different processes to different database directories, proliferating a variety of databases with large numbers of processes running in one instance of *LookoutDirect* can degrade performance, depending on the number of data points being logged.

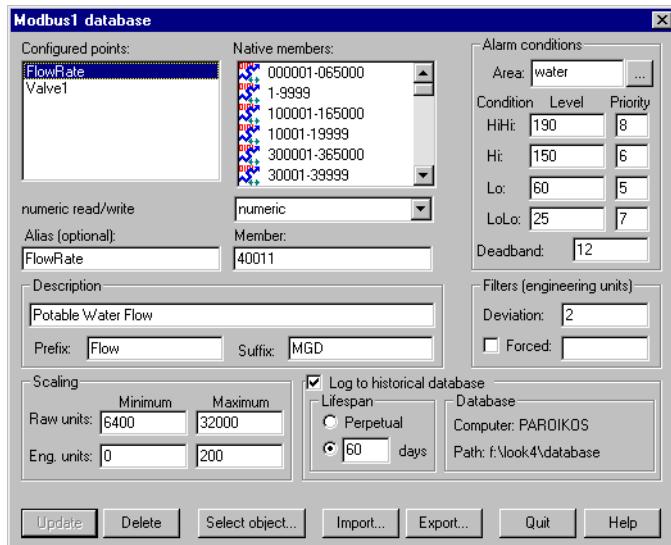
Creating a Historical Database

Citadel stores historical information using traces. Trace refers to the line of continuity for a specific data member name or point. A trace connects all the historical values for a given point, which displays as a continuous line in a HyperTrend object. If *LookoutDirect* is unexpectedly interrupted or a data member is temporarily modified to *not* log to disk, a trace can be broken. If the trace is broken, the HyperTrend plots the trace as a continuous line with void sections to represent gaps in the data. You cannot remove individual

traces from the database, but you can add new traces to the database by configuring a new point.

You can log any numeric or logical data member of any object to the historical database. Use the following procedure to add a trace to the Citadel database:

1. Select **Object>Edit Database** and an object that contains data you want to log. A dialog box similar to the following appears.



2. Identify the value you want to log in the **Member** data field. If the value was previously configured, select it from the **Configured points** list.
3. Check the **Log to historical database** checkbox and choose an appropriate **Lifespan**. See *Logging Criteria* for an explanation of time span settings.
4. Enter a value in the **Deviation** field. See *Logging Criteria* for an explanation of **Deviation** settings.
5. Click on the **Save** or **Update** button. (If you are modifying an existing configured point, the button automatically changes to **Update**.)
6. Configure all the object data members that you want logged to the Citadel database.
7. Click on the **Select object** button and select the next object containing data you want to log.
8. Repeat steps 2 through 6 until you have created the traces that you need.
9. Click on **Quit**.

Logging Criteria

To create a new trace in the Citadel historical database, Lookout*Direct* needs the following information: data member name (**Alias**), **Deviation**, and **Lifespan**.

- Data member name (**Alias**)—Lookout*Direct* must name each trace in the historical database for archiving and retrieval purposes. If **Alias** is not specified, Lookout*Direct* uses the native data member name as displayed in the **Member** field. If an alias is assigned to the native data member, Lookout*Direct* uses the **Alias** name instead. The name used for the trace is the same name saved to the **Configured points** list box.
 - **Deviation**—If an analog value surpasses or equals the **Deviation** setting, Lookout*Direct* saves a new value to disk in the database. If you implement **Scaling** parameters, Lookout*Direct* compares the **Deviation** setting to changes in the **Eng. units** values to determine when to log a new point. If you leave the **Scaling** parameters blank, Lookout*Direct* compares the **Deviation** setting to the raw (unscaled) signal. The **Deviation** parameter is not available on logical signals because Lookout*Direct* logs all state transitions of logical values to disk.
 - **Lifespan**—Lookout*Direct* needs to know how long you want to maintain this trace of data in the historical database. Lookout*Direct* maintains the data for *at least* the time span specified in the **Log to historical database** parameters.
- Lookout*Direct* does not discard old data immediately. Disk space containing old data is reused when new data is logged.

Alarm Information Overload

Because Lookout*Direct* creates and maintains the historical database in 1 MB chunks, you must have at least 1 MB of available disk space to begin logging data. When the first 1 MB file fills up with data, Lookout*Direct* creates a second 1 MB file in the same directory. When the second fills up, Lookout*Direct* continues to create 1 MB files until all traces can be maintained on disk for the time spans specified in the **Log to historical database lifespan** data field. Lookout*Direct* monitors the status of your hard drive and generates an alarm when your disk falls below 500 KB of available space.

Use discretion when determining **Deviation** settings on each trace of data. If **Deviation** is not specified, *any* change of an analog value results in a new point being logged to disk, and you might generate too much data. If the **Deviation** setting is too large, very little information might be saved to disk.

Event Logger

When you want to create a chronological schedule of events or an audit trail, log Lookout*Direct* events. You can log all events: operator commands, Event objects, and all global events (running Lookout*Direct*, closing Lookout*Direct*, opening a process file, closing a process file, entering edit mode, exiting edit mode, and logging on and off).

Lookout*Direct* system events, such as logging on and logging off, or entering and exiting edit mode, are automatically logged as events by Lookout*Direct*. Other events, such as adjusting the value of a Pot object, are only logged if you select the **Log events** option when you create or modify the object.

When Lookout*Direct* logs an event, it also logs the account name (the operator), date and time of the event, name of the object adjusted, and the previous and subsequent settings of the object. Because all of this information is logged, you create an exhaustive audit trail for that specific workstation.

Use the Event object class to define and log your own event messages based on a user-defined trigger. For additional information, refer to the *Event* object class in the online help or the PDF *Lookout*Direct* Object Reference Manual*.

Event Data Location

Lookout*Direct* stores event files in the same Citadel database that holds Lookout*Direct* alarms and Lookout*Direct* data for any Lookout*Direct* process.

To view events in real time, select the **Show Events** or the **Show Alarms & Events** option in the **Alarm Filters** dialog box. Print events the same way you print alarms, using the appropriate alarm filter settings.

Event Information Overload

The event logger can store information on an individual object instance for Switches, Pots, and Pushbuttons. Every time you flip a Switch, adjust a Pot, or depress a Pushbutton, Lookout*Direct* logs the event to disk. If you enable the Event Logger for all instances of these object classes, you might collect more data than you need.

By selectively enabling the Event Logger on critical Switches, Pots, and Pushbuttons, you cut down on the amount of information logged to the event file. For example, you might place a dozen Switches on a control panel, but only one of those twelve requires data logging. Rather than log events for all 12 Switches, enable the Event Logger only on that single switch.

Report Generation

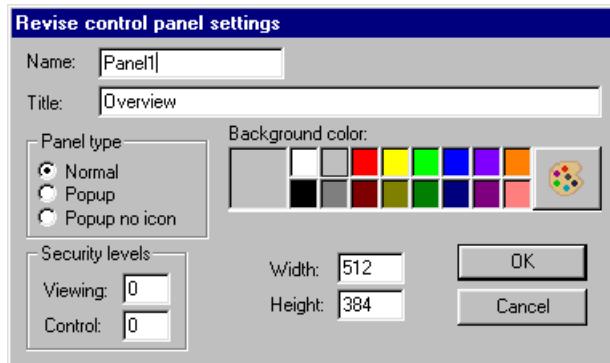
Control Panel Reports

Panel objects have a special data member named `print`. When `print` changes from FALSE to TRUE, Lookout*Direct* copies a picture of the control panel and sends that copy to your printer using the Windows Print Manager. You could use this to record the status of a particular panel at a particular time. For more information, refer to the *Panel* description in the online PDF *Object Reference Manual* or the online help.

To configure your custom report, design a control panel and connect a logical trigger signal to the panel. You can connect any logical expression to the `print` member. The control panel does not have to be visible to be printed. Therefore, an operator can view one screen (panel) while printing another screen. For more information about making connections, refer to the Lookout*Direct* *Getting Started Guide*.

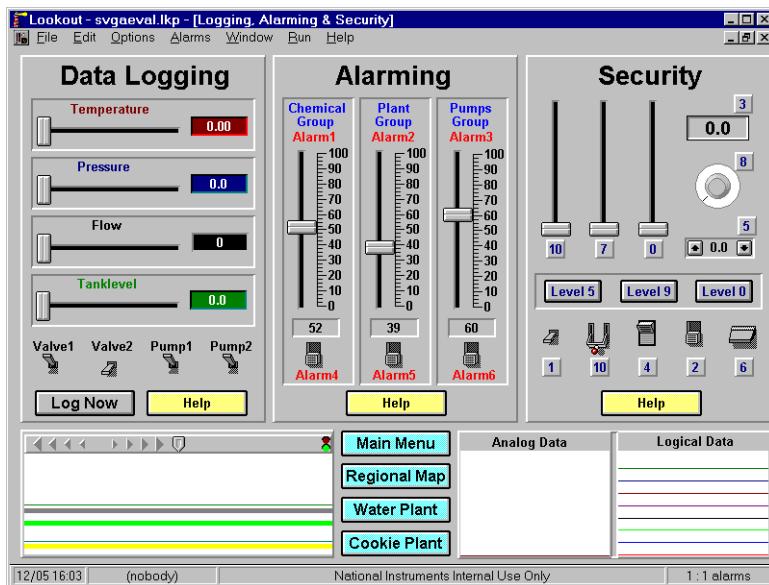
Lookout*Direct* determines the printed panel size based on dimensions entered in the **Revise control panel settings** dialog box. There are several ways to access this dialog box. First, make sure you are in edit mode. Then, do one of the following:

- Right-click on a panel title bar.
- Right-click on a panel in the **Object Explorer** and select **Properties**.
- Select **Object»Modify**, then select a panel from the list.

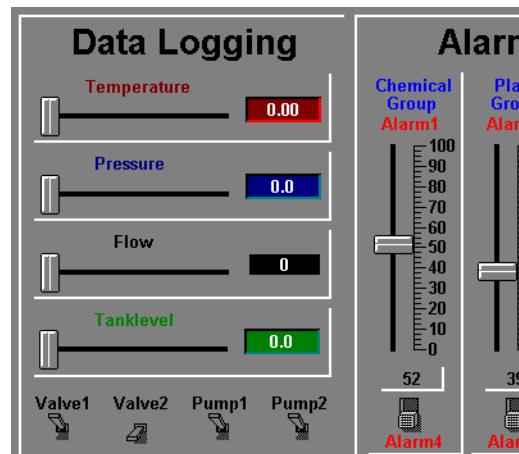


When **Panel type** is set to **Normal**, Lookout*Direct* assumes that the panel is maximized. Even though your control panel might be maximized, Lookout*Direct* prints only the area defined by your Panel object **Width** and

Height settings. For example, you might have a panel that looks like the following when maximized.



The panel shown above has a specified **Height** of 300 pixels and a **Width** of 400 pixels. At a video resolution of 800×600 , the printed panel would look something like the following figure.



Modify the **Height** and **Width** dimensions of your Panel object to match your screen resolution. This problem applies only to **Normal** style panels. **Popups**

and **Popup no icon** style panels print according to the WYSIWYG principle (What You See Is What You Get).

Third-Party Reports

You can generate reports using Lookout*Direct* and a third-party spreadsheet program. Current spreadsheet software provides extensive macro command capabilities you can use to quickly generate reports from Lookout*Direct*. You can write Excel macros that prompt you for the report date, automatically open the correct spreadsheet file(s), extract the data, and print reports with combinations of numeric tables and graphs. You can launch these macros from Lookout*Direct* or automatically invoke them using predefined timers.

8

Structured Query Language

This chapter describes Structured Query Language (SQL), Open Database Connectivity (ODBC), and accessing Citadel data using both SQL and ODBC.

Introduction

The Citadel historical database includes an Open Database Connectivity (ODBC) driver, which enables other applications to directly retrieve data from Citadel using Structured Query Language (SQL) queries.

What is ODBC?

ODBC is a standard developed by Microsoft. It defines the mechanisms for accessing data residing in database management systems (DBMSs). Nearly all Windows-based applications that can retrieve data from a database support ODBC.

What is SQL?

Structured Query Language (SQL) is an industry-standard language used for retrieving, updating, and managing data. In Lookout*Direct*, you can use SQL to build queries to extract data from Citadel. The Citadel ODBC driver also includes many built-in data transforms to simplify statistical analysis of retrieved data.

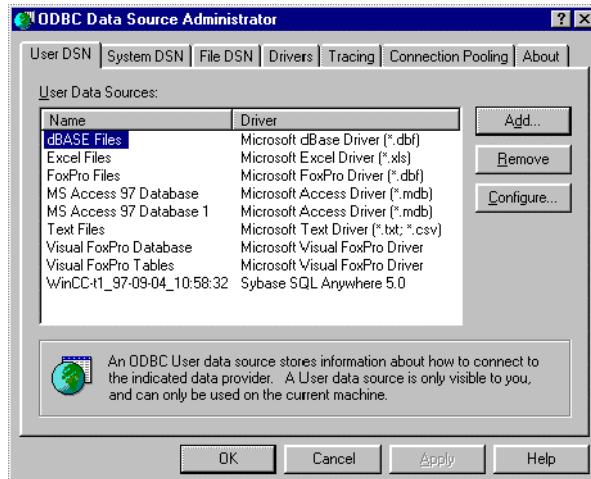
Creating a Citadel ODBC Data Source

Use the following procedure to create a Citadel ODBC data source for use with LookoutDirect.

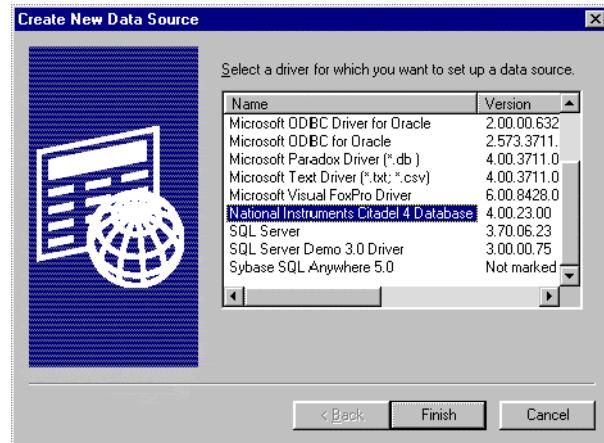
1. Click the Windows **Start** button and select **Settings»Control Panel**.
2. Run the ODBC applet. It might be called **ODBC** or **ODBC Data Sources** or something similar, depending on your operating system version number.



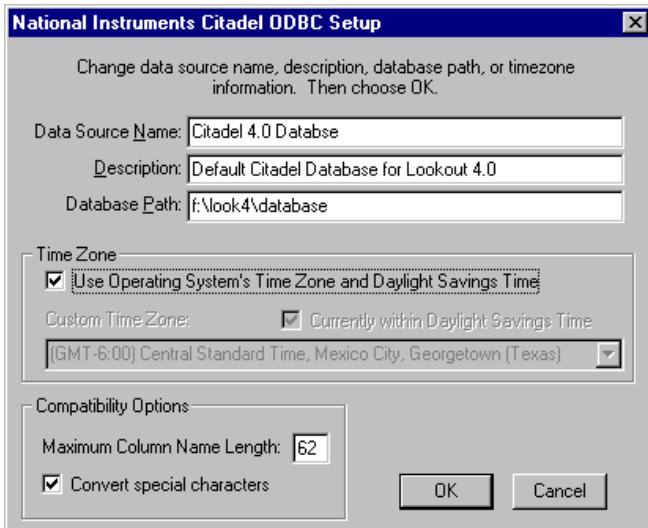
Note Shut down all ODBC applications, such as databases, spreadsheets, word processors, and Microsoft Query, before you run the ODBC applet.



3. Select the **User DSN** tab or the **System DSN** tab, depending on which type of data source you want to create. User DSNs are only visible to the user who created them on the current machine. System DSNs are available to all users on the current machine.
4. Click on the **Add** button. The following dialog box appears.



5. Select **National Instruments Citadel 4 Database**, then click **Finish**.
6. In the **National Instruments Citadel ODBC Setup** dialog box, fill in the **Data Source Name**, **Description**, and **Database Path** fields.
 - a. The **Data Source Name** is the name that ODBC applications use to select your data source.
 - b. **Description** is a free-form text string you can enter to describe the data source.
 - c. **Database Path** should match the location of the Citadel database you intend to access. Use a fully qualified path to a remote computer if you are accessing the Citadel database on a remote computer, such as \\Tripper\c:\lookout\database.



The **Data Source Name** must be different from any other ODBC data source name. The **Description** is arbitrary. The **Database Path** gives the location of the folder where the data for this source is stored. In *LookoutDirect 4*, it is possible to configure multiple Citadel databases. If you do so, you probably want to configure one ODBC data source for each. If you used the **Options»System** dialog box in *LookoutDirect* to configure a default Citadel database on the local computer, you should create a data source for the location specified in the **Default Path** field in the *LookoutDirect System Options* dialog box. If you used **File»Modify Process** in *LookoutDirect* to configure a different database for a process, you probably want to create a separate data source for that process using the location specified in the **Citadel Database Folder** field of the *LookoutDirect Modify Process* dialog box.

7. Click on **OK** in the **Setup** dialog box, then click on **OK** in the **ODBC Data Source Administrator** dialog box.



Note Some applications are not completely ODBC compliant. If you plan to use Microsoft Query, Microsoft Access, or Visual Basic, make sure **Maximum Column Name Length** does not exceed 62 characters. These applications cannot handle longer names. Applications that are completely ODBC compliant can handle names up to 126 characters long. All traces whose names exceed the **Maximum Column Name Length** are excluded from queries.

Because *LookoutDirect* objects may include network path information in their names, and because you can arrange *LookoutDirect* objects in hierarchies inside your processes, you may find it easy to exceed the 62 and 126 character limitations of ODBC. Consideration given to naming and organizing objects can minimize the risk of encountering this difficulty.

If you plan to use Microsoft Access or Visual Basic, select **Convert special characters** to force Lookout*Direct* names into an accepted format by replacing characters within the names with the characters in Table 8-1.

The special characters changed in Lookout*Direct* 4. If you are converting Lookout*Direct* processes from a version earlier than Lookout*Direct* 4, you might have to rewrite any SQL queries you set up in your earlier processes.

Table 8-1. Special Access SQL Characters

Special Character	Converted Character
period (.)	at sign (@)

Accessing Citadel Data

Traces Table

The ODBC driver presents Citadel data to other applications as a traces table. The table contains a field or column for each data member logged to the Citadel database and three fields you can use to specify query criteria and to time stamp retrieved data: **Interval**, **LocalTime**, and **UTCTime**.

Interval specifies the query value sample rate. **Interval** can range from 10 ms to several years. **Interval** defaults to 1 (one day). **WEEK** is a standard seven days, but **MONTH** and **YEAR** account for different month lengths and leap years.

Because Citadel is event-driven, it only logs a value when the value changes. Using **Interval**, you can query Citadel for values evenly spaced over a period of time.

LocalTime and **UTCTime** are time-stamps that indicate when values are logged. Citadel stores the time in **UTCTime** format and derives **LocalTime** from the stored time.

The following where clause query uses **Interval** and **LocalTime** to select data over a specified time at one-minute intervals. Notice that time and date formats are the same as those used in Lookout*Direct*.

```
SELECT * FROM Traces
WHERE LocalTime>"12/1 10:00"
    AND LocalTime<"12/2 13:00"
    AND Interval="1:00"
```

Points Table

The Points table is used to retrieve the actual values logged by Lookout*Direct* for a data member and the times they were logged. As Lookout*Direct* logs values for data members asynchronously, there is no correlation between the timestamps for one data member and another. For this reason, when querying the Points table, you may only query one data member at a time.

The where clause using **LocalTime** and **UTCTime** is supported for the Points table; however, **Interval** is not relevant to the Points table. The data transforms are also not relevant to the Points table and are not supported.

An example of a query using the Points table could be

```
SELECT LocalTime, Pot1 FROM Points
WHERE LocalTime > "12/1 10:00" AND
LocalTime < "12/2 10:00"
```

Data Transforms

Your queries can include special commands that perform data transforms to manipulate and analyze historical data. Table 8-2 lists data transform commands.

Table 8-2. Data Transform Commands

Command	Transformation
Min{Datapoint}	Returns the minimum for <i>Datapoint</i> across the interval.
Max{Datapoint}	Returns the maximum for <i>Datapoint</i> across the interval.
Avg{Datapoint}	Returns the average for <i>Datapoint</i> across the interval.
Stdev{Datapoint}	Returns the standard deviation for <i>Datapoint</i> across the interval.
Starts{Datapoint}	Returns the number of starts (that is, the number of transitions from OFF to ON) for <i>Datapoint</i> across the interval. For numeric points, 0.0 is interpreted as OFF, and all other numbers are treated as ON.
Stops{Datapoint}	Returns the number of stops (that is, the number of transitions from ON to OFF) for <i>Datapoint</i> across the interval.
ETM{Datapoint}	Returns the amount of time <i>Datapoint</i> was in the ON state across the interval.
Qual{Datapoint}	There might be gaps in the historical data traces in Citadel because of machine shutdown, Lookout <i>Direct</i> shutdown, or a similar occurrences. Qual returns the ratio of time for which valid data exists for <i>Datapoint</i> across the interval to the length of the interval itself. If valid data exists for only one-half of the interval, Qual returns 0.5.

Using these data transforms, you can directly calculate and retrieve complex information from the database such as averages and standard deviations, so you do not need to extract raw data and then manipulate it in another application.

For example, you need to know how many times a compressor motor started in December. You also need to know its total runtime for the month. Use the following query to get your answers:

```
SELECT "Starts{PLC.MotorRun}" ,
       "ETM{PLC.MotorRun}"
  FROM Traces
 WHERE LocalTime>="12/1/95"
   AND LocalTime<"1/1/96"
   AND Interval="31"
```

SQL Examples

The following examples are typical query statements; however, your queries might be much more involved, depending on your system requirements.

```
SELECT *
  FROM Traces
```

Retrieves the current value of every data member logged to Citadel. Because your query does not occur at the same moment in time as a PLC poll, signals scanned from PLCs are not included in the retrieved data.

```
SELECT *
  FROM Traces
 WHERE Interval="0:01"
```

Retrieves the value of every data member logged today in one-second increments. Notice that the interval value is enclosed in quotation marks.

```
SELECT LocalTime, "Pot1"
  FROM Traces
 WHERE LocalTime>"8:50"
   AND Interval="0:01"
```

Retrieves and time stamps the value of Pot1 in one-second increments from 8:50 this morning to now. Names are enclosed by quotes.

```
SELECT LocalTime, "AB1.I:3", "Max{AB1.I:3}"
  FROM Traces
 WHERE LocalTime>"10/1/95"
   AND LocalTime<"11/1/95"
   AND Interval="1:00"
```

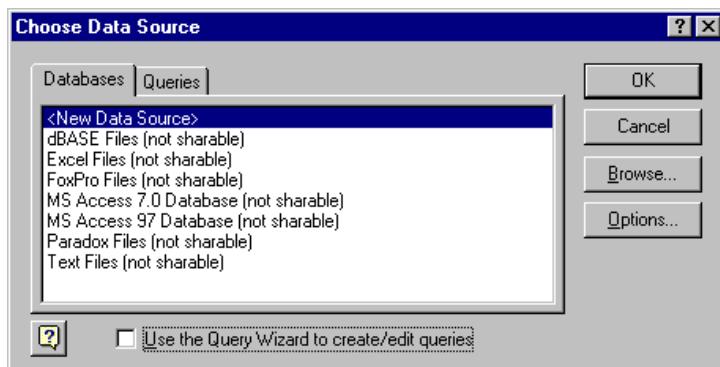
Retrieves and time stamps an Allen-Bradley PLC input in one-minute intervals for the month of October. This query also indicates the highest occurring input value of each minute.

```
SELECT LocalTime, "OVEN1_SP", "PLC.OVEN1_PV",
"Max{PLC.OVEN1_PV}", "Min{PLC.OVEN1_PV}",
"Avg{PLC.OVEN1_PV}"
FROM Traces
WHERE LocalTime>="14:00"
    AND LocalTime <"15:00"
    AND Interval="1:00:00"
```

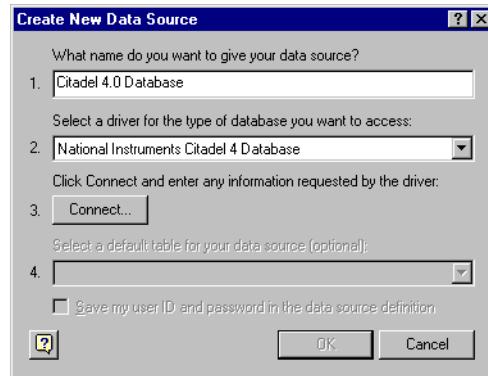
Retrieves an oven temperature at 3:00 p.m. and shows the highest, lowest, and average temperatures between 2 p.m. and 3 p.m.

Accessing Citadel Data with Microsoft Query

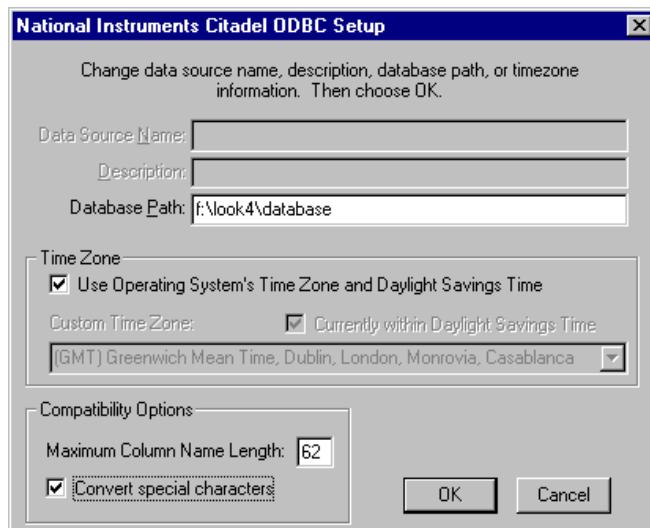
1. Launch Microsoft Query.
2. Choose **File»New**. If the Citadel data source is not visible, you must create it. If it is visible, skip to step 3.
 - a. Select the **<New Data Source>** entry in the **Databases** tab. Make sure the **Use the Query Wizard to create/edit queries** option is *not* selected.



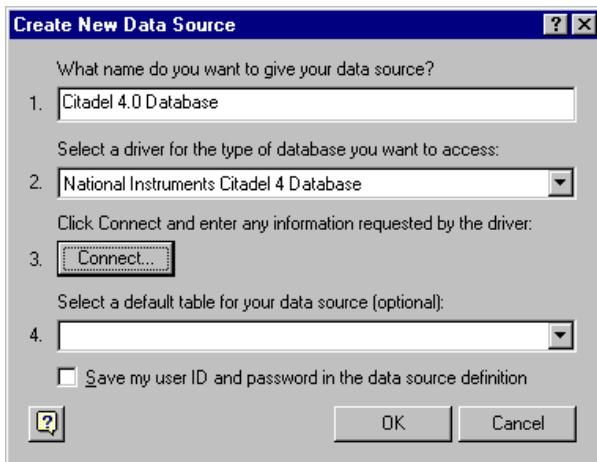
- b. The **Create New Data Source** dialog box appears. Fill in the fields as shown in the following illustration.



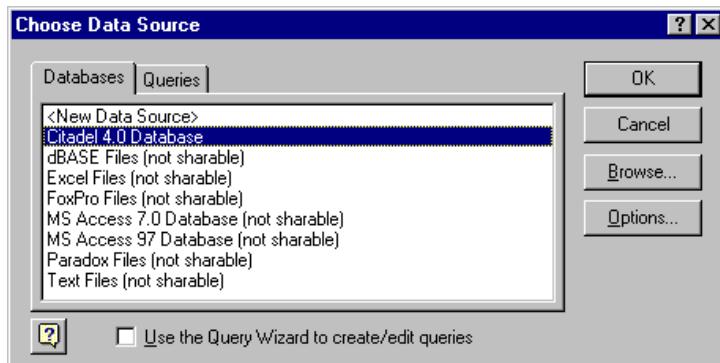
- c. Click on the **Connect** button. The **National Instruments Citadel ODBC Setup** dialog box appears. Fill in the fields as shown in the following illustration. Enter the location of your database in the **Database Path** field.



- d. Click on **OK**. The **Create New Data Source** dialog box reappears, and should look like the following illustration.



- e. Do *not* select a default table or save your ID and password. Click on **OK**. The **Choose Data Source** dialog box appears, this time with the Citadel database as one of your data source choices. Choose Citadel as your data source and click on **OK**.



3. Select the Citadel data source. Make sure the **Use the Query Wizard to create/edit queries** option is *not* selected.

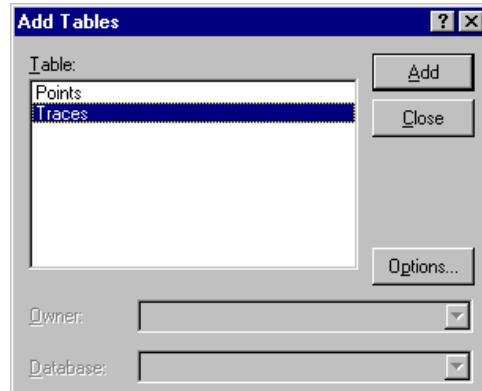


Note If Microsoft Query is unable to connect to Citadel, make sure you have logged data to Citadel and entered the correct database path in the **ODBC Setup** dialog box.



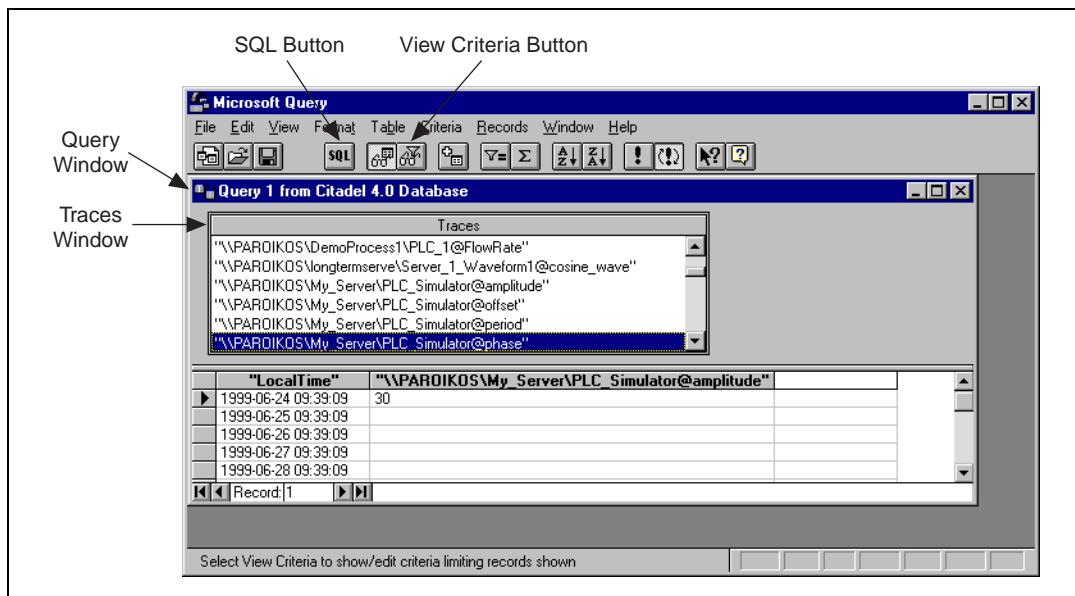
Note If Citadel is not listed in the **Data Source** dialog box, make sure you have created Citadel as a data source. See the *Creating a Citadel ODBC Data Source* section for more information.

4. In the **Add Tables** dialog box, double-click Traces or Points, depending on whether you want to view raw data (Points) or resampled data (Traces). In the following figure, Traces are used.



5. Close the dialog box.

Microsoft Query presents the full Query Window with the Traces and Points tables. The names in these tables are a comprehensive list of all name values that have been logged to Citadel.



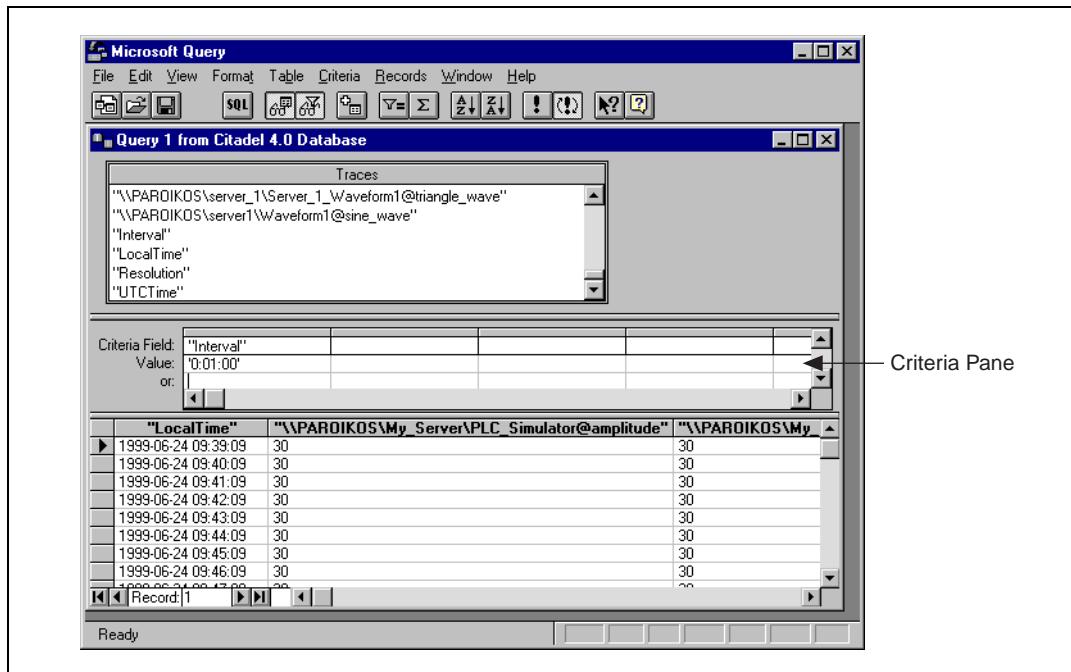
6. To view a trace, double-click on or drag the field you want to view to a blank column in the data pane. In the figure above, LocalTime and

"\\PAROKOS\My_Server\PLC_Simulator@amplitude" have been dragged down.

7. To view a data transform value, enter the function directly into a blank column. For example, to view the minimum value of PLC_Simulator.amplitude, you would enter "min{\\PAROKOS\My_Server\PLC_Simulator@amplitude} ". You must include the quotation marks and braces.

The data set in the figure above was retrieved using no specific criteria, so the ODBC driver used the default. Although there are several ways to specify criteria, this example uses the criteria pane.

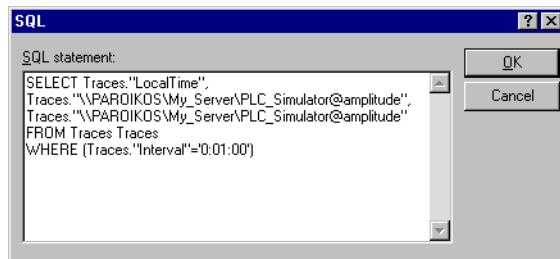
8. Click on the **View Criteria** button. The pane appears in the **Query** window.
9. Add a field to the criteria pane by double-clicking on the field, or by dragging it to the blank column in the criteria pane. In the following example, an Interval criteria of one minute was chosen.



When you enter qualifying criteria values, use the syntax demonstrated in *SQL Examples* earlier in this chapter.

As soon as you specify criteria, Microsoft Query retrieves the specified data. You can save your query at any stage of development, and as you build your query, the application builds an SQL statement.

10. Click on the **SQL** button to view or edit the query statement.



Accessing Citadel Data with Microsoft Excel

1. Launch Excel.
2. Choose **Data»Get External Data»Create New Query**. This Excel command directly launches Microsoft Query.
3. Use an existing query or create a new one. See the section *Accessing Citadel Data with Microsoft Query* for information on querying.
4. When you finish building your query, return the result set with the **File»Return Data to Microsoft Excel** command. Excel prompts you with the **Get External Data** dialog box, enabling you to change or confirm the destination cells of the result set. If you want to query Citadel later to update the result set, check the **Keep Query Definition** checkbox.
5. Click on **OK** to write the data to an Excel worksheet.
6. To update your result set, select any cell within the worksheet result set, choose **Data»Get External Data**, and click on the **Refresh** button.

Accessing Citadel Data with Microsoft Access



Note The SQL/92 standard states that a delimited identifier is any string of no more than 128 characters enclosed in quotation marks. It further states that characters within a delimited identifier are exempt from SQL syntax checking. Unfortunately, Microsoft Access performs its own syntax checking for ODBC queries using a non-standard SQL syntax—even within delimited identifiers. For this reason, National Instruments provides a Convert Special Characters selection in the Citadel ODBC Setup dialog box. When selected, the ODBC driver converts the special characters to accepted Access characters. Refer to Table 8-1 for a complete list of special characters.

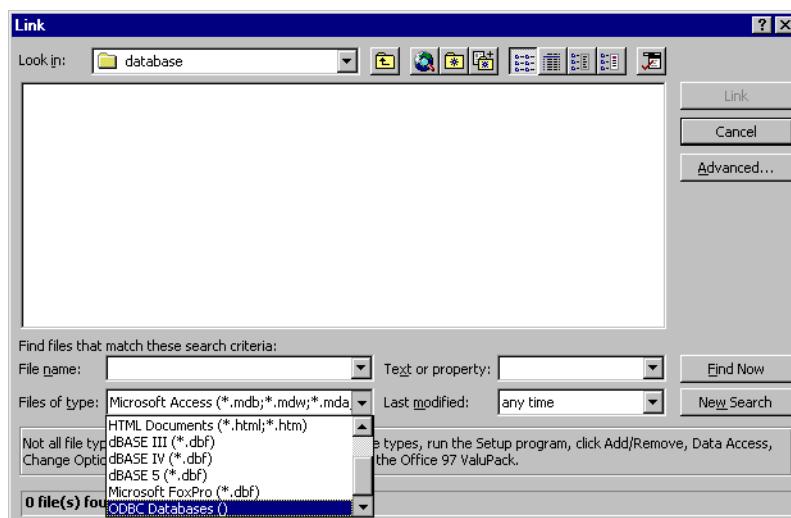
When you query Citadel using Microsoft Access, you must use the Microsoft Access non-standard SQL syntax in your select statement. Remember to consider the following elements:

- Convert special characters for Access compatibility (see Table 8-1)
- Place double quotes around LookoutDirect trace names to identify them as delimited identifiers
- Enclose identifiers in square brackets ([])
- Place pound signs (#) around time stamps

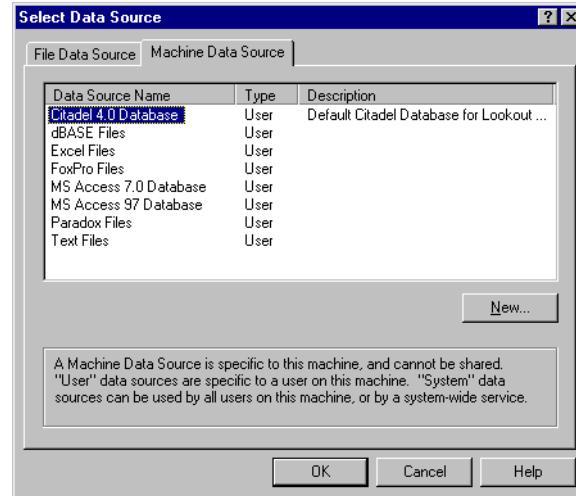
For example, to retrieve LocalTime, Pot1, and Tiway1.V101, where LocalTime is greater than 11/20/95 18:00:00, and where Interval is one second, enter the following query:

```
SELECT LocalTime, ["Pot1"], ["Tiway1@V101"]
FROM Traces
WHERE LocalTime > #11/20/95 6:00:00 PM#
AND Interval = '0:01'
```

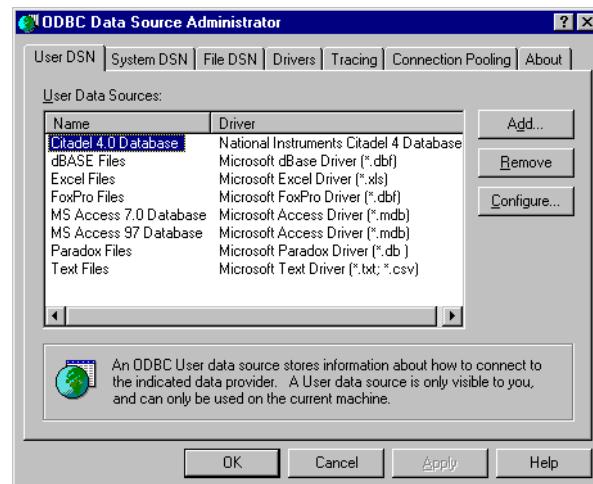
1. To query Citadel from within Microsoft Access, open a database and select **File»Get External Data»Link Table**. The **Link** dialog box appears, as shown in the following illustration.



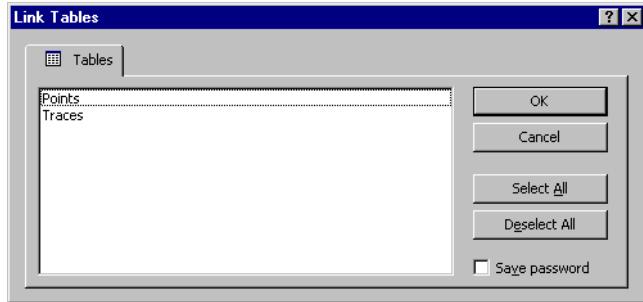
2. Set the **Files of type** field to **ODBC databases()**. The **Select Data Source** dialog box appears.



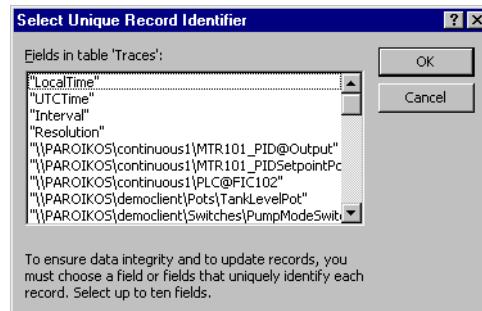
3. Choose the **Machine Data Source** tab.
4. Highlight **Citadel 4.0 Database.dsn**, as shown in following illustration. (This example assumes you have created a Citadel data source as described in the *Creating a Citadel ODBC Data Source* section. If you have not created a Citadel data source, you may not be able to complete this example exercise.)



5. Click on **OK**. The **Link Tables** dialog box appears.



6. Choose **Traces**. The new table links to your database.
7. The **Select Unique Record Identifier** dialog box appears. Click on **Cancel**, because LookoutDirect does not support unique record identifiers.



8. At this point, you can build queries in Access to extract data directly from the Citadel database. See your Microsoft Access documentation for more detailed information on this process.

Accessing Citadel Data with Visual Basic



Note Visual Basic software relies on Microsoft Access DLLs for performing ODBC queries. Because it uses the Access non-standard SQL syntax, be sure that **Convert Special Characters** is selected in the Citadel **ODBC Setup** dialog box. Refer to the note under *Accessing Citadel Data with Microsoft Access*.

You use the Citadel ODBC Driver in Visual Basic the same way you would use any other ODBC driver. To retrieve and view data, you have to create a data control and at least one text control.

1. Place a data control on an open form.
2. Set its **Connect** property to DSN=Citadel and double-click on its **Record Source** property to identify Traces as its source table.
3. If you want to select all of the data for all traces in the Citadel database, leave the **Record Source** property set to Traces. If you want to narrow your query, enter an SQL select statement in the **Record Source** property.

For example, to retrieve LocalTime, Pot1, and AB1.I:3 where LocalTime is greater than 11/20/95 18:00:00 and less than 18:30:00, and where Interval is one minute, enter the following query:

```
SELECT LocalTime, ["Pot1"], ["AB1@I:3"]
FROM Traces
WHERE LocalTime > #11/20/95 6:00:00 PM#
AND LocalTime < #11/20/95 6:30:00 PM#
AND Interval = '1:0'
```



Note Remember to use the Microsoft Access SQL syntax in the select statement, convert special characters for Access compatibility (see Table 8-1), place double quotes around LookoutDirect trace names to identify them as delimited identifiers, enclose identifiers in square brackets ([]), and place pound signs (#) around time stamps.

4. Place a text control on the form.
5. Set its **Data Source** property to the name of your data control—for example, Data1.
6. Click on the **Data Field** property to highlight it and then use the property sheet drop-down combo box to select the desired field name. All logged data members should be listed including LocalTime, Interval, and Pot1.
7. Repeat steps 4 through 6 for each data member you want to display on your form.

If you are using Visual Basic 4 or later, you might get a Visual Basic warning telling you Object variable or with block variable not set.

The following steps should help if this problem arises.

1. Make sure that the Citadel ODBC is properly installed. Select **Control Panel»32-bit ODBC** and choose the **Users DSN** tab. You should see your Citadel data source listed. If you do not, you must create it. See *Creating a Citadel ODBC Data Source* for more information.
2. Make sure you have placed a Data Control on your open form.
3. Set the **Connect** property of the Data Control to **ODBC;DSN=Citadel** (or any other name you defined in the **Users DSN** tab in step 1). You should now be able to see **Traces** in the **Record Source** property.

Alarms and Events

The Lookout*Direct* alarm service keeps track of error messages and any process elements you have defined alarm conditions for. You can define alarm conditions for most Lookout*Direct* objects and data members. You can filter, display, log, and print alarms. You can also organize alarms into hierachal areas to aid in performing the other alarm tasks. All Lookout*Direct* alarms and events are automatically logged to the Citadel database.

When the alarm display window is visible, a pair of thin lines serve as a cursor. You can scroll through the alarms with your keyboard cursor keys, selecting individual alarms with the spacebar.

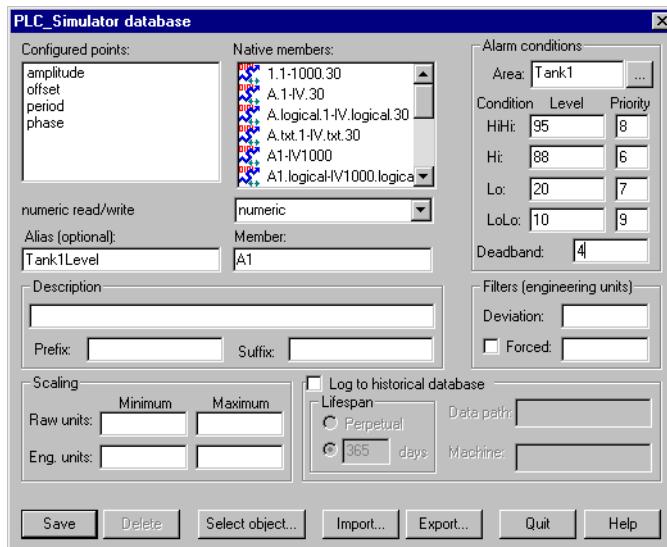
Right-clicking in the alarm window opens the **Alarms** menu, which includes options you can use to acknowledge alarms and access all other alarm properties.

Defining Alarm Conditions

You can create an alarm condition in two ways. First, you can define alarm parameters when modifying an object database (*Database-Generated Alarms*). Second, you can use Alarm objects to define alarm conditions (*Alarm Object* method).

Database-Generated Alarms

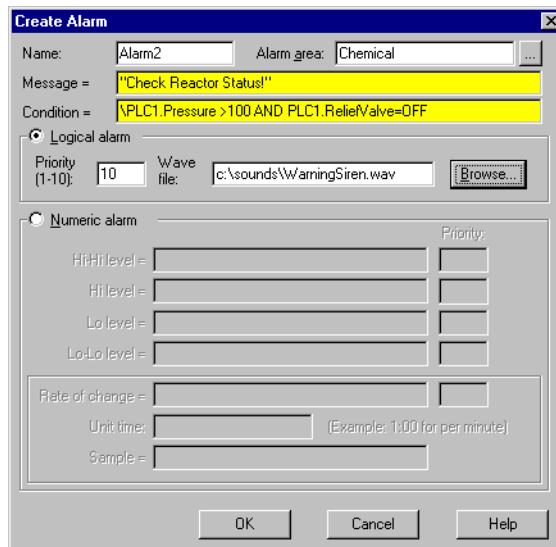
Most objects contain numeric or logical data members. If they exceed specified conditions, you want Lookout*Direct* to generate an alarm. These conditions can include high, low, high-high, and low-low. Select **Object»Edit Database** or right-click on the object in the Lookout*Direct* Object Explorer to modify alarm parameters for an object data member in its database dialog box as illustrated in the following figure. You can browse for an existing alarm area for the alarms, or create a new one.



This method offers the most efficient way to define standard or simple alarm conditions. Furthermore, this method is especially useful when you configure alarms for data that originate from objects with large databases, like Driver objects and DataTable objects.

Alarm Objects

For dedicated Alarm objects, you define and create Alarm objects on an individual basis with the **Object>Create** command, or by right-clicking on a process in the LookoutDirect Object Explorer. The **Create Alarms** dialog box appears, as shown in the following illustration.



You can create alarms that are triggered by any event or combination of events you specify in the **Create Alarm** dialog box using as simple or complex an alarm criterion as needed. You do not have to use the traditional set of alarm conditions (that is, High, Low, Rate-of-Change). See the *Alarm* topic in your online help or in the online PDF *LookoutDirect Object Reference Manual* for additional information.

Alarm objects are not the only source of alarm signals. Most object classes generate their own alarm signals—Driver, Spreadsheet, and Expression objects can also generate alarms.

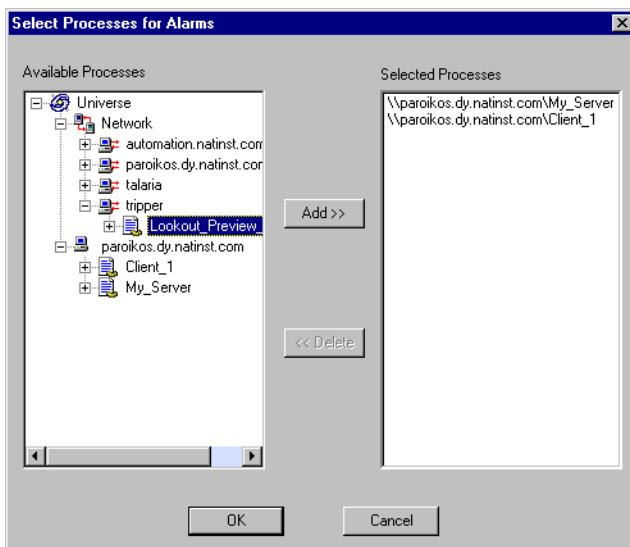
Several object classes generate circular reference alarms. A circular reference alarm defines a condition where the signal generated by an object is sent back to that object as a parameter, either directly or indirectly. Circular references are always priority 10 alarms, and you should correct them during process file testing.

The Alarm Subsystem

You use the alarm subsystem, which is comprised of several components, to control how alarms are displayed, acknowledged, printed, grouped into areas, prioritized, and filtered. Before you implement a structure for your alarming system, you should understand these various components.

Selecting Processes to Monitor for Alarms

Before you can monitor alarms or events running on other computers, you must add the processes those events might rise from to the list of processes your monitoring computer is keeping track of. To select processes to monitor, choose **Alarms»Select Processes**. The following dialog box appears.



Select the processes in the **Available Processes** list that you want to monitor and click the **Add** button. To stop monitoring a process, select the process in the **Selected Processes** list and click the **Delete** button.

Alarm Areas

For organizational purposes, you can classify alarms into areas. Alarm areas allow operators to filter unwanted alarms and acknowledge alarms on an area-by-area basis.

You can create any number of alarm areas and assign any number of alarms to any area. However, you should carefully plan your alarming structure so operators can filter alarms by meaningful categories. With a carefully planned

structure, you can handle specific areas and/or priorities if your system experiences a large number of alarms.



Note One reason to decide on your alarm area arrangement in advance has to do with maintaining alarm data continuity. If you move alarm notification from one area to another, only new notifications will be stored in the new area. To access old alarms, you have to search in the original area.

LookoutDirect has hierachal organization of alarms based on alarm areas. When you create an alarm area, *LookoutDirect* creates a folder in the global \$Alarm object to hold those alarms. These folders make browsing for alarm areas when setting filters more convenient.

When you first start *LookoutDirect*, it has one default alarm area called *LookoutDirect*. There are also a number of alarm areas built into *LookoutDirect* that become visible when you create or open a process that uses one of the objects associated with those areas. Table 9-1 lists the various alarm areas.

Table 9-1. *LookoutDirect* Alarm Areas

Alarm Area	Description
<i>LookoutDirect</i>	<i>LookoutDirect</i> default alarm area. Includes alarms not directed to a specific alarm area.
Serial Ports (comm)	Alarms involving serial ports and serial communication.
Citadel	Alarms concerning the Citadel database.
DDE	Alarms generated by any DDE object or action.
Disk	Disk alarms.
Math	Mathematical errors, such as an attempt to divide by zero.
Print	Printer alarms.

You create your own alarm areas by entering an alarm area name in the **Area** field of the **Alarm conditions** section of the database dialog box or in the **Alarm area** field of the Alarm object.

After you create or activate a *LookoutDirect* alarm area, that area becomes a part of your Citadel database and appears in the alarm area hierarchy, whether or not the process directing alarms to that area is running.

See the *Alarm* object in the LookoutDirect online PDF *Object Reference Manual*.

Alarm Priorities

Assign each alarm a priority level ranging from 1 to 10, where Priority 1 is the lowest priority alarm and Priority 10 is the highest priority alarm.

Priority 1 alarms do not require operator acknowledgment. When they are deactivated, Priority 1 alarms are removed from the alarm window by LookoutDirect, regardless of acknowledgment.

You should assign Priority 1 to only minor, inconsequential alarms. Often, priority 1 alarms do not exist in control strategies. If an occurrence is significant enough to be classified as an alarm condition, you should assign a priority level that requires operator acknowledgment.

Priority 2 and higher alarms require operator acknowledgment before they are removed from the alarm list.

The **Data files location** parameter in the **System Options** dialog box specifies the root directory for the Citadel database.

Alarm Window

The **Alarm** window lists all active and unacknowledged alarms. Alarms are listed in chronological order with the most recent alarm at the top of the list. If there are too many alarms to see at once, you can use the scroll bar at the right of the **Alarm** window to scroll through the list.

If you have your Alarm window display options set to minimize the window, you can view the alarm window by pressing <Ctrl-A>, selecting **Alarms>Show**, or clicking on the alarm indicator box on the far right side of the LookoutDirect status bar at the bottom of the screen, shown in the following figure.



To quickly identify alarm status, use the LookoutDirect **Alarm** window color scheme, as defined in Table 9-2.

Table 9-2. Alarm Colors

Color	Alarm Status
Red	Active
Blue	Unacknowledged, Inactive
Red and Black	Acknowledged, Active (Alarm information, including time, description, priority, and name, appears in black and the area in red).

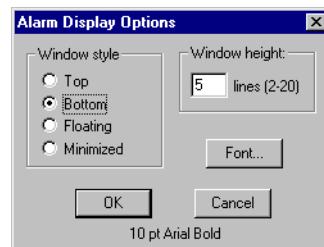
When you acknowledge an inactive alarm or an acknowledged alarm deactivates, *LookoutDirect* removes the alarm from the alarm window. Because a new line is added to the list every time the alarm activates, *LookoutDirect* might list the same alarm condition multiple times in the alarm window.

When you acknowledge an alarm or a block of alarms, *LookoutDirect* prompts you to enter a comment, which is logged to the database along with the acknowledgement time and operator. This comment is optional, and you can omit entering it if there is nothing to make a note of.

To view detailed information about a particular alarm, right-click on the alarm in the alarm window and select **Properties**.

Alarm Display Options

The **Alarm** window displays unacknowledged alarms that meet alarm filter requirements you set with the **Alarm Filter** dialog box. With the **Alarms»Display Options** command, you can change the display style of the **Alarm** window. The **Alarm Display Options** dialog box appears, as shown in the following figure.



The **Window style** determines the position of the alarm window in the *LookoutDirect* workspace. If you select **Floating**, the alarm window appears

as a pop-up style control panel that you can resize and move on the screen. You can minimize a floating alarm window at any time.

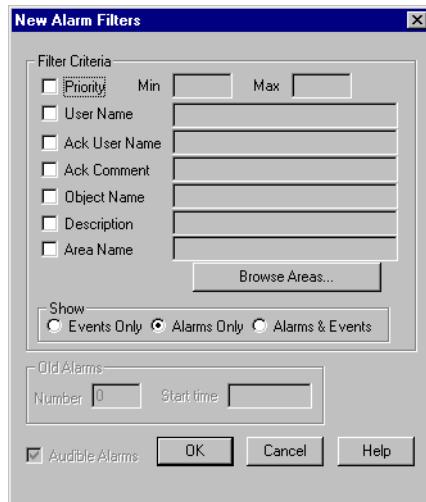
If you use either the **Top** or **Bottom** window type, the **Window height** specifies the number of alarms LookoutDirect can display in the **Alarm** window. The actual height of the **Alarm** window adjusts automatically depending on the selected font and **Window height** setting. You can resize a floating **Alarm** window at any time with the sizing border. If more alarms occur than can be displayed in the **Alarm** window at once, a scroll bar appears along the right side of the window.

The following illustration shows an example of an **Alarm** window with **Window style** set to **Bottom** and **Window height** set to 4.

AlarmsWindow					
Time	Process	Area	P..	Object Name	Description
14:20:44.2	\kali\Saturn_5_Station	Saturn\Station5\Sector3\FuelCells	4	\kali\Satur...	Temperature on fuel cell 4 exceeds specification.
14:20:44.2	\kali\Saturn_5_Station	Saturn\Station5\Sector7\Pumps\Gas	7	\kali\Satur...	Clogged Intake Valve
14:18:16.5	\kali\Saturn_5_Station	Saturn\Station5\Sector4\Conveyers	9	\kali\Satur...	Conveyor 9 is jammed

Alarm Filters

To filter the alarms that appear in your alarm window, select **Alarms>Filter Options**, or right-click in the alarm window and select **Filter Options**. The following dialog box appears.



To monitor alarms with specific priorities, set the **Min** and **Max** values of the **Priority** criterion.

Set **User Name** to restrict your alarm monitoring to alarms generated while that particular user is logged on. You can only select one user name at a time, but you can use wild card characters to widen the scope of the alarms reported.

Set **Ack User Name** to restrict your alarm monitoring to alarms acknowledged by that particular user. You can only enter one user name at a time, but you can use wild card characters to widen the scope of the alarms reported.

The **Ack Comment** filter restricts your alarms displayed to those with the specified acknowledgement comment.

Set **Object Name** to restrict your alarm monitoring to alarms involving the name you enter. You can only enter one name at a time, but you can use wild card characters to widen the scope of those objects reported.

Set **Description** to restrict your monitoring alarms that meet your criteria. You can only choose one description category at a time, but you can use wild card characters to widen the scope of the alarms reported. The LookoutDirect categories HiHi, Hi, Lo, or LoLo are added as a prefix to any descriptions and are ignored by description filtering.

Set **Area Name** to restrict your monitoring to the alarm area you choose. You can only enter one alarm area at a time.

Use the **Browse Areas** button to locate and select the alarm area you want to use as a filter.

You can choose to have the LookoutDirect **Alarm** window show alarms only, events only, or both alarms and events by checking the appropriate box in the **Show** section of this dialog box.

Use the parameters in the **Old Alarms** section to display alarms after they have been acknowledged.

Select **Audible Alarms** to enable a sound alert when an alarm takes place. The sound depends on your windows system setting for error sounds.

The **Alarm Alert** box in the right corner of the status bar displays the number of alarms currently visible in the alarm window.

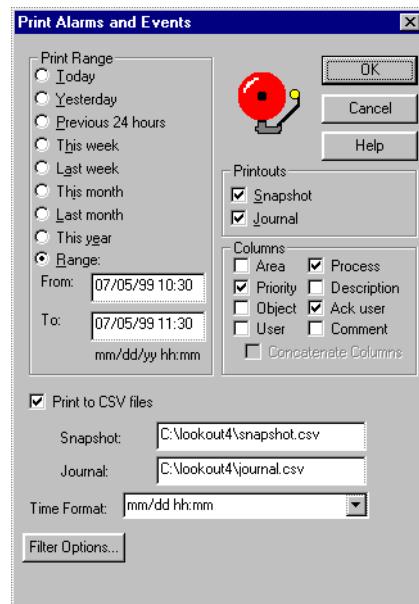


If you are filtering alarms, alarms that do not meet your filter criteria do not show up in this status bar count. To see all alarms, click on the **Display All**

button in the **Alarm Filters** dialog box, then click on **OK**. The alarm window displays all active and unacknowledged alarms. The status bar reflects the change.

Alarm Print

You can print alarms in LookoutDirect, based on your filtering. To print your alarms, select **Alarms»Print**. The following dialog box appears.



Select the time range you want to print alarms and events from with the items in the **Print Range** section of the dialog box. Notice that when you define your own range, you use month, day, and year, followed by hour and minute.

The **Printouts** selections determine the exact alarm information included in your printout. **Snapshot** only prints the status of alarms at the beginning of the specified **Range** but does not indicate what happened during the time span. **Journal** creates a printout of everything that happened during the time span from the beginning of the **Range**.

Specific information about each alarm is presented in columnar format. Lookout*Direct* prints only the information you designate. Select which columns you want printed in the **Columns** section of the dialog box.

To print to a comma separated file (.csv), select the **Print to CSV file** option. Enter the file names for your journal and snapshot files, including a complete

path to where you want the file written. If you enter a file name only, LookoutDirect will create the file in the LookoutDirect directory.

Set the format for printing times in the **Time Format** list.

You can adjust your alarm filters for printing by clicking on the **Filter Options** button to access the alarm filter options.



Note You can print alarms as they happen by specifying a printer port in the **Log alarms to** field of the **System Options** dialog box, accessed by selecting **Options»System** from the menu bar. This works well for a printer directly connected to your computer. To print alarms directly to a network computer, you have to capture a port in the network printer driver and link it to your networked printer. Consult your operating system documentation for detailed instructions on how to capture a port for a printer driver.

Alarm Acknowledgment

Before you can acknowledge an alarm, you must select it. To select an alarm, click on the alarm line in the **Alarm** window. Highlighting indicates that the alarm is selected. To deselect an alarm, click on the alarm line. <Ctrl-click> to select multiple individual alarms; <Shift-click> to select blocks.

You can access all the acknowledgement options from the Alarms menu item, or by right-clicking in the **Alarm** window.

Right-click an alarm in the window and choose **Select All** to select all alarms for acknowledgment. You can also select **Acknowledge All**. Because selecting each alarm individually can be very time consuming, either technique can be especially useful if your process is experiencing a high number of alarms.

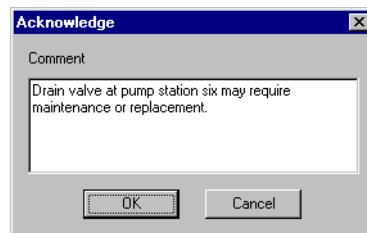
Right-click and select **Deselect All** to deselect all alarms that were previously selected for acknowledgment. If you want to deselect only some of the alarms, <Ctrl-click> on the individual alarms you want deselected.

Selecting **Acknowledge** acknowledges alarms that you have selected for acknowledgment. If you have not selected alarms when you invoke this command, the following message box appears.

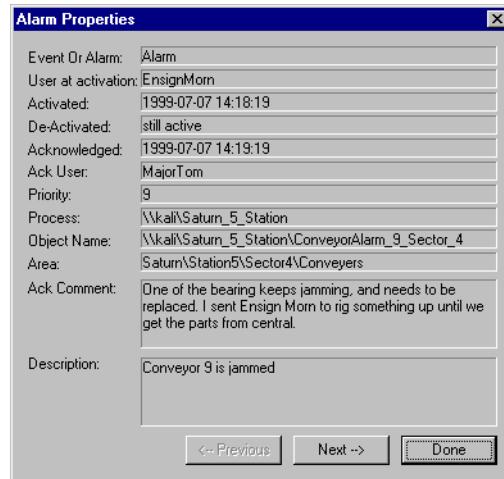


If you select one or more alarms for acknowledgment, a dialog box appears for an operator to enter a comment concerning the alarm. Comments are optional, and you can click on **OK** to finish acknowledging alarms without entering a comment.

You can search the database for alarms or print out the alarms based on comments, so using certain standard comments (in addition to comments on specific circumstances) can make the filtering process easier. You should create your own master list of standard comments if you want your operators to use them.



You might want further information about a specific alarm or event. Right-click on an alarm and select **Properties**. The following dialog box listing specific information about the alarm or event appears.



You can scroll through alarms and events using the **Previous** and **Next** buttons.

Acknowledging Alarms Programmatically

By connecting to the appropriate data members of the global \$Alarm object, you can acknowledge alarms programmatically with Pushbuttons or other logical expressions. This way, you can bypass the menu commands, or acknowledge alarms from remote Lookout*Direct* network nodes. For instance, you could create a single Pushbutton that acknowledges all alarms, or you could create multiple Pushbuttons for acknowledging specific areas of alarms.

With the \$Alarm object, you can access numeric signals that indicate the number of currently active alarms or the number of unacknowledged alarms in a particular area. For more information, refer to *\$Alarm* in your online help or in the online PDF *LookoutDirect Object Reference Manual*.

10

Redundancy

With *LookoutDirect*, you can configure two process control computers for redundancy, providing automatic transfer of control if the primary computer should fail. Redundancy only applies to systems that implement networking of multiple *LookoutDirect* nodes. It does not apply to stand-alone systems. You should understand process files (.L4P) and state files (.L4T) before you attempt to use this chapter.

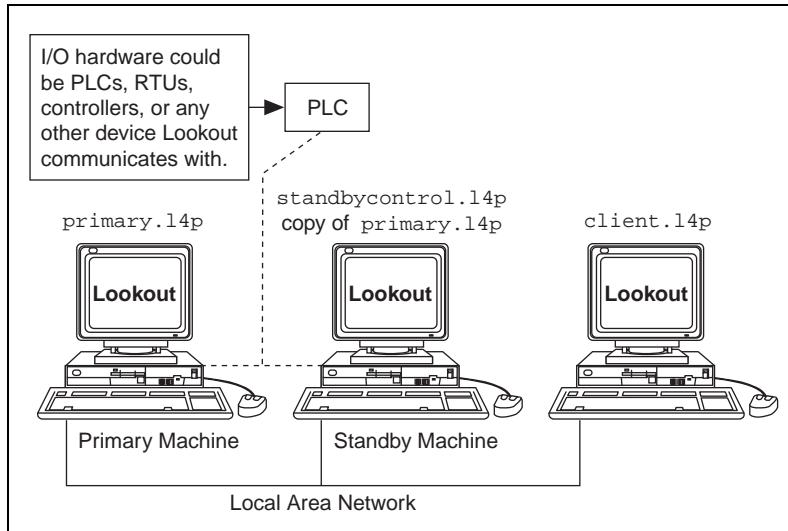


Note In versions of *LookoutDirect* earlier than *LookoutDirect* 4, redundancy was handled through NetDDE. National Instruments recommends you use *LookoutDirect*'s TCP/IP-based networking when possible, but you can still use your old NetDDE-based processes with the newer version of redundancy. However, you must equip your backup computer with a monitor and load process like the one described in the *Setting up the Standby Process and Computer* section of this chapter. You must also modify the process properties of both primary and standby processes to handle state file and data logging paths appropriately, as described in the *Implementing Redundancy* section.

No other alterations should be necessary to upgrade your redundancy processes, though you should test the new arrangement thoroughly. The need remains for following NetDDE naming conventions as well as the other instructions given in Appendix A, *Networking With DDE*. If your client process uses NetDDE in shifting access from one computer to another, it will continue to work in the current version of *LookoutDirect*.

You set up computer redundancy using the *LookoutDirect* Monitor, Loader, and Symbolic Link objects.

Standby refers to a network configuration in which one computer is designated the *primary* computer and another is designated the *standby* computer. The primary computer normally monitors and controls the process while the standby computer monitors the primary computer. If the standby computer quits receiving information from the primary computer, the standby computer automatically takes over and assumes the role of the primary computer. The following illustration depicts a typical *LookoutDirect* network structure with the standby option configured.



Standby Basics

There are three basic principles assumed in the illustration shown above. First, both the primary *and* standby computers must have direct access to your field I/O. If the primary computer fails for any reason, you cannot rely on it anywhere in your backup strategy. Your standby computer must have the same direct access to all your I/O, because it will assume the responsibility of the primary computer. There are several ways to accomplish this, depending on your hardware, network, and communications topology.

The second principle is that you can designate any node on the network as the standby computer. It does not need to be the closest physical computer on the network, though this is typically the case due to the restraints imposed by the first principle. During normal operations (that is, while the primary computer is functioning properly), the standby computer is just another computer on the network.

However, the standby computer should also have a copy of the primary computer process file in its LookoutDirect directory, PRIMARY.L4P in this example. As soon as the standby computer fails to receive a signal from the primary computer, it automatically loads its local copy of the PRIMARY.L4P file. From that point on, the standby node assumes the role of the primary computer, and all other computers running LookoutDirect processes automatically recognize this fact. At the same time, client processes accessing a server running on the primary computer switch from accessing the primary to accessing the standby.

When the primary computer comes back up on the network, the standby computer and any client processes affected return to their previous operation.

The third standby principle takes no additional configuration on your part, but is important to understand for peace of mind. A stand-alone *LookoutDirect* application is responsible for periodically updating its own state file. In a standby system configuration, the primary computer not only updates its own state file, but also the state file of the standby computer. When a failover occurs, the standby computer takes over where the primary computer left off, ensuring bumpless transfer of control (meaning a transfer without discontinuities in control values).

Failover Scenarios

There are four reasons for the standby computer to take control of the process.

- The primary computer is down (that is, it is not turned on, Windows is not running, or an application fault has locked the computer).
- The primary computer is running, but it is not running *LookoutDirect*.
- The primary computer is running *LookoutDirect*, but it is not running the correct process file.
- The primary computer is running the correct process file within *LookoutDirect*, but the two computers cannot communicate over the network.

The purpose of the standby feature is to provide automatic failover to a second computer, restoring your *LookoutDirect* network to its original setup (that is, with the primary computer running its `PRIMARY.L4P` file and the standby computer running its monitor process).

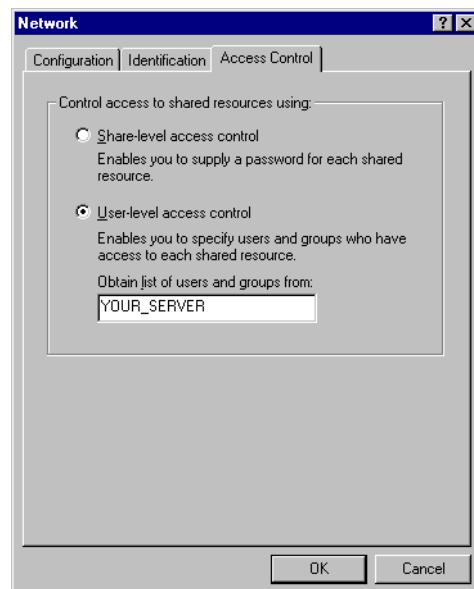
Configuring Standby

For *LookoutDirect* to provide standby operation, all the computers involved must be properly set up on your network. If your computers passed the ping test described in the *Testing TCP/IP* section of Chapter 1, *Installing LookoutDirect*, in the *Getting Started with LookoutDirect* manual, you should be able to implement redundancy with no difficulties. If you cannot ping each computer necessary for your backup system to work, consult your network administrator.

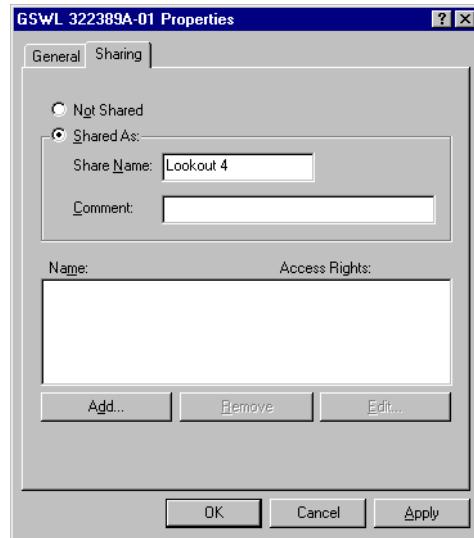
Enabling File Sharing in Windows 98/95 and Windows NT

One part of proper redundancy is making sure that *LookoutDirect* has the most recent state file for a process available when it loads that process. This maximizes your chance of a bumpless transfer of control. For this to occur, your computers must have file sharing permission with each other.

1. Configure the computers you need to share files between using the sharing options in the Network control panel, as shown in the following illustration.



2. Select the *LookoutDirect* directory on the standby computer and configure it to be shared, as shown in the following illustration.



3. Using the **Add** option, add the authorized computers you want to be able to share to. Make sure full access privileges are granted.

Time Synchronization with Standby Computers

If timing and time-stamped data are crucial to your backup system, consider synchronizing both your primary and secondary computers to a third computer.

In any case, you should make certain that your primary, backup, and client computers are all appropriately synchronized. See the *Time Synchronization* section of Chapter 4, *Networking*, for detailed information about time synchronization.

Implementing Redundancy

To set up your redundancy system, you must configure your primary and standby processes correctly and incorporate certain *LookoutDirect* objects in primary, standby, and client processes.

Redundancy Overview

A *LookoutDirect* Monitor object watches the process running on the primary computer. When that process disappears from the network, the Monitor signals the other elements of the *LookoutDirect* redundancy system into action. When the primary computer and process reappear on the network, the Monitor object signals the standby process and clients to return to their previous mode of operation.

A *LookoutDirect* Loader object handles the job of loading and unloading the standby process on command from a *LookoutDirect* Monitor object.

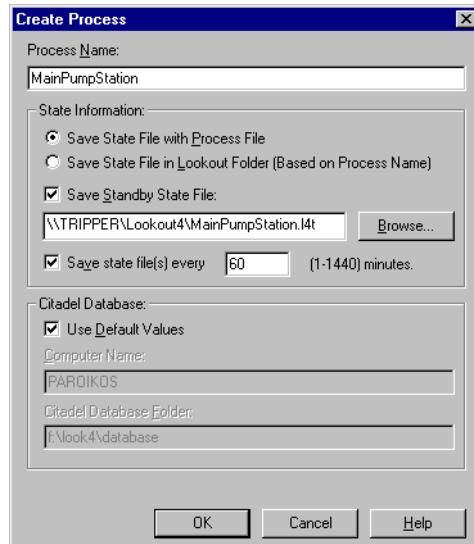
A *LookoutDirect* Symbolic Link redirects a client process from interaction with the primary process to interaction with the secondary process when signaled by a Monitor object. When the Monitor object signals that the primary process is back on line, the client switches back to interacting with the primary process.

For detailed information on the Monitor, Loader, and Symbolic Link objects used in implementing redundancy, see your *LookoutDirect* online help or the online PDF *LookoutDirect Object Reference Manual*.

Setting up the Primary Process and Computer

If your primary process should go down, you want the transfer to be as bumpless as possible. For this to happen, your standby process needs to open with the most current *LookoutDirect* state file (.14t) available. To do this, configure the **State Information** options of your process.

1. When you create your primary process, you do so using the **Create Process** dialog box. If you are setting up redundancy on an existing process, you can access these same options by right-clicking on the process name in the *LookoutDirect* Object Explorer and selecting **Properties**. In either case, the following dialog box appears.



2. Select **Save State File with Process File** to save the state file in the location where the process file was opened from. This is your local state file for your process.
3. Select the **Save Standby State File** checkbox to save one (or more) copies of the state file for your standby process. Enter a complete path, including state file name, to each location to which you want to save a state file. If you are saving the state file to more than one backup or alternative location, separate the paths with the vertical bar (|) operator symbol.

In the preceding illustration, Lookout*Direct* saves the state file to the local directory containing your process file and to the Lookout*Direct*4 directory of the standby computer, named TRIPPER in this case.

Remember that you must make sure your primary computer has write permission to the Lookout*Direct*4 directory of TRIPPER for this backup state file to work.

When your primary computer goes off line, your process begins to run on your standby computer. The state file the standby computer uses to initialize its version of your process file must be in the directory containing the standby copy of the process file.

When your process opens, Lookout*Direct* checks the dates of the state files you have selected and uses the most recent file as the state file to initialize values in your process.

4. Set how often Lookout*Direct* saves the state file in the **Save state file(s) every [xxxx] minutes** field. The Lookout*Direct* default is 60 minutes. Your state file save rate should be the same in your primary and backup processes.

Any time Lookout*Direct* opens this process, it will open with the most recent state file available.



Note The process file you use on your standby computer must be identical to the process file on your primary computer it is backing up except in one respect. The standby version of the process file need only save its state file to the location specified by your primary computer's process file. In this way, when your primary computer comes back online, it can locate the most recent version of the state file to use in initializing.

You might think that you would need to alter your process file before moving a copy of it to your standby computer. The Lookout*Direct* Loader object, however, overrides a number of settings for any process it loads, including the state file information. Therefore, you do not need to alter your process file in any way when you move a copy of it over to your standby computer to be used for redundancy.

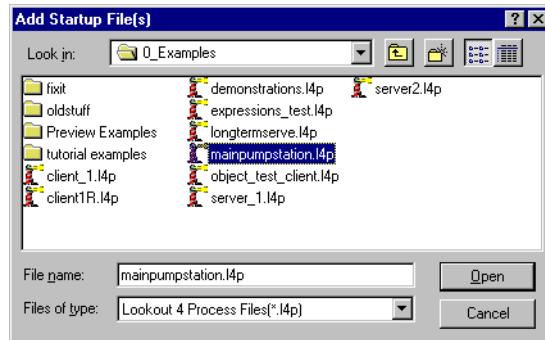
Make sure that when your primary computer comes back on line after being down that Lookout*Direct* loads as soon as the computer boots, and that Lookout*Direct* then loads the proper process.

To make sure your computer launches Lookout*Direct* when it boots, consult your operating system documentation and follow the instructions for your computer. After you have made sure Lookout*Direct* will launch when your computer boots, follow these steps to make sure Lookout*Direct* runs your primary process.

1. Select **Options»Startup**. The following dialog box appears.



2. To add a file to your list of startup processes, click on the **Add** button. A dialog box you use to browse for a file appears. You can autoload as many processes as you want on startup.



3. Select the file you want to run when *LookoutDirect* opens, then click on **Open**.
4. Click on **OK**

Repeat these steps for each process running on your primary computer that you need to run on your standby computer.

You can also reboot a computer, load *LookoutDirect*, bring a state file up to date, and load a process manually if you choose or if it becomes necessary.

Setting up the Standby Process and Computer

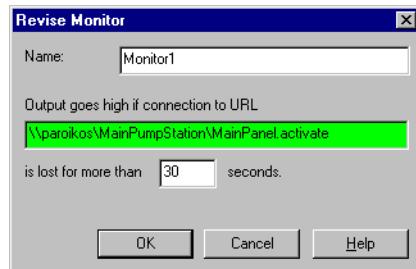
Your standby computer must be connected to your *LookoutDirect* network and running a copy of *LookoutDirect* for *LookoutDirect* redundancy to work.

The standby *LookoutDirect* must run a simple process to monitor the process being run on your primary computer. When data ceases to be available from the primary computer, this process loads the standby copy of the primary process. When the primary computer comes back on line and data begins to flow from the primary process again, this same process unloads the standby version of the primary process.

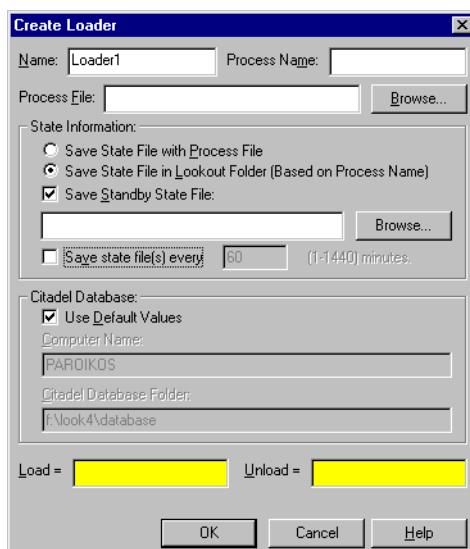
Create this standby control process using the following steps.

1. Create a new process. In this example, it is named `StandbyControl`. You do not need to have a front panel for this process, at least in this simple case. You can cancel the **Create Panel** dialog box if you choose.

2. Create a Monitor object. The following dialog box appears.



3. Enter the URL to some data member in the process you want to back up from your primary computer. The value of this data member does not matter, nor does it matter if that data member is used by any other Lookout*Direct* object. If the Monitor object does not detect the presence of that data member for the amount of time you enter in the **seconds** field, the Monitor will go high (TRUE).
4. Create a Loader object. The following dialog box appears.



5. When using the Loader for redundancy, set the **Process Name** the same as the name of the process you are standing by for. The **Process File** should be identical to the process file you intend it to stand in for.
6. The **State Information** settings are important for achieving bumpless transfer back to your primary computer and process when they come back on line. The process file Lookout*Direct* loads using this Loader object will use the state file you configured your primary process to save

on your standby computer. It is important that you update that same state file while the standby process is running, because if it is more recent than the local state file, *LookoutDirect* on the primary computer will load the updated state file from your standby computer.

Because you earlier set the state file to be located in the *LookoutDirect* directory of your standby computer, you should select the **Save State File in LookoutDirect Folder** option. If you have reason to save copies of the state file in other locations as well, you can select the **Save Standby State File** option.



Note Settings in the Loader object override any settings you might have made when you created or set the properties for your process.

7. Set how often you want the state file saved in the **minutes** field.
8. Set the location for your Citadel database logging to take place, just as you would when creating a new process.
9. In this example, the Monitor object is named `Monitor1`. Enter `Monitor1=TRUE` in the **Load** field, and `Monitor1=FALSE` in the **Unload** field.
10. Click on **OK**.

When the primary computer goes off line and the Monitor object goes high, the Loader object loads the standby process. When the primary computer comes back on line, the Monitor object reverts to low and *LookoutDirect* unloads your standby process.

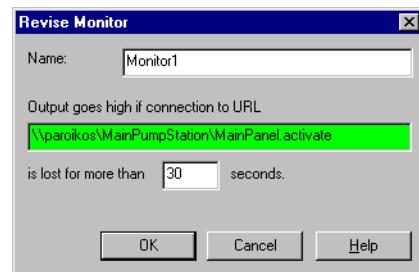
Configuring Clients for Standby Operation

It is most likely that you will use redundancy for server processes on *LookoutDirect*. You may have any number of *LookoutDirect* client processes running on other computers in your *LookoutDirect* network. These clients also must respond to a change from primary to standby computers.

When you create a *LookoutDirect* client process, you always make connections to the server process and not directly to field hardware, frequently by using the remote position source type of connection.

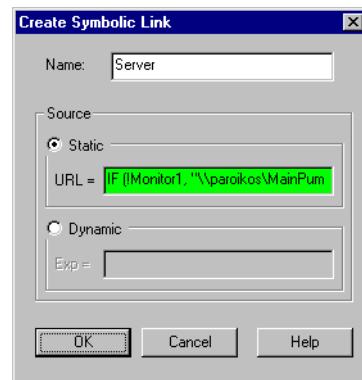
When you create a *LookoutDirect* client for use with a redundancy-equipped system, you use the *LookoutDirect* Symbolic Link to switch your client process access from one process to another. Instead of connecting your client objects directly to objects in another process, you connect them to the symbolic representation of those objects. The Symbolic Link can shift from primary to standby process when necessary and your client will continue to function, having made its connections through the Symbolic Link.

- To create this kind of client process, you begin by making a Monitor object, just as you did with the standby control process.



You should set your Monitor object to the same setting you used in the standby control process on your standby computer.

- You next create a Symbolic Link. The following dialog box appears.



If the primary computer process goes off line for some reason, the Monitor object goes TRUE and your client process will connect to objects in the standby server process running on TRIPPER.

After you have made this preparation, develop your client process as you ordinarily would, except that you make your connections to the MainPumpStation process through your symbolic link instead of with the process itself. Use your Symbolic Link to insert expressions, create remoted controls, and display historical data in HyperTrend.

Editing Object Databases

LookoutDirect creates the *native* database of an object automatically when you create the object. The native database is documented at the end of each object class definition in the online help and in the *LookoutDirect Object Reference Manual*.

You can create new data members through aliasing, or modify the parameters of any existing native data member. These parameters include such things as alarm setpoints, deviation filters, scaling factors, historical logging, and alias names.



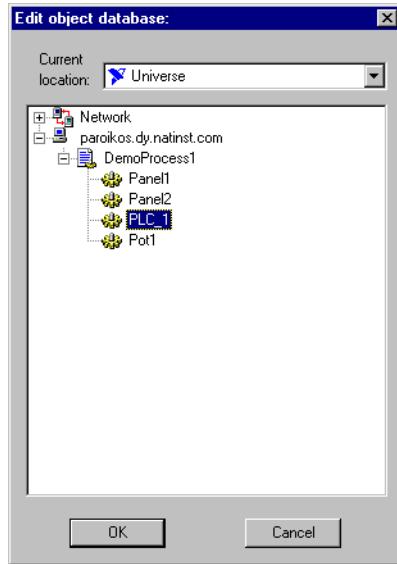
Note Any object in *LookoutDirect* can have its native database modified. However, this is most practical for objects with large native databases, such as driver objects and DataTable objects.

Editing Database Parameters

1. For the purposes of this illustration, create a Modbus object, the same way you created the a Pot object. Use the name `PLC_1` for your object, and accept the default parameters.

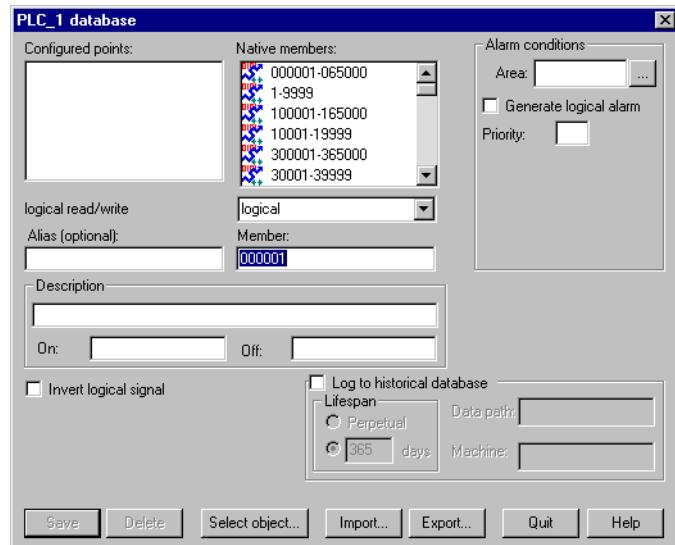
The Modbus object can be added to your system without a physical connection while you are learning to use *LookoutDirect*. Let the default settings stand when you create the object.

2. There are two ways to access the dialog box you use to edit a database.
 - From the *LookoutDirect* menu bar, select **Object»Edit Database**. In the **Edit object database** dialog box, navigate to your demo process running on your local computer and select `PLC_1`.



- In the LookoutDirect Object Explorer, right-click on the object you want to edit the database for, and select **Edit Database**.

The following dialog box appears.



3. Follow these steps for each data member to be configured:
 - a. Identify the desired data member by entering it into the **Member** data field. If you are modifying a data member that has been

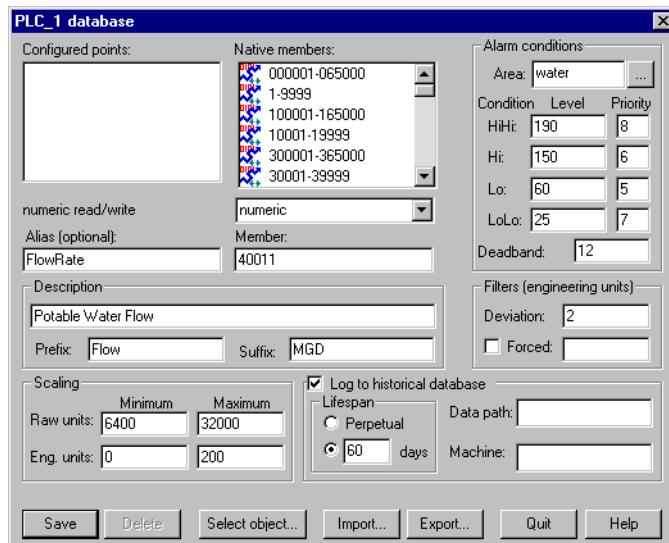
previously configured, you can select it from the **Configured points** list box.

For this example, select 40011 as the modbus data member.
Enter FlowRate as the alias.



Note Parameter fields automatically change depending on the data member you select. LookoutDirect automatically determines whether the data member is logical or numeric, and presents you with the appropriate parameter attributes.

- b. Configure appropriate parameters shown in the following illustration. See the individual parameter sections in this chapter for details on setting each parameter.



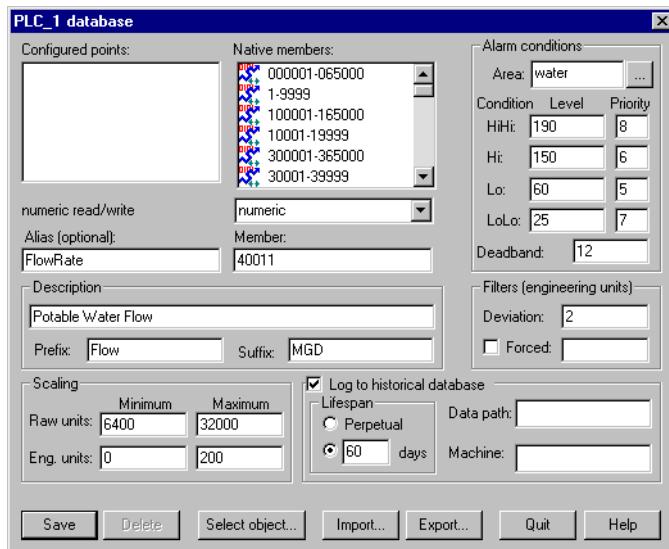
- c. Select **Save** or **Update**. (If you are modifying a data member that was previously defined, the **Save** button changes to an **Update** button.)

LookoutDirect stores all the new parameter settings for the specified data member when you select **Save** or **Update**. In addition, LookoutDirect adds the modified data member to the **Configured points** list box for future reference. LookoutDirect immediately reflects these changes throughout your configuration.

4. Select **Quit** to exit the dialog box.

Numeric Member Parameters

The following diagram and paragraphs describe numeric data member parameters. Logical data members, covered in the following section, share a number of the same parameters.



Alias renames any native data member. You can think of an alias as a sort of nickname. For example, the Modbus driver object includes the native data member 40011 that represents an analog input. You can give this native data member an alias like `FlowRate`. From then on, you can reference the alias `FlowRate` instead of its native name 40011. All associated parameters (such as **Scaling**) are also applied to the alias value.

An alias is a good way to insulate your LookoutDirect configuration from changes in your PLC, RTU, or I/O configuration. For example, consider a flow transmitter wired to an analog input at 40011. You can give 40011 the alias name `FlowRate`, just as you did in the example. Multiple control panels can then display the `FlowRate` data member and numerous other objects in LookoutDirect can use it. If you later rewire the transmitter to the analog input at 40012, you need only modify the alias `FlowRate` to reflect the new I/O address. LookoutDirect instantly reflects this change everywhere `FlowRate` is used.

You can modify all associated parameters of an existing alias *except* the alias name itself. If you attempt to modify an existing alias name, the **Update** button changes to a **Save** button and you will only create a new alias.

Most developers implement aliases on objects with large native databases, such as driver objects (like Modbus and Tiway) and DataTable objects.



Note An alias is optional in some instances. You can apply alarming and logging parameters to any native data member and save it to the *Configured points* list without giving it an alias name. Scaling, however, operates somewhat differently. In order to read scaled data, you read it from the alias, not the native data member.

The **Description** appears as the message text in the alarm window. It can have spaces, and it can be lengthy. You do not have to enter quotes in this field.

Prefix and **Suffix** are part of the description, but do not appear in the alarm window. They are just additional descriptive text.

Define **Scaling** by entering **Raw units** and **Eng. units**. The raw numeric data member is converted (scaled) to an engineering unit value. The PLC in this example generates a raw value ranging from 6,400 to 32,000. LookoutDirect converts that raw signal to range from 0 mgd to 200 mgd. The conversion is linear. See your hardware specifications and calibration records for the minimum and maximum raw units associated with analog devices. If you leave the **Raw units** and **Eng. units** fields blank, LookoutDirect performs no scaling on the signal.

In order to scale this data, you can use an alias in combination with the native data member. Read or write the raw units by connecting to the native data member, and read or write out the engineering units from the aliased value.

Deviation filters out insignificant variations of numeric signals. The following figure shows two values plotted on a trend. One line is the raw unfiltered value. The other, stair-stepped line, represents the filtered value after passing through a **Deviation** of 2.



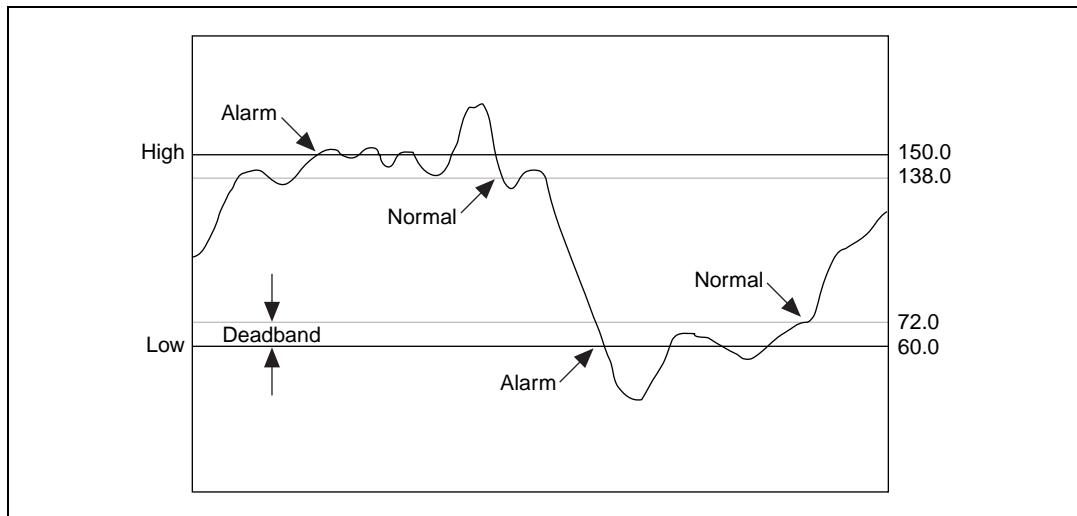
The Citadel database also uses **Deviation** as the criteria which triggers logging of new historical data to disk. See Chapter 7, *Logging Data and Events*, in the *LookoutDirect Developer's Manual* for more information on logging data.

Use the **Forced** data field to manually enter a constant value for the data member. When you select the **Forced** checkbox and enter a value in the field, *LookoutDirect* forces the engineering unit value to be equal to the value you entered, regardless of the actual value of the native data member. You might use this when a sensor fails or during sensor maintenance, or any time a PLC receives a bad signal from the transmitter.

Use the **Alarm condition** parameters to define alarm limits and their associated priorities. *LookoutDirect* compares the alarm setpoints to the engineering units value (that is, the post-scaled, post-filtered number). If you do not enter scaling parameters, *LookoutDirect* applies the alarm parameters directly to the raw signal.

You can assign an alias or native data member to any existing alarm **Area** or you can create a new **Area**. To create a new **Area**, enter the new area name in the field. See Chapter 9, *Alarms*, in the *LookoutDirect Developer's Manual* for more information on alarms.

Use the alarm **Deadband** parameter to prevent fluttering between alarm and normal states when the signal value hovers near an alarm limit. The following figure shows a value plotted against its **Hi** and **Lo** alarm setpoints.



LookoutDirect generates an alarm the moment the value violates the **Hi** or **Lo** alarm setpoints. The alarm returns to normal when the value drops below the high alarm setpoint minus the Deadband, or goes above the low alarm setpoint plus the Deadband. The Deadband also applies to all **HiHi** and **LoLo** alarm limit setpoints.

The **Log to historical database** parameters define how long to store a value in the Citadel database on your hard drive. If you do not select this option, *LookoutDirect* does not log the value to your disk. If any scaling or filtering parameters are defined, *LookoutDirect* logs the scaled, filtered value (that is, the engineering unit value).



Tip If you intend to display data with a *LookoutDirect* Hypertrend object, you must log the data to the historical database.

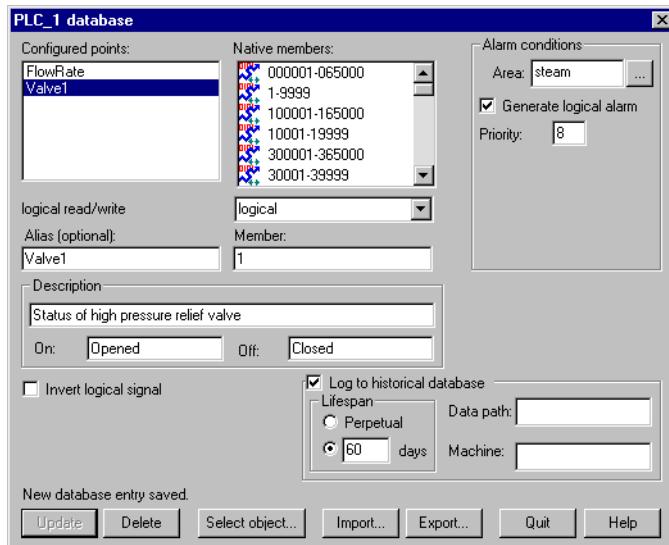
When you log data, it will go to the database directory you set up in the **Create Process** dialog box.

See Chapter 7, *Logging Data and Events*, in the *LookoutDirect Developer's Manual* for more information on logging data.

Logical Member Parameters

Some of the parameters of logical data members are different from those of numeric data members. Scaling of a logical signal consists of the **Invert Logical Signal** checkbox. When you choose this checkbox, an ON value is represented by an OFF value, and so on. When you do not select it, ON is ON and OFF is OFF.

A logical database is shown being configured in the following figure.



Alarm parameters of a logical signal include the alarm **Area** assignment field and the **Generate Logical Alarm** checkbox. When selected, the data member generates an alarm whenever the value is ON; the alarm condition clears whenever the value is OFF. Notice that if the **Invert Logical Signal** checkbox is selected, the value used here is the inverted value. See Chapter 9, *Alarms*, in the *LookoutDirect Developer's Manual* for more information about alarms.

Text Member Parameters

The text data member database contains only **Alias**, **Member**, and **Description** fields. You can log text data to the database, but the only way to query for the logged text string is through the LookoutDirect SQLExec object.

Importing and Exporting Object Databases

Use the import database service to copy database member parameters from an Excel spreadsheet file directly into an object. Use the export database service to copy an object database into an Excel spreadsheet file. This is what you can do using these services:

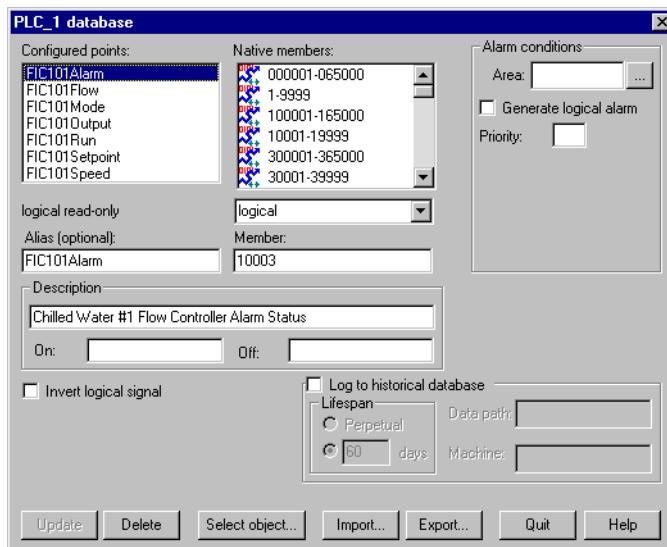
- Export object database parameter definitions to Excel for the purpose of documentation.
- Export an object database to Excel, perform global replacements on data member parameters, and then import the changes.
- Create a name list in Excel or in an application that exports to Excel, then copy that name list into *LookoutDirect*.
- In a process using multiple duplicate driver objects (such as a gas pipeline or water distribution system), define a single driver object database parameters in Excel. Import that database into multiple driver objects.

Although you can import and export any object database, you may find that these services are most useful for objects with large native databases, such as driver objects and Data Table objects.

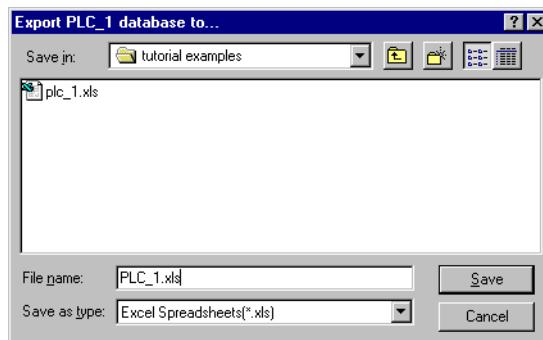
Exporting an Object Database

Follow these steps to export an object database:

1. Open the **Edit Database** dialog box, either through the menu or the Object Explorer, choosing the object you want to export from.
The following diagram shows a Modbus object that already has a number of logical and numeric data members defined.



2. In the database dialog box, click on the **Export** button.



3. In the **Export Object Database to** dialog box, choose a directory path, enter a filename and click on **OK**.

When you export a database, Lookout*Direct* does not export every possible data member (many driver objects have a capacity for thousands of members). Instead, Lookout*Direct* exports configured points; that is, data members that have at least one parameter already defined. Lookout*Direct* also exports native members that are in use (that is, connected to other objects).

The spreadsheet file that the Lookout*Direct Export* command creates is in Excel Version 2.0 format. An example of the .xls file is shown in the following illustration.

	A	B	C	D	E	F	G
1	Command	Alias	Member	Description	Prefix	Suffix	Eng min
2							
3	insert	FIC101Alarm	10003	Chilled Water #1 Flow Controller Alarm Status			
4	insert	FIC101Flow	40001	Chilled Water #1 Flow Rate	CFS		0
5	insert	FIC101Mode	10002	Chilled Water #1 Flow Controller Mode	Auto	Manual	
6	insert	FIC101Output	40003	Chilled Water #1 Flow Controller Output	%		0
7	insert	FIC101Run	10001	Chilled Water #1 Feed Pump	On	Off	
8	insert	FIC101Setpoint	40004	Chilled Water #1 Flow Controller Setpoint	%		0
9	insert	FIC101Speed	40002	Chilled Water #1 Feed Pump Speed	%		0
10							

Creating a Database Spreadsheet

The easiest way to create a database spreadsheet for an object is to create the object in Lookout*Direct*, define the parameters for one logical data member, one numeric data member, and one text member (if necessary). You then export the database to a spreadsheet file.

This creates a basic spreadsheet file with all the necessary column labels for Lookout*Direct* to read. The three data members you edited in Lookout*Direct* furnish examples you can follow to rapidly create or edit more data members of the same type, using the convenient tools of your spreadsheet program.

Notice the figure above, that row 1 contains column labels. These include the names of all possible data member parameters. Lookout*Direct* references the labels of a spreadsheet file, not the column letters, so if you were to create a spreadsheet to import into Lookout*Direct* manually, you would have to spell all column labels correctly. Lookout*Direct* ignores white space and is case-insensitive. The following list contains all possible column labels, in the order in which Lookout*Direct* inserts them when creating a spreadsheet file.

- Command
- Member
- Alias
- Description
- Prefix
- Suffix
- Eng min
- Eng max
- Raw min

- Raw max
- Invert?
- Deviation
- Force?
- Forced value
- Alarm area
- Lolo level
- Lolo priority
- Lo level
- Lo priority
- Hi level
- Hi priority
- Hihi level
- Hihi priority
- Logical priority
- Log data?
- Lifespan

LookoutDirect requires the column labels **Command** and **Member**. All other column labels are optional. **Command** must be in cell A1. The order in which the other column labels appear makes no difference.

Any time you import a database, *LookoutDirect* reads the first 30 columns (A–AD) and ignores columns that do not have labels.

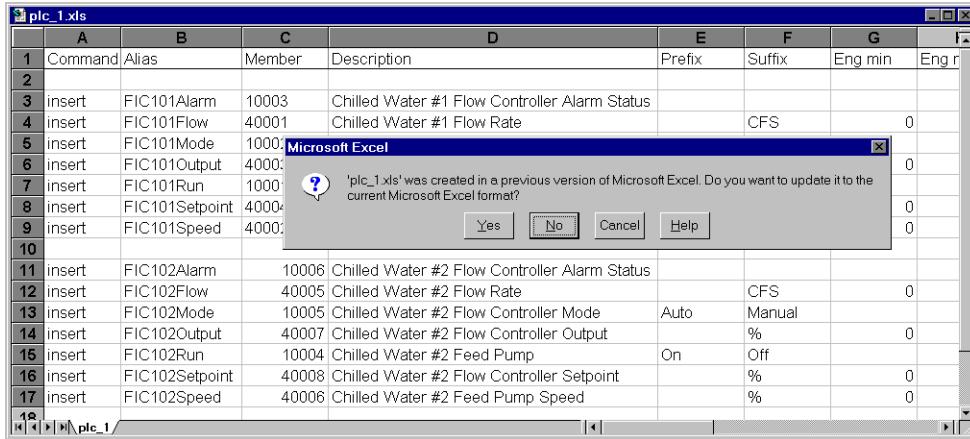
The rows below the column labels (below row 1) each represent a database data member. For example, row 5 in the spreadsheet in the previous figure represents the data member whose alias name is FIC101Mode.

You can easily add rows to define new data members. Copy the rows associated with FIC101, for example, and then modify the new rows slightly by identifying different native members and giving them new aliases and descriptions.

After you have edited and expanded your spreadsheet file, you can save it and import the new values back into *LookoutDirect*, adding all the new data members to those you configured earlier as template samples.

If you are working with a version of Excel more recent than 4.0, the program asks you if you want to update your spreadsheet to a newer format when you

select **File»Save**. This dialog box is shown in the following illustration . Be sure to select **No**. Lookout*Direct* does not currently accept Excel spreadsheet files from versions greater than 4.0.



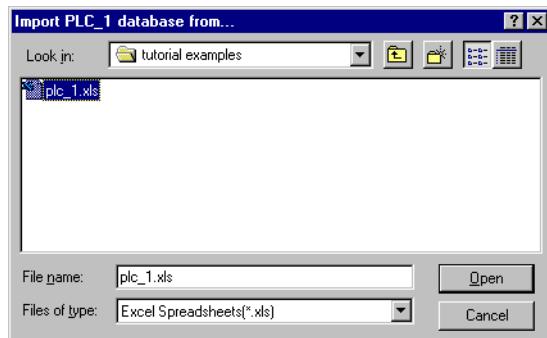
Importing an Object Database

Editing a spreadsheet file for several similar data members, and then importing it as an object database, can be much faster than working in the Lookout*Direct* dialog boxes one at a time. When you import a database, Lookout*Direct* reads the first 30 columns (A–AD) It ignores columns that do not have labels and columns beyond AD.

Each row in the Command column (column A) contains either the keyword `insert` or the keyword `delete`. When you import a database, Lookout*Direct* ignores rows that do not have the `insert` or `delete` keyword. It adds those records whose command keyword is `insert`. It removes those records whose command keyword is `delete`. To determine exactly which records to delete, Lookout*Direct* uses the record alias name; or if the record does not have an alias name, it matches the record member name.

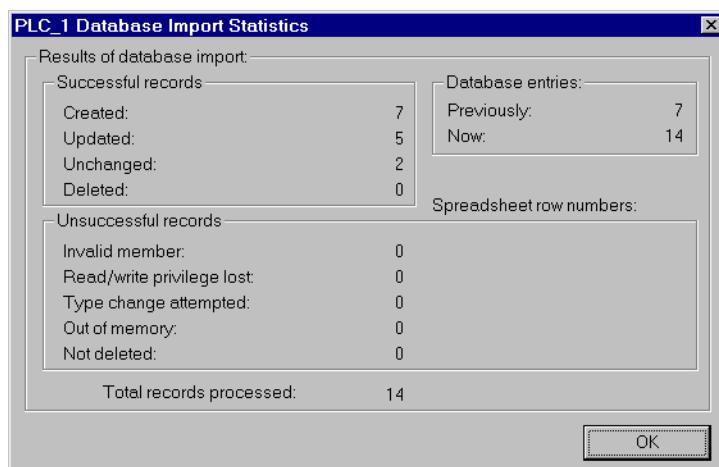
Follow these steps to import an object database:

1. Open the **Edit Database** dialog box, either through the menu or the Object Explorer, choosing the object you want to import from.
2. In the object database dialog box, click on the **Import** button.

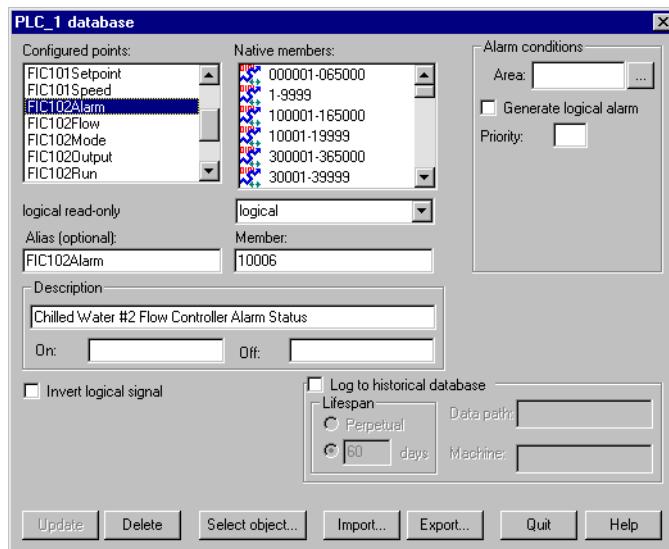


3. In the **Import Object Database from** dialog box, choose a directory path, select the filename and click on **OK**.

When finished, LookoutDirect presents you with a set of database import statistics, as shown in the following illustration.



As you can see in the following figure, the data members added to the database spreadsheet were successfully added to the **Configured points** list.



Copying an Object Database

The import and export features make it easy to copy the database of an object. This is especially useful when creating large SCADA applications, such as gas pipelines with multiple compressor stations.

The key to defining multiple driver objects that require duplicate databases is to first create an object in Lookout*Direct* for each RTU or PLC. Then create a single database in Excel. Next, import that database into each driver object.

Glossary

Prefix	Meanings	Value
m-	milli-	10^{-3}
k-	kilo-	10^3
M-	mega-	10^6

A

absolute date absolute time	Numeric system Lookout <i>Direct</i> uses for keeping track of dates and times, in which midnight (0 hours), January 1, 1900 is represented by 1, midnight of January 2, 1900 is represented by 2, and so on. The absolute date/time number 36234.47222250 represents 11:20 A.M., March 15, 1999. The numeric value for 1 second in Lookout <i>Direct</i> is .000011574, the numeric value for 1 minute is .000694444, and the numeric value for 1 hour is .041666667.
ACK	Acknowledge (an alarm or event).
active notification	A feature of event-driven software systems in which the application is alerted of value changes when they occur instead of through continuous, loop-driven queries.
address space	An OPC term for the area you browse to find what items are available on an OPC server. Part of the standard OPC interface, this space may arrange items hierarchically.
alarm	Software notification of a condition in a process. This alarm may call attention to a value that has exceeded or fallen below certain levels, set in the object database or in an Alarm object.
alias	Name given to a data member using the Edit Database dialog box. This name can be descriptive or mnemonic, and can be associated with other data member configurations such as scaling, logging, and alarming. A data member can have more than one alias, each with different associated configurations.

B

baud rate	Measurement of data transmission speed, formally defined as the number of electronic state changes per second. Because most modems transmit four bits of data per change of state, is sometimes misused or misunderstood—a 300 baud modem is moving 1200 bits per second. <i>See bps.</i>
.bmp files	Graphic files in bitmap format. If you are using a .BMP file in <i>LookoutDirect</i> , you cannot resize it on screen. <i>See Windows metafile.</i>
bps	Bits per second—measure of the rate of transfer of data.

C

CBL compiler	<i>LookoutDirect</i> uses the CBL (Control Block Language) compiler to compile a <i>LookoutDirect</i> source file (.1ks) into a binary file (.14p).
.cbx file	A <i>LookoutDirect</i> file containing a <i>LookoutDirect</i> object class. A .CBX (Control Block Extension) file can have one or more object classes in it.
checksum	A method of verifying that the number of bits received is the same as the number of bits transmitted. Used by TCP/IP and serial protocols.
Citadel	The <i>LookoutDirect</i> historical database that stores your data for access later.
classes	<i>See object classes.</i>
client	A <i>LookoutDirect</i> process that monitors a <i>LookoutDirect</i> server process. <i>LookoutDirect</i> clients should be computer independent so that they can be run from any computer on your network. <i>LookoutDirect</i> server processes run on computers actually connected to your control hardware.
comm port	Term sometimes used for a serial port.
connection	Input to a <i>LookoutDirect</i> object's writable data members. For more information, refer to Chapter 4, <i>Using LookoutDirect</i> , in your <i>Getting Started with LookoutDirect</i> manual.
control objects	<i>LookoutDirect</i> objects you use to control a process, change a data value, adjust a register, and so on.
controllable objects	<i>LookoutDirect</i> objects you can control with a <i>LookoutDirect</i> control object.

.csv files	Comma Separated Value file, a format widely accepted by spreadsheet and other data handling programs.
CTS	Clear to Send. Part of a handshaking protocol for certain devices that connect the serial port of a computer. See the <i>RTS/CTS Handshaking Settings</i> section of Chapter 3, <i>Serial Port Communication Service</i> , for detailed information.
cursor (data table)	The Lookout <i>Direct</i> data table can activate one row of data at a time using the data table cursor. See the Data Table reference in the online help or the <i>LookoutDirect Object Reference Manual</i> .

D

DAQ	Short for Data AcQuisition.
data member	Data source or sink associated with a Lookout <i>Direct</i> object. A readable data member, or source, can be used in expressions or as inputs to other objects. A writable data member, or sink, can have at most one connection into it, created using the Object»Edit Connections dialog box. A data member can be both readable and writable. <i>See also</i> native data member and <i>alias</i> .
data type	Kind of value (numeric, logical, or text) that a parameter or data member can hold.
database	Collection of data stored for later retrieval, display, or analysis.
datagram	Message sent between objects in Lookout <i>Direct</i> . A datagram contains a route and a value.
DCOM/COM	Distributed Component Object Model, a Microsoft standard in which client program objects request services from server program objects. The Component Object Model (COM) is a set of interfaces, clients, and servers used to communicate within the same computer (running Windows 98/95 or Windows NT).
DDE	Dynamic Data Exchange, currently used in Lookout <i>Direct</i> to exchange data with other programs (such as Microsoft Excel) running on your network.

Glossary

deadband	A value that must be exceeded for an alarm to sound or a change in state to be recorded. For instance, if you have a low-level alarm set at 5 with a deadband of 2, the alarm will not trigger until the value being monitored drops to 5. The alarm will then stay active until the value being monitored moves above 7. A deadband keeps small oscillations of value from triggering an alarm and then canceling it too rapidly.
deviation	Set a deviation to filter out small changes in value when logging data. Before being logged to a database, a value must change by at least the deviation amount of the last logged value.
dialing prefix	Part of the Hayes AT command set for use with modems. See the <i>Dial-Up Modem Settings</i> section of Chapter 3, <i>Serial Port Communication Service</i> , for detailed information.
displayable objects	A Lookout <i>Direct</i> object class that has a displayable component, such as a Pot, a Switch, or a Pushbutton.
DLL	Dynamic Link Library, which is a collection of small, special-purpose programs which can be called by a larger program running on the computer. Sometimes called Dynamically Linked Library.
driver objects	Lookout <i>Direct</i> objects used to communicate with PLCs, RTUs, and other I/O devices.

E

edit mode	Lookout <i>Direct</i> mode in which you can alter and create objects within a process. Switch in and out of edit mode by pressing <Ctrl-space> or by selecting Edit»Edit Mode .
engineering unit	In Lookout <i>Direct</i> , used to refer to scaled or converted data. Thermocouple data, for instance, arrives in volts as the raw unit, and must be converted to degrees, an engineering unit.
environment services	Tasks Lookout <i>Direct</i> performs as a part of making your SCADA/HMI work easier. Lookout <i>Direct</i> environment services include serial communications, database and logging, security, networking, alarming, and so on.
Ethernet	A widely used, standardized local area networking technology, specified in the IEEE 802.3 standard.

event	Anything that happens can be an event. In Lookout <i>Direct</i> , events include such things as adjusting a control value, entering or exiting edit mode, opening or closing a control panel, and logging in or logging out of the system.
expression functions	Mathematical, logical, and other functions used by Lookout <i>Direct</i> expressions.
expressions	Lookout <i>Direct</i> expressions are often paths to a data member. They can also function like variables that, using a spreadsheet cell-type formula, become capable of performing flexible, real-time math operations, condition testing, and other complex operations functions. See Chapter 1, <i>Expressions</i> , for more information on expressions.

F

failover	A failover is the takeover of a process by a standby computer when the primary computer fails for any reason.
FieldBus	An all-digital communication network used to connect process instrumentation and control systems.
FieldPoint	A National Instruments hardware product line for industrial automation, control, monitoring, and reporting.
frame	Sequence of bytes sent from a computer to a device or vice versa. The syntax of the frame depends on the protocol being used. A read frame contains enough information to specify a set of variables whose values the device should return. A write frame specifies a variable in the device and a new value to write into that variable. Some protocols support the writing of multiple variables in a single frame. A response frame is returned from the device to the computer, indicating whether the frame just sent to it was received successfully. If the frame just received was a read frame, the response frame contains a set of requested values.
functionality	The way an object works, operates, or performs a task. Functionality is a general concept that applies in the same way to all objects in a given object class. Parameters define the specific functionality of an individual object.
functions	<i>See</i> expression functions.

Glossary

G

gray proximity A term used in Lookout*Direct* color animation. This sets what percentage of gray will be replaced by a given color as conditions change in a monitored value or set of values.

H

Hi and HiHi Alarm settings. Both warn that a value has gone above some setpoint. Generally a Hi alarm is used to alert an operator of a need for intervention. A HiHi alarm is usually used to alert an operator that the value has been exceeded by an even greater margin than a Hi alarm indicates, and is usually used to indicate an urgent need for action.

historical logging The process of storing data in a database for use at another time, or from another location.

HOA Hand-Off-Auto control, used to set whether a value must be changed manually, is completely turned off, or functions automatically. You can use a Pot object and a complex expression to create this sort of control in Lookout*Direct*, or you can use a RadioButton object, depending on the particular requirements of the task you need to accomplish.

I

I/O point Every read-only, write-only, or read-write connection Lookout*Direct* makes to external hardware is counted as an I/O point. Lookout*Direct* is licensed for use with a set number of I/O points. If you exceed the number you are licensed to use with your copy of Lookout*Direct*, a warning message appears on your computer screen warning you to shut down one of your processes within a specified time before Lookout*Direct* cuts back on I/O usage.

(implicit) data member A Lookout*Direct* data member containing the fundamental data for certain object classes. When you make a connection to an (implicit) data member, you only use the name of the object, not the name of the object followed by the data member name.

L

.l4p files	File extension for Lookout <i>Direct</i> process files. These are the compiled files Lookout <i>Direct</i> runs when it runs a process.
.l4t files	File extension for a Lookout <i>Direct</i> state file, which stores the values for Lookout <i>Direct</i> controls and other objects with state information.
.lka files	File extension for Lookout <i>Direct</i> security files.
.lkp files	File extension for Lookout <i>Direct</i> process files in versions of Lookout <i>Direct</i> earlier than Lookout <i>Direct</i> 4.
.lks files	File extension for a Lookout <i>Direct</i> source file, which Lookout <i>Direct</i> compiles to make a Lookout <i>Direct</i> process file that Lookout <i>Direct</i> can run. This is the file you should make sure you keep backed up in case you need to recreate a corrupted process file, or in case some future version of Lookout <i>Direct</i> cannot run a process file compiled in an earlier version of Lookout <i>Direct</i> .
logging	The process of storing data in a computer database file. See Chapter 7, <i>Logging Data and Events</i> , for more information on logging data in Lookout <i>Direct</i> .
logical data member	A Lookout <i>Direct</i> data member of the logical data type.
.lst files	Extension for the Lookout <i>Direct</i> state file in versions of Lookout <i>Direct</i> earlier than Lookout <i>Direct</i> 4.

M

multiplex	A method of working with more than one data stream using only one communications channel. There are a number of different methods of multiplexing, depending on the hardware and software being used. A number of Lookout <i>Direct</i> driver objects support multiplexing hardware.
-----------	---

Glossary

N

native data member	Data members built into a <i>LookoutDirect</i> object class, as opposed to data members you create by using aliases.
NetDDE	A way of networking using DDE (dynamic data exchange), retained in <i>LookoutDirect</i> 4 for compatibility with earlier versions of <i>LookoutDirect</i> .
numeric data member	A <i>LookoutDirect</i> data member of the numeric data type.

O

object	A specific instance created from an object class.
object classes	Software modules you use to create individual objects to perform tasks in <i>LookoutDirect</i> .
object connections	Software links between objects used to transmit data and commands from one object to another.
ODBC	Open DataBase Connectivity, a standard application programming interface (API) for accessing a database. You can use ODBC statements to access files in a number of different databases, including Access, dBase, DB2, and Excel. ODBC is compatible with the Structured Query Language (SQL) Call-Level Interface. ODBC handles SQL requests by converting them into requests an ODBC database can use.
OPC	OLE for Process Control, an industry standard interface providing interoperability between disparate field devices, automation/control systems, and business systems. Based on ActiveX, OLE, Component Object Model (COM), and Distributed COM (DCOM) technologies.

P

parameter	Input to an object, similar to a writable data member, whose value is specified in the object parameter list in a <i>LookoutDirect</i> source (.LKS) file. Typically, parameter values are set in the object Object»Create or Object»Modify dialog box.
ping	A small utility program in Windows and DOS that checks to see if a computer can be reached across a network. Also used to indicate the running of that program.

pixel	Picture Element, the smallest bit of a picture. Has one color or shade of grey. The number of pixels per inch determine the resolution of an image.
PLC	Programmable Logic Controller.
poll	A software event in which a computer checks some value in a device or register. In Lookout <i>Direct</i> , a logical command that forces a device poll to check data member values.
poll rate	How often a device is polled.
pop-up panel	One variety of Lookout <i>Direct</i> control panel that can only be displayed at the size set by the process developer, and which cannot be maximized. When open, a popup panel remains on top of other panels until minimized.
process	In Lookout <i>Direct</i> , process refers to a Lookout <i>Direct</i> “program”, used for industrial automation, control, monitoring, or reporting.
process file	The Lookout <i>Direct</i> binary file Lookout <i>Direct</i> executes when running a process. Carries the .14p extension.

R

raw unit	Data as it arrives in your process, such as voltage or amperage. Thermocouple data, for instance, arrives in volts as the raw unit, and must be converted to degrees, an engineering unit.
receive gap	A serial communications setting that determines the number of empty bytes (or amount of time) a driver receives before recognizing the end of a message frame and requesting another message. See the <i>Setting Receive Gap</i> section of Chapter 3, <i>Serial Port Communication Service</i> , for more information about the receive gap.
redundancy	A system for making sure that a computer can come online and run a Lookout <i>Direct</i> process if the computer currently running that process fails for some reason.
remote	In the context of Lookout <i>Direct</i> , remote is a position source location for a control. See the <i>Remote Position Source</i> section of Chapter 4, <i>Using Lookout<i>Direct</i></i> , in the <i>Getting Started with Lookout<i>Direct</i></i> manual for detailed information on the Lookout <i>Direct</i> remote position source.
resolution	The smallest signal increment that can be detected by a measurement system. Also, the number of pixels per inch on a computer monitor screen or dots per inch in printer output.

Glossary

RTS	Request to Send, part of a handshaking protocol for certain devices that connect the serial port of a computer. See the <i>RTS/CTS Handshaking Settings</i> section of Chapter 3, <i>Serial Port Communication Service</i> , for detailed information.
RTU	Remote Terminal Unit, a device similar to a PLC for use at a remote location, communicating with a host system through radio or telephonic connections.
run mode	Lookout <i>Direct</i> mode in which processes run but no editing changes can be made. Switch in and out of run mode by pressing <Ctrl-space> or selecting Edit»Edit Mode .

S

SCXI	Signal Conditioning eXtensions for Instrumentation, a National Instruments product line for conditioning low-level signals.
security accounts	Also called user and group accounts, Lookout <i>Direct</i> uses security accounts to define what users or group of users have different operation privileges in Lookout <i>Direct</i> . See Chapter 6, <i>Security</i> , for detailed information on Lookout <i>Direct</i> security.
server	A process that provides data (services) to client processes. In Lookout <i>Direct</i> , server processes are intended to be run on one computer only, with direct connections to field hardware. Client processes interact with field hardware through server processes.
source file	Lookout <i>Direct</i> file that can be compiled to produce a binary Lookout <i>Direct</i> process file that runs a process. Uses a .lks file extension.
SQL	Structured Query Language, used to get information from and update information in a database.
standby	A computer standing by to take over running a process if the primary computer fails or falls offline.
startup file	A Lookout <i>Direct</i> process file (.14p) you designate in the System Options dialog box that Lookout <i>Direct</i> will open and run any time Lookout <i>Direct</i> is opened.
state file	The Lookout <i>Direct</i> file that stores the value of all Lookout <i>Direct</i> control parameters and object data members in use in a process. Uses the file extension .14t.

system objects Lookout*Direct* objects used to control other objects or process and analyze data.

T

TCP Transmission Control Protocol, a method (protocol) for sending data between computers. Used with IP, the Internet Protocol.

TCP/IP TCP/IP sends data as packets, with IP handling the delivery of data and TCP keeping track of the individual packets.

text data member Lookout*Direct* data member used for text data.

trace A term for data from a single source over some period of time, stored in an ODBC-compliant database.

traces table ODBC databases present data in the form of traces tables. A traces table contains a field or column of data for each data member being logged, along with a field you can use to query the database.

trend Historical data showing the change in a value over time. Often used in connection with graphing the data for display.

W

.wav files File extension given to sound files. You can play a .wav file in Lookout*Direct* to add sounds or speech to alarms or events.

Windows metafile A standard graphics file type for use in the Microsoft Windows operating environment. If you use a metafile graphic in Lookout*Direct*, you can enlarge or reduce it on the screen, use them as masks without specifying transparent pixels, and use the Lookout*Direct* Animator to animate the colors of the graphic.

.wmf files File extension given to Windows Metafile graphic files.

X

.xls files File extension given to Microsoft Excel files.

Glossary

A

Networking With DDE

This appendix explains how to network *LookoutDirect* using DDE. This information is retained in *LookoutDirect* 4 for compatibility with earlier versions of *LookoutDirect*.

While in *LookoutDirect* 4 you can do most process networking with the faster and more efficient TCP/IP, you might want to use DDE to communicate with other applications, such as Microsoft Excel (as described in Chapter 5, *Dynamic Data Exchange*).

You can use *LookoutDirect* DDE networking capabilities to do the following:

- View the same or different screens simultaneously on separate nodes
- Make setpoint adjustments from any node
- Acknowledge alarms from any node
- Configure specific nodes for monitoring only
- Configure a peer-to-peer architecture
- Configure a client-server architecture
- Configure network nodes to communicate through standard telephone lines
- Configure network nodes to communicate through radios

Networking multiple *LookoutDirect* nodes is a powerful and advanced capability. This appendix refers to many concepts and terms explained elsewhere in the manual. Before you attempt to work with networking, make sure you have a strong understanding of the *LookoutDirect* architecture and are comfortable working within *LookoutDirect* on a single node.

To network using *LookoutDirect*, you must meet the following requirements:

- Purchase and install on each computer a separate, licensed copy of *LookoutDirect*.
- Equip each computer on the network with compatible network hardware.
- Install Microsoft Windows NT/98/95 on each computer.

Because *LookoutDirect* is a native Windows application, it supports all the networks that Microsoft Windows supports, including Microsoft Windows

Appendix A

Network, Microsoft LAN Server, Novell NetWare, Banyan VINES, 3Com 3+Open, DEC Pathworks, IBM LANServer, and so on.

NetDDE Networking Considerations

Networking with Lookout*Direct* involves passing data back and forth between two or more Lookout*Direct* nodes. In Lookout*Direct*, data is passed between and contained within objects. There are two basic methods for passing data between multiple Lookout*Direct* nodes: *Multilink Networking*, which requires a NetDDE (Network Dynamic Data Exchange) link for every value being passed between the nodes, and *Table Networking*, which implements a *data concentrator* at each node where only tables (which concentrate the data) are linked between nodes. Although both methods involve connecting objects over a network using NetDDE, the implementation and total effect can be quite different.

If you plan to use one of the NetDDE methods, you have to configure automatic NetDDE activation and possibly trusted DDE shares. See the *Running NETDDE.EXE Automatically* and *Adding a Trusted DDE Share* sections in this appendix for setup instructions.

There is one more method for implementing network-like capabilities between multiple Lookout*Direct* nodes. Unlike the first two methods, however, this technique does not rely on data being passed between Lookout*Direct* nodes. Instead, you use PLCs, RTUs, or I/O to share data between Lookout*Direct* nodes. This method is called *Hardware Networking*.

Multilink NetDDE Networking

Multilink networking consists of passing data from an object on one computer to another object on a different computer. When linked, any change to the value of one object is instantly propagated to and reflected by the second object on the other computer. The way in which data is passed between two objects across a network depends on whether they are controllable objects.

Linking Controllable Objects

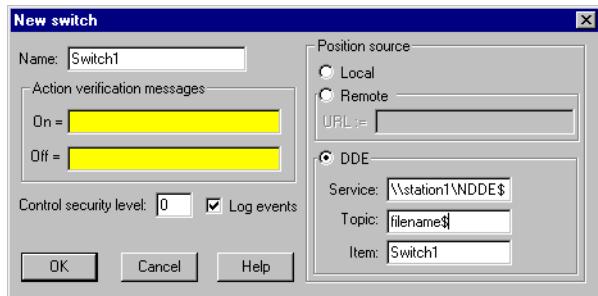
User-controlled objects have a DDE option. These objects include Pot objects, Switch objects, and Pushbutton objects.

Controllable objects each have three **Position source** options: **Local**, **Remote**, and **DDE**. When you choose DDE, you instruct the object to take its value from a remote source. This could be a cell in a spreadsheet, another DDE-aware application, or a second copy of *LookoutDirect* running on the network.

Linking Controllable Objects Together Across a Network

You need at least two computers connected by a functioning network to use the following example. This example refers to the two computers as Station1 and Station2.

1. At Station1, create a Switch object called `Switch1`. Leave **Position source** at the default setting of **Local**.
2. At Station2, create a corresponding Switch object called `Switch1`. However, change the **Position source** to **DDE**. (Although you are not required to give objects matching names over a network, it makes your documentation easier to follow and your applications easier to maintain.)



3. In **Service**, enter `\station1\NDDE$`, where `Station1` is the network name of the other computer. The backslashes, NDDE, and dollar sign are required by Microsoft Windows.

4. In **Topic**, enter `Filename$`, where `Filename` is the name of the `LookoutDirect` process file on the other computer.
Again, the dollar sign is required but the `.14p` file extension is not.
5. In **Item**, enter the name of the object at the other computer that you want to link to (in this case, the other object is also called `Switch1`).
6. Click on **OK**.

Test your network by flipping the switch. When you toggle the switch at either station, it instantly flips at the other workstation. You can add to, modify, delete, or revise your logic on either computer, completely on-line, and still maintain the network link between the two objects.

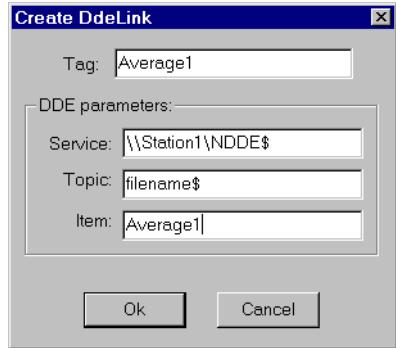
Linking Non-Controllable Objects

If a `LookoutDirect` object is not a Pot, Switch, or Pushbutton, it probably does not have built-in DDE capabilities. However, all you are really trying to do is pass data between two computers. After the data is in `LookoutDirect`, any object can access it, so you can create a `DdeLink` object to receive data from another application using DDE or NetDDE. For more information, refer to the discussion of the `DdeLink` object in the online PDF *Object Reference Manual* and online help. In this example, the other application is `LookoutDirect` running on a second computer.

To Access Real-Time Data at Another Computer

You need at least two computers connected by a functioning network to follow this example. This example refers to the two computers as Station1 and Station2.

1. At Station1, create an Average object. It can average whatever values you want, but give it the name `Average1`.
2. At Station2, create a `DdeLink` object and name it `Average1`. Although you are not required to give objects matching names over a network, it makes your applications easier to maintain.



3. In **Service**, enter \\Station1\NDDE\$, where Station1 is the network name of the other computer. The backslashes, NDDE, and dollar sign are required by Microsoft Windows.
4. In **Topic**, enter Filename\$, where Filename is the name of the LookoutDirect process file on the other computer.
Again, the dollar sign is required but the .14p file extension is not.
5. Enter the name of the object at the other computer that you want to link to in the **Item** field (in this case, it is also called Average1).
6. Click on **OK**.
7. Finish the definition of the DdeLink object by inserting the DDE expression into your panel.

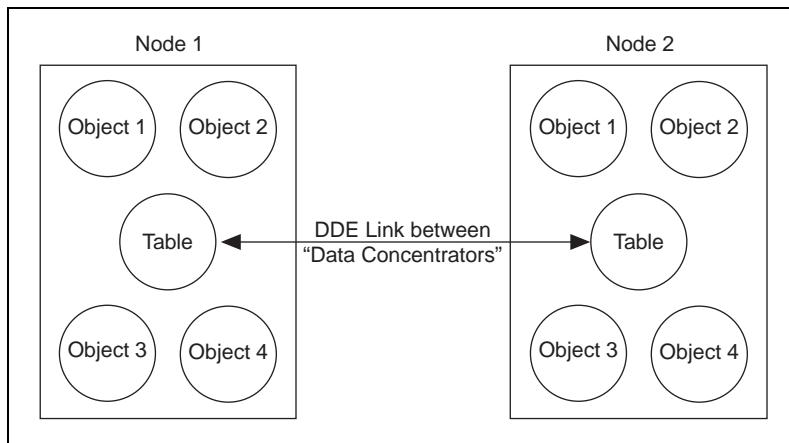
After you establish the link, you can connect other objects at Station2 to the Average1 DdeLink object. Remember, just because you named the DdeLink object Average1 does not make it a LookoutDirect Average object. It just receives data from the Average object at Station1.

Table Networking

Table networking consists of consolidating all the data that needs to be shared over the network in a single table or array of values. You might think of these tables as “data concentrators”—although they can do much more than just serve as a repository for data. For more information, refer to the discussion of the *DataTable* object in the online PDF *Object Reference Manual* or the online help. You can link a table through NetDDE to a corresponding table on another computer. When linked, the tables update each other on all changes within their databases.

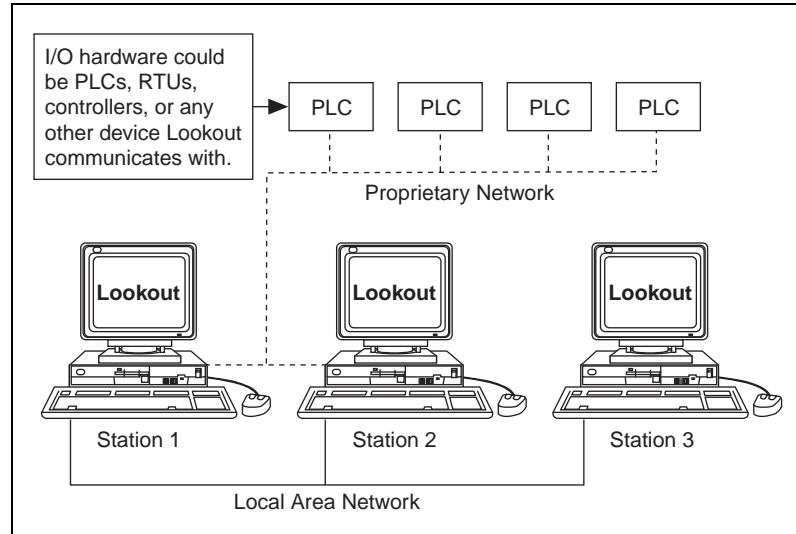


Note Table networking is one of the most involved and advanced concepts in *LookoutDirect*. You should have a complete understanding of objects, database connections, events, and general *LookoutDirect* architecture before you attempt table networking.

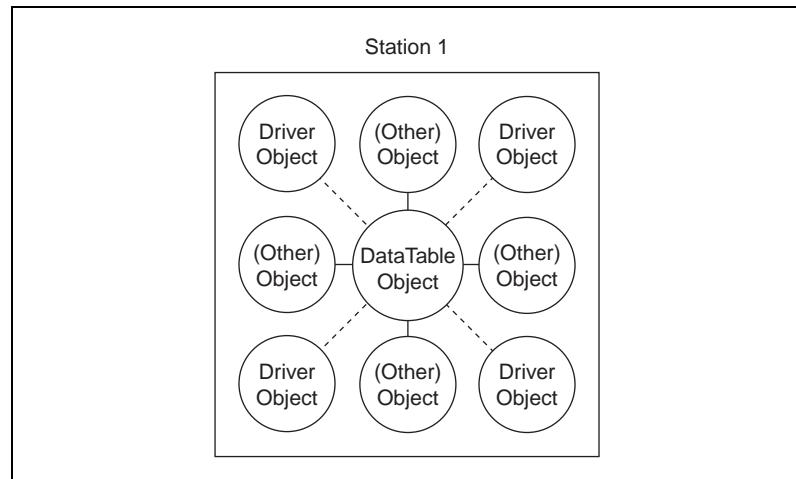


You might have better results linking multiple *DataTable*s for networking purposes if you plan before jumping headfirst into your application. The following example shows how table networking is done.

This example consists of three computers, each running a copy of *LookoutDirect*. Set up three nodes on the network with the names Station1, Station2, and Station3. Station1 is the only node directly connected to a PLC, but you want to monitor and control the I/O from any of the three nodes. The following diagram depicts the network structure of this example.



By doing a little planning, you can configure Lookout*Direct* at Station1 so that you can reuse its process file at Station2 and Station3. The following diagram depicts the general design of the Lookout*Direct* process file at Station1. Notice that Driver objects are connected to the DataTable object using dotted lines while all other objects are connected using solid lines. This is for illustrative purposes only.



The focus of this picture is the Driver objects and their connections to the DataTable. Eventually, you will modify this process file to run at Station2 and

Appendix A

Station3 by deleting all connections to Driver objects, then deleting the Driver objects themselves at Stations 2 and 3.

It is important to route all signals to and from Driver objects through the DataTable. If you want to display an analog input from a PLC on a control panel, do not connect the PLC signal directly to the control panel. Instead, first connect it to the DataTable and then connect the appropriate DataTable cell to the control panel. If you need to connect an operator setpoint to an output on a PLC, do not connect the setpoint directly to the PLC. First connect the setpoint to the DataTable, then connect the appropriate DataTable cell to the output on your PLC. This is only necessary for signals that originate from, or are destined for, a Driver object. Although this procedure adds another step to your LookoutDirect configuration for Station1, it greatly simplifies the development of the process files for Station2 and Station3.

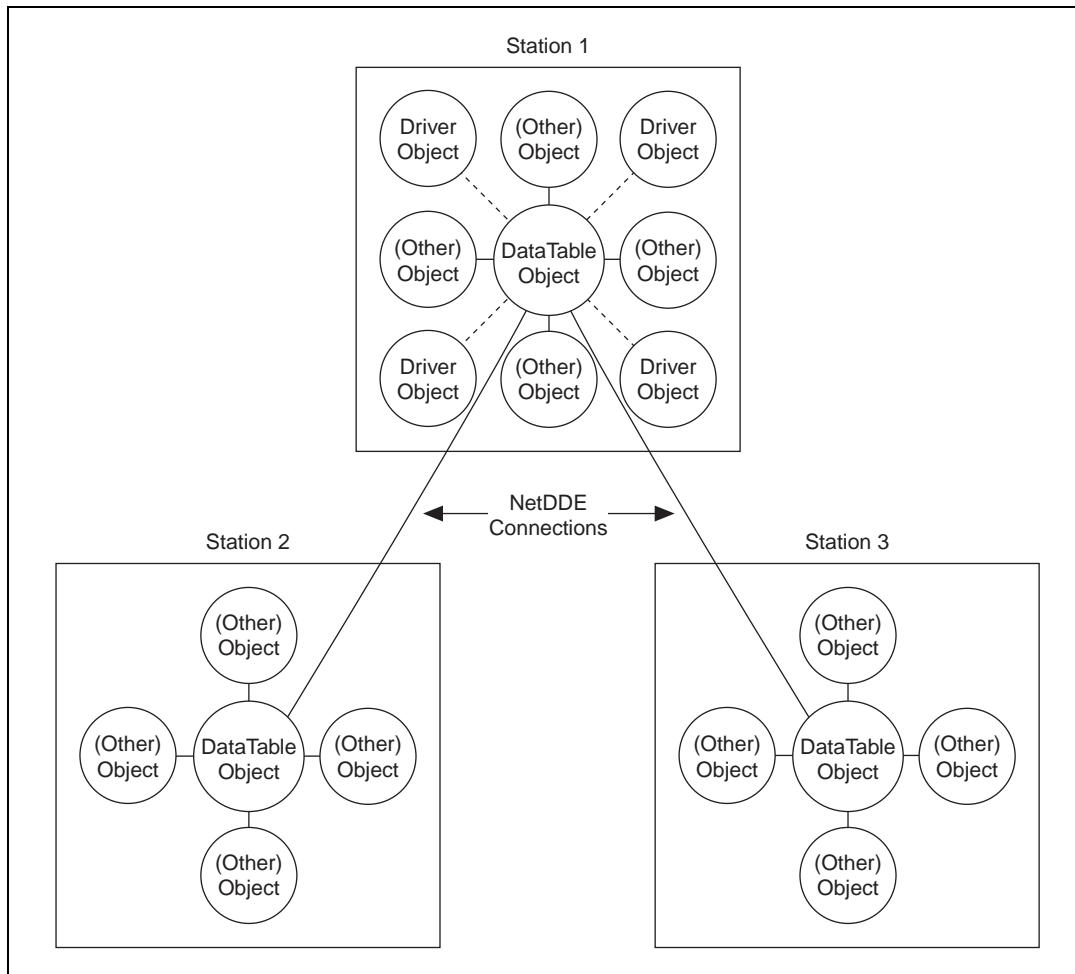
Now that you have developed an application at Station1, move over to Station2. Copy the process file to the Station2 computer and delete all the connections to and from your Driver objects. Then, delete the Driver objects themselves.

Remember, you do not want Station2 attempting to communicate with the PLCs because there is no physical connection to that proprietary network. Station2 should get its information from Station1.

The only thing left to do is modify the DataTable object at Station2. Change its **Table location** parameter to **DDE**. In **Service**, enter \\Station1\\NDDE\$. In **Topic**, enter **Filename\$**, where **Filename** is the name of the process file at Station1. In **Item**, enter the name of the DataTable object at Station1. Your DataTables are now bidirectionally connected through a NetDDE link.

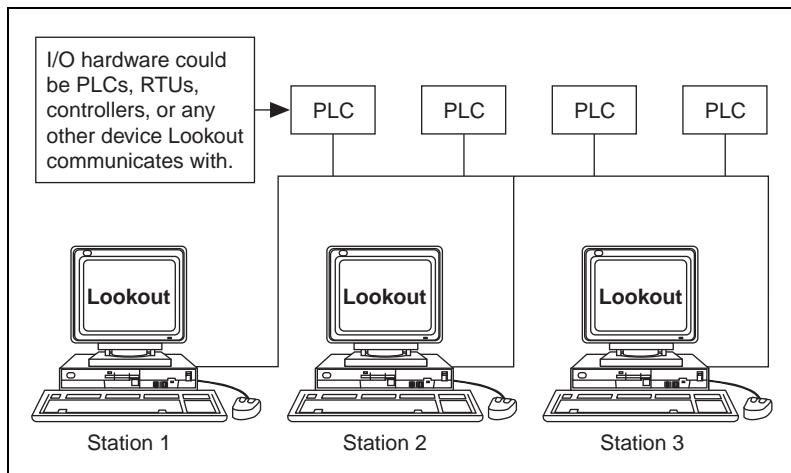
At this stage, you can add more operator stations to the network. Copy the process file from Station2 to Station3, or to any number of other LookoutDirect nodes on the network. No changes to the files themselves are necessary.

Your LookoutDirect network topography should now look something like this.



Hardware Networking

In general, you should try to avoid hardware networking as described here with *LookoutDirect* 4. To implement the hardware networking method, all *LookoutDirect* nodes you want networked must have direct communication access to all your hardware as displayed in the following diagram. Because of this, the hardware networking method is only applicable in certain circumstances.



This is the most straightforward of the three networking methods to implement. Configure a process file at one node and copy it to all other nodes that have direct communication access to your hardware. The only thing to remember when building your process file is to configure all Pots, Switches, and Pushbuttons with the **Remote** parameter setting. This ensures that they stay in sync with the point they are writing to. In other words, if a switch at one *LookoutDirect* node changes the status of an output, all the respective switches at the other *LookoutDirect* nodes will reflect the change and automatically flip to stay in sync. For more information on the **Remote** parameter, refer to the *Pot*, *Switch*, and *Pushbutton* descriptions in the online PDF *LookoutDirect Object Reference Manual* in the online help.

Your hardware, network, and communications topology are the determining factors on whether you can use hardware networking. The hardware networking method is seldom available because it is rare that all your *LookoutDirect* nodes can communicate directly with all your hardware.

Multilink and Table Networking Comparison

There are advantages and disadvantages to both multilink and table networking. Multilink is quick and easy for small jobs. However, every NetDDE link requires processor time, so the more links you have, the more CPU overhead is used. Remember, each value passed between computers requires a separate DdeLink object. Your computer can quickly become bogged down if hundreds of links are constantly sending values back and forth.

On the other hand, while the table method might require more initial planning and development time, there will be only one NetDDE link between computers connecting the DataTables. This method dramatically reduces the amount of DDE traffic on the network. That is why table networking is the method of choice for large systems where computers are sharing hundreds or thousands of values across the network.

So how many points are too many? There is no single answer, but *LookoutDirect* has been used with the table networking method on systems having over 6000 values, and the network update time is well under one second with only three percent CPU usage.

Setting up Networking with DDE

LookoutDirect uses Network DDE (Dynamic Data Exchange) for client/server networking.

Two components of Microsoft Windows are required when using Network DDE: the Network DDE agent (`NETDDE.EXE`) and the NetDDE Share Database Manager (`DSDM`). Exactly how these components are used depends on the operating system on which *LookoutDirect* runs: Windows 98/95 or Windows NT.

The program `NETDDE.EXE` is an agent that runs in the background and transfers DDE messages between computers. This program should be running before an instance of *LookoutDirect* is started if you intend for that session to act as a server. *LookoutDirect* does not automatically run `NETDDE.EXE` itself.

Running NETDDE.EXE Automatically

The following sections show how to make sure that `NETDDE.EXE` runs every time you start Windows.

Windows 98/95

1. In Windows 98/95, Click **Start»Settings»Taskbar**.
2. In the **Taskbar Properties** dialog box, choose the **Start Menu Programs** tab.
3. Click on the **Add** button.
4. In the **Command line** field of the **Create Shortcut** dialog box, enter `NETDDE.EXE`, then click **Next**.
5. Choose the **Start Menu»Programs»StartUp** folder, then click **Next**.
6. Select a name (such as `NetDDE`) for the shortcut; select **Finish** and **OK**.

Windows NT

1. In the Program Manager, double-click on the **Control Panel** icon, or run `CONTROL.EXE` using the Program Manager **File»Run** dialog box. If you are using NT version 4.0, click on **Start**, point to **Settings**, and choose the **Control Panel**.
2. In the **Control Panel** window, double-click on the **Services** icon.
3. From the list of services, select **Network DDE**.
4. Click on the **Startup** button.
5. In the **Startup Type** box, select **Automatic**.

6. Select **OK** and **Close** to exit the **Control Panel**.

Adding a Trusted DDE Share

To allow someone else to connect to a DDE share (for example, `process$`) on your computer when you are logged in, you must *trust* the DDE share. When other users connect to the share remotely, the application they connect to is running in your security context because you are the logged-on user. You must give permission for the other users to access the share. Even another person who is an administrator cannot trust a share for your account.

In Windows 98/95, trusted shares are automatically created by *LookoutDirect*.

In Windows NT, you must use the program `DDESHARE .EXE` to add a trusted share. Exactly how you set this up depends on the level of security you require. The following sequence of steps describes how to set up a share that gives everyone on the network full DDE access to *LookoutDirect*.

1. From the Program Manager, select **File»Run**. If you are using Windows NT version 4.0, click **Start** and select **Run**.
2. Enter `DDESHARE .EXE`.
3. Select **Shares»DDE Shares**.
4. In the **DDE Shares** dialog box, select **Add a Share**.
5. In the **DDE Share Properties** dialog box, do the following:
 - a. For the **Share Name**, enter the name of the *LookoutDirect* process file that you want to share, omitting the `.LKP` extension, and adding a ‘\$’ at the end. For example, if your process file was `PROCESS .LKP`, you would enter
`PROCESS$`
on this line.
 - b. For the **Static Application Name**, enter *LookoutDirect*.
 - c. For the **Static Topic Name**, enter the process file without extension, such as `PROCESS`.
 - d. Leave the **Allow start application** and **is service** boxes unchecked.
 - e. In the **Item Security** box, select **Grant access to all items**.
 - f. Click on the **Permissions** button.
 - g. In the **DDE Share Name Permissions** dialog box, do the following:
 - In the **Names** list, select **EVERYONE**. If there is no entry for **EVERYONE**, use the **Add** button and associated dialog box to create it.

Appendix A

- In the **Type of Access** box, select Full Control.
 - Select **OK**.
- h. Select **OK**.
6. You should now be back in the **DDE Shares** dialog box. Select the share you just created and click on the **Trust Share** button.
 7. In the **Trusted Share Properties** dialog box, select the **Initiate to Application Enable** button, then click on **OK**.
 8. In the **DDE Shares** dialog box, click on **OK**.
 9. Exit the DDE Shares program.



Note For the DDE share to be available, it must be trusted by the user currently logged in to the computer (using steps 6 through 9). Alternatively, the share can be provided as a service by selecting the **Is service** box in the **DDE Share Properties** dialog box. In this case, the share is always available to anyone listed in the **DDE Share Name Permissions** dialog box.



Note When a LookoutDirect client tries to connect to a DDE server on a computer running NT, it might query the user for a user name and password that are valid on the server computer. In this case, the user password should not be NULL, or the attempt to connect to the server will fail.

B

CBL Compiler

LookoutDirect automatically compiles source files when you open the .lks file instead of the .l4p file. There is always the possibility that an automatic compile might fail, so you can also use *LookoutDirect*'s CBL compiler to manually compile a *LookoutDirect* source file from a DOS command line.

If the source file has been corrupted, you can view the error file for compilation errors and either repair the source file or delete corrupted sections and rebuild your process from an intact foundation.

This appendix explains how to use the CBL command-line compiler to compile an .lks file into an .l4p file.

You can re-compile an .lks file in *LookoutDirect* by using **File»Open** and choosing *LookoutDirect* Source Files (*.lks) as the type of file to open.

When you save a process in *LookoutDirect*, *LookoutDirect* creates an .l4p file and an .lks file. The .lks file is a source code file that includes object definitions, names, I/O configuration, communications, control logic, control panel layout, and other parameters. All .lks files are standard ACSII text files that you can print or view in any text editor. The .l4p file is a compiled executable that contains complete configuration information for a particular process control application. An .lks file can be compiled into an .l4p file with the CBL compiler. You can also generate an .lks file by hand or modify an existing .lks file and use the compiler to produce an .l4p file.

CBL is a command-line compiler that takes three arguments: sourcefile, targetfile, and error log file. To use it, type the following at the command line in a DOS window, where *file* is the name of your file:

```
cbl file.lks file.l4p file.err
```

This command compiles the .lks file into the .l4p file and writes any error messages into the error file. If *file.err* is not included in the command line, errors will appear in the DOS window. If the compiler generates error messages, you need to debug your object or your .lks file.

You can directly edit your .lks file with any text editor. In general, if there is no simple repair to be made as indicated by an error message, you can often repair a file by deleting the problem object and recreating it in *LookoutDirect*.

CBL Compiler Error Messages

You can use the .err file to track down compilation errors.

The error file records errors one error per line, with a number at the beginning of each error message. This number corresponds to the line number of your .lks file where this error was encountered. You can go to these line numbers on your .lks file to correct the errors before recompiling your .lks file.

Here is a partial list of the error messages that you might encounter and tips on what might be causing them and how they can be corrected.

Class not found: *class_name*

The .lks file referred to a class that Lookout*Direct* does not know about (for which no corresponding .cbx file exists). It might be that the desired class name was spelled incorrectly, or it might be that the Lookout.dat file needs to be removed so that Lookout*Direct* loads the .cbx file that defines the class. For example, the line

```
    Foo1 = new Foo ();  
will produce the error  
    Class not found: Foo
```

Not a member of *object name*: *member_name*

The name used is not recognized as a valid member name for the object. For example, the line

```
    Pot1.fred = 3;  
produces the error  
    error: Not a member of 'Pot1': 'fred'
```

Member is not writable for class *class name*: *member_name*

The data member is read only. For example, the line

```
    Modbus1.update = 3;  
produces the error  
    error: Member is not writable for class 'Modbus':  
          'update'
```

Name is not valid

The name is not valid. For example, the line

```
    Foo@ = new Pot (0, 100, 1, yes, 0);  
produces the error  
    error: Tag is not valid: Foo@
```

Member is not readable for class *class_name*: *member_name*

The data member is write only. For example, the line

```
Modbus1.l = Pot1.increment;  
produces the error
```

```
error: Member is not readable for class 'Pot':  
'increment'
```

Appendix B

C

Lookout.INI File

Lookout*Direct* configurations are contained in the Lookout.*.INI* file, located in your main Lookout*Direct* directory.

While it is generally best to configure Lookout*Direct* through dialog box options, you can edit this file yourself with any standard text editor. The *.INI* file sections and entries are listed below.

The syntax for an *.INI* file setting is *Key=setting*, where *Key* represents the particular feature or function of Lookout*Direct* you want configured, and *setting* represents the configuration value or choice.

Table C-1 Lookout.INI File Sections, Keys, and Settings

Section	Key	Setting
[System]	NoReloadOfCBXes	=0 CBXs are not reloaded automatically (default) =1 CBXs are reloaded automatically
	CategorizeClasses	=0 Classes are not categorized in the Object Create dialog box =1 Classes are categorized in the Object Create dialog box (default)
	VBuffer	=0 off-screen bitmap buffer inactive =1 off-screen bitmap buffer active (default)
	VBufferSize	= <i>(width*height)</i> of off-screen bitmap buffer (default = 250000)
	SmoothMoves	=0 SmoothMoves off (default) =1 SmoothMoves on

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[System] (continued)	AskBeforeShutdown	=0 Skip "OK to abort current process?" message window on process exit =1 View "OK to abort current process?" message window on process exit (default)
	SuppressRedundantWrites	This setting pertains to remoted Pots. It essentially halts a write (a sending out of a datagram from the Pot) if the last value that the Pot received is the same value it was about to write out =0 Do not suppress redundant writes from Pots (default) =1 Suppress redundant writes from Pots
	MaxWavesQueued	This integer value specifies how many wave files the PlayWave object can queue up (default=10)
	VirginEval	This line is in the Lookout.ini file that comes with an evaluation version of Lookout <i>Direct</i> . If the line is there and equal to a non-zero number, Lookout <i>Direct</i> checks for the following entries VGAeval=vgaeval.1kp SVGAeval=svgaeval.1kp and copies the result into Startup based on the user's screen resolution. If neither of these entries is there, nothing gets written to Startup.
	Startup	Name of the startup process

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[System] (continued)	SaveSource	You can tell LookoutDirect to save or not save the .LKS (source) file =0 Do not save .LKS file =1 Save .LKS file (default)
	AlwaysMaxed NoTaskSwitch NoMenuBar NoCaptionBar LimitPopups	Security levels that limit certain abilities. They correspond to what you see in the System Options dialog box. (default = 0)
	PopupLimit	How many popups can be visible at one time. The security level associated with this feature is LimitPopups. (default = 4)
	LVirtualKeyboard	(default = 0) LVirtualKeyboard determines if the virtual keyboard appears on a left and/or right mouse click.
	RVirtualKeyboard	(default = 0) RVirtualKeyboard determines if the virtual keyboard appears on a left and/or right mouse click.
[Alarms]	Header	=0 Do not show column headers in alarm window =1 Show column headers in alarm window (default)
	Group	=0 Do not show group info in alarm window =1 Show group info in alarm window (default)

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Alarms] (continued)	ShowPriority	=0 Do not show priority in alarm window =1 Show priority in alarm window (default)
	ShowTag	=0 Do not show object name in alarm window =1 Show object name in alarm window (default)
	MostRecent	=0 Display all instances of multiple alarms (default) =1 Display only one instance of multiple alarms
	AllGroups	=0 Show alarms for specified groups only (default) =1 Show alarms for all groups
	Audible	=0 Give no audible notification of alarms =1 Give audible notification (beep) of alarms (default)
	Priority	Filter out alarms displayed in the alarm window that are below this priority. (default = 1)
	OldAlarmsLimit	How many old occurrences of the same alarm to display in the alarm window. (default=0)

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Alarms] (continued)	Style	Display style for the alarm window. (default = 1622) =1620 top =1621 bottom =1622 floating =1623 minimized
	Lines	Number of lines that can be displayed in the alarm window. (default = 3)
	fHeight	Font height of font in the alarm window. (no default)
	fWeight	Font weight of font in the alarm window. (no default)
	fItalic (no default)	=0 text in alarm window NOT italics =1 text in alarm window italics
	LogDevice	Which device to log alarms to. (default="(none)") " (none) " "LPT1:" "LPT2:" "LPT3:" "LPT4:" "COM1:" "COM2:" "COM3:" "COM4:"

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Alarms] (continued)	fFace	Font name of font in the alarm window. (no default)
[Citadel]	DatabasePath	Path to your Citadel database.
	DatabaseMachine	Name of the computer you set for your Citadel database.
[COM1, COM2, COM3, ...]	LineType	Comm port configuration =0 wired (default) =1 dial-up =2 RTS-CTS
	CTSTimeOut	How long (in ms) LookoutDirect waits after asserting RTS for a CTS before sending the frame. (default = 100) (minimum = 0) (maximum = 1000)
	RTSDelayOff	How long (in ms) after LookoutDirect has finished sending the frame LookoutDirect continues to assert RTS. (minimum = 2) (maximum = 2000)
	ReceiveGap	How many empty bytes LookoutDirect must receive before recognizing the end of a frame. (default = 20) (minimum = 0) (maximum = 1000) Lookout 3.8 and later default = 20 Lookout 3.7 and earlier default = 5

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[COM1, COM2, COM3, ...] (continued)	DTR_HangUp	Determines how <i>LookoutDirect</i> hangs up a modem =0 Use +++ATEH to hang up modem =1 Use DTR to hang up modem (default)
	DialSecs	Length of time <i>LookoutDirect</i> waits to receive a connect signal back from the modem it is calling. The time period begins when <i>LookoutDirect</i> first sends the local modem the dialing prefix command. (default = 60) (minimum = 20) (maximum = 1200)
	Retries	The number of times <i>LookoutDirect</i> dials the specified phone number and attempts to connect to the modem at the other end of the line. (default = 3) (minimum = 1) (maximum = 10)
	AlarmPriority	Alarms associated with this comm port will have this alarm priority. (default = 1) (minimum = 1) (maximum = 10)
	HangUpSecs	Length of time <i>LookoutDirect</i> waits after hanging up before it sends the local modem the next dialing prefix signal. (default = 2) (minimum = 1) (maximum = 10)

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[COM1, COM2, COM3, ...] (continued)	DiagnosticFile	<p>Full path to a location where <i>LookoutDirect</i> creates a log file recording all serial transactions on this comm port. If a file already exists at this location, it will be appended to. This file is ASCII text, and is primarily used to diagnose serial port communications problems. (no default)</p> <p>After editing the <i>Lookout.INI</i> file, reload your <i>LookoutDirect</i> process file to force <i>LookoutDirect</i> to reread the <i>.INI</i> file.</p>
	DialPrefix	<p><i>LookoutDirect</i> sends these Hayes AT commands before every use of the modem.</p> <p>(default="ATX4MVEDT")</p>
[Database]	AlarmPriority	<p>Priority of database alarms.</p> <p>(default = 9) (minimum = 1) (maximum = 10)</p>
	MaxLatency	<p>How often (in minutes) database data gets flushed to disk.</p> <p>(default = 10)</p>
[DDE]	FullSendSeconds	<p>How often (in seconds) to update all DDE tables.</p> <p>(minimum = 30)</p>
	DefService	<p>Default service and topic parameters for all DDE connections.</p> <p>(no defaults)</p>

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[DDE] (continued)	DefTopic	Default service and topic parameters for all DDE connections. (no defaults)
[Events]	fWeight	Weight of the font used when printing events. (no default)
	fItalic	Italics of the font used when printing events. (no default)
	fHeight	Height of the font used when printing events. (no default)
	fFace	Name of the font used when printing events. (no default)
	Units	Height (in points) of the font used when printing events. (default = 10)
[Alarm Reports]	fWeight	Weight of the font used when printing alarms. (no default)
	fItalic	Italics of the font used when printing alarms. (no default)
	fHeight	Height of the font used when printing alarms. (no default)

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Alarm Reports] (continued)	fFace	Name of the font used when printing alarms. (no default)
	Units	Height (in points) of the font used when printing alarms. (default = 10)
[Edit]	Position	Where the next created panel element appears relative to the currently selected panel element. =309 right (default) =310 left =311 top =312 bottom =313 center
	Pixels	How many pixels away in the direction selected by Position to create the next panel element. (default = 1) (minimum = 0) (maximum = 200)
	SnapToGrid	=0 off (default) =1 on
	ShowGrid	This only means showing the dots, not doing the actual snapping =0 off =1 on (default)
	GridSize	How far apart (in pixels) are the snap to grid hotspots. (default = 20) (minimum = 2) (maximum = 200)

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Edit] (continued)	GridColor	Color of the snap to grid dots. =5570569 invert the panel color (default) =66 black =16711778 white
[Run commands]	Show00	This setting affects the run command identified by the two digits at the end. This controls how the command is run. =0 no command defined (default) =1 iconic =2 normal =3 maximized
	Desc00	This setting affects the run command identified by the two digits at the end. It defines the description of the command that will be displayed in the Run menu.
	Cmnd00	This setting affects the run command identified by the two digits at the end. This is the DOS-style command itself.
	Levl00	This setting affects the run command identified by the two digits at the end. This is the security level required to run the command. (default = 10)
[Fieldbus]	AlarmPriority	Alarm priority for Fieldbus objects. (default = 0)

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Pots]	SnapDelay	When a remoted Pot's value is changed from a control panel, Lookout <i>Direct</i> allows this amount of time (measured in seconds) to go by and if the remote value does not change to coincide with the newly selected value, the remoted Pot snaps back to the old value. (default = 10) (maximum = 65)
[tagname of a pot]	SnapDelay	Same value as described above, except you can manipulate this on a Pot by Pot basis. (default = [Pots]SnapDelay setting)
[Recipe]	tagname of a recipe.alarmpriority	Alarm priority used by a Recipe object. (default = 8)
[S5-AS511]	Diagnostics	Turns on diagnostic recording functions in S5-AS511 driver. When activated, this option logs diagnostic information to the s5as511.dai file in the Lookout <i>Direct</i> directory. =0 diagnostics are off (default) =1 diagnostics are on
[Sixnet]	IOMAPAlarmPriority	Priority of the "Unable to load Sixnet IOMAP library: iobase32.dll" alarm in Sixnet. (default = 5)
	ProjectAlarmPriority	Priority of the "No Sixnet configuration currently loaded" alarm in Sixnet. (default = 5)

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[LocalTable]	Delay	Periodically post a DDE table with this frequency (measured in ms). (default = 500)
	EchoCursor	=0 no echoes (default) =1 only echo a cursor value if the cell the cursor is pointing to does not have a connection to it
[tagname of a table]	Delay	Same value as described above, except you can manipulate this on a table-by-table basis. (default = [LocalTable]Delay setting)
	EchoCursor	Same value as described above, except you can manipulate this on a table-by-table basis. (default = [LocalTable]EchoCursor setting)
[Tiway]	UpdateOutputs	=0 never update writes (purely event driven) (default) =1 update all writes every 100 polls
[tagname of a Tiway]	UpdateOutputs	Same value as described above, except you can manipulate this on an object-by-object basis. (default = [Tiway]UpdateOutputs setting)
[Graphics]	Directory	Full path to the Lookout <i>Direct</i> graphics directory
[DataPath]	Default	Full path to the Lookout <i>Direct</i> data directory
	tagname of a spreadsheet	Full path name for an individual spreadsheet

Appendix C

Table C-1 Lookout.INI File Sections, Keys, and Settings (Continued)

Section	Key	Setting
[Registration]	SerialNumber	Serial number of your Lookout <i>Direct</i> software
	Key	Key code issued by National Instruments
	Organization Company	These two entries are for the same piece of information, the organization for which the key was issued.
	NIInternalCode	This is how many days Lookout <i>Direct</i> has been running. It is only updated in versions of Lookout <i>Direct</i> where this is important, like the integrator package.
	Name	Name for which the key was issued
	HardwareKey	Hardware key information
[Defaults]	Login	Default login name

Index

A

ABS function, 1-21
accounts
 assigning security levels (note), 6-4
 definition, 6-1
 forgetting your password (note), 6-3
 modifying, 6-4
acknowledging alarms, 9-11 to 9-13
ACOS function, 1-29
action verification, 6-20
Alarm object class
 creating alarm objects, 9-2 to ??
alarms, 9-1 to 9-13
 acknowledging, 9-11 to 9-13
 alarm subsystem, 9-4 to 9-13
 areas, 9-4
 circular reference alarms, 9-3
 database-generated alarms, 9-1 to ??
 DDE, 5-5 to 5-6
 defining alarm conditions, 9-1 to ??
 deselecting, 9-11
 display options, 9-7 to ??
 filters, 9-8
 priorities, 9-6
 selecting, 9-11
Alarms window, 9-6 to 9-7
 color scheme (table), 9-7
 illustration of, 9-6
 viewing alarms, 9-6
aliases
 optional use of (note), 11-5
 purpose and use, 11-4 to 11-5
AND function, 1-18
Animator object class
 displaying dynamic graphics (table),
 2-6 to 2-7
arithmetic operators, 1-15

ASIN function, 1-29

ATAN function, 1-29

ATAN2 function, 1-30

AVG function, 1-24

B

bitmap (.BMP) graphics
 compared with Windows metafiles,
 2-18 to 2-19
 displaying, 2-4 to 2-6
.BMP files. *See* bitmap (.BMP) graphics

C

CBL compiler, B-1
circular reference alarms, 9-3
Citadel Historical Database Logger, 7-5 to 7-9
 accessing data with ODBC driver,
 8-1 to 8-17
 data transforms, 8-6 to 8-7
 SQL examples, 8-7 to 8-8
 traces table, 8-5
 using Microsoft Access, 8-13
 using Microsoft Excel, 8-13
 using Microsoft Query, 8-8 to 8-13
 using Microsoft Visual Basic, 8-17
 creating historical database, 7-7 to 7-8
 data location, 7-5 to 7-6
 information overload, 7-9
 logging criteria, 7-9
client, DDE, 5-3 to 5-4
colors
 alarm status (table), 9-7
 using in graphics, 2-3 to 2-4
comma-separated value (.CSV) files
 for Spreadsheet Logger, 7-3
communications service. *See* serial
 communications.
comparison operators, 1-15 to 1-18
control panels
 report generation, 7-11 to 7-13

- viewing security, 6-10
- controllable objects
 - networking considerations, A-3 to A-4
 - security considerations, 6-9 to 6-10
 - viewing security, 6-10 to 6-11
- conventions, manual, vii to ??
- copying object databases, 11-15
- COS function, 1-30
- .CSV (comma-separated values) files, 7-3
- CTS timeout setting, 3-5
- custom graphics
 - See also* graphics
 - creating, 2-14 to 2-17
 - step-by-step example, 2-14 to 2-17
 - displaying static graphics, 2-4 to 2-6
 - exporting to Lookout, ?? to 2-16
 - exporting to LookoutDirect, 2-16 to ??
 - testing in Lookout, ?? to 2-17
 - testing in LookoutDirect, 2-16 to ??
- D**
- data logging
 - Citadel Historical Database Logger, 7-5 to 7-9
 - report generation, 7-11 to 7-13
 - Spreadsheet Logger, 7-1 to 7-4
- data transforms, 8-6 to 8-7
- dates
 - date/time functions (table), 1-31
- DDE (dynamic data exchange), 5-1 to 5-6
 - alarms, 5-5 to 5-6
 - client, 5-3 to 5-4
 - linking Lookout to other applications, ?? to 5-5
 - linking LookoutDirect to other applications, 5-2 to ??
 - networking considerations, A-12 to A-14
 - adding trusted DDE share, A-13 to A-14
 - running NETDDE.EXE
 - automatically, A-12 to A-13
 - overview, 5-1 to 5-2
- peer-to-peer, 5-4 to 5-5
- server, 5-2 to 5-3
- DialGauge object class
 - displaying dynamic graphics (table), 2-6 to 2-7
- Dialing prefix settings (table), 3-6
- Dial-up serial connection, 3-5 to 3-7
- displaying
 - expressions, 1-8 to ??
- driver objects
 - purpose and use, 3-1 to 3-2
 - types of, 3-1
- dynamic data exchange. *See* DDE (dynamic data exchange)
- dynamic graphics, 2-6 to 2-13
 - See also* graphics
 - displaying
 - logical signals, 2-7 to 2-10
 - numeric signals, 2-10 to 2-11
 - text signals, 2-12 to 2-13
 - tools for displaying (table), 2-6 to 2-7
- E**
- editing database parameters, 11-1 to 11-8
 - logical parameters, 11-7 to 11-8
 - numeric parameters, 11-4 to 11-7
 - text parameters, 11-8
- editing object parameters (example), 11-1 to 11-3
- Event Logger, 7-10
 - See also* logging data and events
 - data location, 7-10
 - information overload, 7-10
- EXACT function, 1-26
- EXP function, 1-22
- exporting object databases, 11-9 to 11-13
 - copying object databases, 11-15
 - creating database spreadsheet, 11-11 to 11-13
 - overview, 11-9
 - procedure, 11-9 to 11-11
- Expression dialog box, 1-12
- expression functions, 1-18 to 1-31

- date/time functions (table), 1-31
- logical functions (table), 1-18 to 1-20
- lookup functions (table), 1-20, 1-21
- mathematical functions (table),
 - 1-21 to 1-24
- statistical functions (table), 1-24 to 1-25
- text functions (table), 1-26 to 1-29
- trigonometric functions (table),
 - 1-29 to 1-30
- E**
 - Expression object class
 - creating Expression objects, 1-10 to 1-11
 - expressions, 1-1 to 1-31
 - as connections, 1-12
 - as parameters, 1-11 to 1-12
 - creating, 1-3 to 1-12
 - displaying on control panels, 1-8 to ??
 - examples, 1-1 to 1-2
 - functions, 1-18 to 1-31
 - illegal conditions, 1-2 to 1-3
 - result of, 1-1
 - syntax, 1-14 to 1-18
 - arithmetic operators, 1-15
 - comparison operators, 1-15 to 1-18
 - text operator, 1-15
 - white space, 1-14
 - tools for displaying dynamic graphics (table), 2-6 to 2-7
 - F**
 - FACT function, 1-22
 - FALSE function, 1-19
 - FIND function, 1-26
 - FIXED function, 1-26
 - G**
 - Gauge object class
 - displaying dynamic graphics (table), 2-6 to 2-7
 - graphic file types
 - bitmaps *vs.* metafiles, 2-18 to 2-19
 - memory considerations, 2-19
 - graphics, 2-1 to 2-19
 - animating. *See* Animator object class
 - creating custom graphics, 2-14 to 2-17
 - dynamic graphics, 2-6 to 2-13
 - file types, 2-18 to 2-19
 - screen resolution considerations (note),
 - 2-1
 - static graphics, 2-2 to 2-6
 - H**
 - hardware networking, A-10
 - Hardwired serial connections, 3-4
 - historical database logging. *See* Citadel Historical Database Logger
 - I**
 - importing object databases
 - copying object databases, 11-15
 - overview, 11-9
 - procedure, 11-13 to 11-15
 - insets
 - displaying in graphics, 2-2 to 2-4
 - INT function, 1-22
 - Is equal to (=) operator, 1-16
 - Is greater than (>) operator, 1-15
 - Is greater than or equal to (>=) operator, 1-15
 - Is less than (<) operator, 1-15
 - Is less than or equal to (<=) operator, 1-15
 - Is not equal to (<>) operator, 1-16
 - L**
 - LCHOOSE function, 1-21
 - LEFT function, 1-26
 - LEN function, 1-27
 - LIF function, 1-19
 - lines, displaying in graphics, 2-2 to 2-4
 - LN function, 1-22
 - LOG function, 1-22
 - LOG10 function, 1-22
 - logging data and events, 7-1 to 7-13
 - Citadel Historical Database Logger, 7-5 to 7-9
 - Event Logger, 7-10
 - report generation, 7-11 to 7-13
 - Spreadsheet Logger, 7-1 to 7-4
 - logical data members

editing parameters, 11-7 to 11-8
logical expressions
 displaying dynamic graphics (table), 2-6 to 2-7
 functions (table), 1-18 to 1-20
logical signals, displaying in graphics, 2-7 to 2-10
L
LOWER function, 1-27
M
mathematical functions (table), 1-21 to 1-24
MAX function, 1-24
menu bars
 viewing security, 6-11
Microsoft Access
 accessing Citadel data, 8-13
 compliance with ODBC (note), 8-4
Microsoft Excel, accessing Citadel data, 8-13
Microsoft Query
 accessing Citadel data, 8-8 to 8-13
 compliance with ODBC (note), 8-4
Microsoft Visual Basic
 accessing Citadel data, 8-17
 compliance with ODBC (note), 8-4
MID function, 1-27
MIN function, 1-24
MOD function, 1-23
modem settings, 3-5 to 3-7
multilink networking, A-3 to A-5
 compared with table networking, A-11
 linking controllable objects, A-3 to A-4
 linking non-controllable objects, A-4 to A-5
Multistate object class
 displaying dynamic graphics (table), 2-6 to 2-7
N
NCHOOSE function, 1-21
networking, 4-1 to 4-7, A-1 to A-14
 See also redundancy
 capabilities, A-1 to A-2
 DDE considerations, A-12 to A-14
adding trusted DDE share, A-13 to A-14
running NETDDE.EXE
 automatically, A-12 to A-13
hardware networking, A-10
methods, A-2
multilink, A-3 to A-5
 compared with table networking, A-11
 linking controllable objects, A-3 to A-4
 linking non-controllable objects, A-4 to A-5
 overview, A-2
 requirements, A-1
 table networking, A-6 to A-9
 compared with multilink networking, A-11
 examples, A-6 to A-8
 routing signals to and from driver objects, A-8
 topography (figure), A-9
networking considerations
 controllable objects, A-3 to A-4
 non-controllable objects, A-4 to A-5
NIF function, 1-19, 1-19
NOT function, 1-19
NOW function, 1-31
numeric data members
 editing parameters, 11-4 to 11-7
numeric expressions, for displaying dynamic graphics (table), 2-6 to 2-7
numeric signals, displaying in graphics, 2-10 to 2-11
O
object databases
 copying, 11-15
 creating database-generated alarms, 9-1 to ??
 editing parameters, 11-1 to 11-8
 exporting, 11-9 to 11-11
 importing, 11-13 to 11-15

- object parameters
 - editing, 11-1 to 11-8
 - example, 11-1 to 11-3
 - expressions as parameters, 1-11 to 1-12
 - logical parameters, 11-7 to 11-8
 - numeric parameters, 11-4 to 11-7
 - text parameters, 11-8
- objects
 - connecting
 - expressions as connections, 1-12
 - networking considerations
 - controllable objects, A-3 to A-4
 - non-controllable objects, A-4 to A-5
- ODBC
 - definition, 8-1
- ODBC driver
 - accessing Citadel data, 8-1 to 8-17
 - data transforms, 8-6 to 8-7
 - ODBC-compliant applications (note), 8-4
 - SQL examples, 8-7 to 8-8
 - traces table, 8-5
 - with Microsoft Access, 8-13
 - with Microsoft Excel, 8-13
 - with Microsoft Query, 8-8 to 8-13
 - with Microsoft Visual Basic, 8-17
- Open Database Connectivity (ODBC). *See*
 - ODBC
- operators
 - arithmetic, 1-15
 - comparison, 1-15 to 1-18
 - text, 1-15
- OR function, 1-19
- P**
- passwords
 - forgetting your password (note), 6-3
 - protecting process files, 6-19 to 6-20
 - revising accounts, 6-4
- Pause between calls (modem) setting, 3-7
- peer-to-peer links, DDE, 5-4 to 5-5
- PI function, 1-23
- Pipe object class
 - displaying dynamic graphics (table), 2-6 to 2-7
- plates
 - displaying in graphics, 2-2 to 2-4
- Poll Rate, 3-2
- Popup control panels
 - viewing security, 6-11 to 6-12
- Pot object class
 - displaying dynamic graphics (table), 2-6 to 2-7
 - displaying dynamic numeric signals, 2-10 to 2-11
- printing
 - alarms, 9-10 to 9-11
 - prioritizing alarms, 9-6
- process files
 - protecting, 6-19 to 6-20
- PRODUCT function, 1-23
- PROPER function, 1-27
- Pushbutton object class
 - action verification, 6-20
 - displaying dynamic graphics (table), 2-6 to 2-7
- R**
- Radio (RTS/CTS) serial connection, 3-4 to 3-5
- RAND function, 1-23
- Receive gap setting, 3-3 to 3-4
- rectangles, displaying in graphics, 2-2 to 2-4
- redundancy, 10-1
 - basic standby principles, 10-2
 - failover scenarios, 10-3
 - standby configuration, 10-4
- REPLACE function, 1-27
- report generation, 7-11 to 7-13
 - control panel reports, 7-11 to 7-13
 - third party reports, 7-13
- REPT function, 1-27
- Retries (modem) setting, 3-6
- RIGHT function, 1-28

- ROUND function, 1-23
RTS delay off time period, 3-5
RTS/CTS handshaking settings, 3-4 to 3-5
- S**
screen resolution
 graphics (note), 2-1
SEARCH function, 1-28
security, 6-1 to 6-20
 action verification, 6-20
 control security, 6-9 to 6-10
 overview, 6-1
 process file security, 6-19 to 6-20
 viewing security, 6-10 to 6-12
 control panels, 6-10
 controllable objects, 6-10 to 6-11
 system settings, 6-11 to 6-12
serial communications, 3-1 to 3-7
 defining serial port settings, 3-2 to 3-7
 dial-up modem settings, 3-5 to 3-7
 driver objects, 3-1 to 3-2
 Hardwired connections, 3-4
 overview, 3-2
 Receive gap setting, 3-3 to 3-4
 RTS/CTS handshaking settings,
 3-4 to 3-5
 selecting serial port, 3-3
Serial port data field, 3-3
server, DDE, 5-2 to 5-3
SIGN function, 1-23
SIN function, 1-30
space, in expressions, 1-14
Spinner object class
 displaying dynamic graphics (table),
 2-6 to 2-7
Spreadsheet Logger, 7-1 to 7-4
 concurrent file access, 7-4
 .CSV files, 7-3
 data location, 7-2 to ??
 file and disk errors, 7-3
 information overload, 7-4
Spreadsheet object class
 exporting object database to spreadsheet
 file, 11-11 to 11-13
- SQL**
 See also ODBC driver
 definition, 8-1
SQRT function, 1-24
standby
 basic principles, 10-2
 configuration, 10-4
 failover scenarios, 10-3
static graphics, 2-2 to 2-6
 custom graphics, 2-4 to 2-6
 displaying text, plates, insets, rectangles,
 and lines, 2-2 to 2-4
statistical functions (table), 1-24 to 1-25
STDEV function, 1-25
STDEVP function, 1-25
Structured Query Language. *See* SQL
SUM function, 1-25
Switch object class
 action verification, 6-20
 displaying
 dynamic graphics (table), 2-6 to 2-7
 logical signals in graphics,
 2-7 to 2-10
 system options, setting for security,
 6-9 to 6-10
- T**
table networking, A-6 to A-9
 compared with multilink networking,
 A-11
 examples, A-6 to A-8
 routing signals to and from driver objects,
 A-8
 topography (figure), A-9
tabs, in expressions, 1-14
TAN function, 1-30
TCHOOSE function, 1-20, 1-21
text
 displaying in graphics, 2-2 to 2-4
text data members
 editing, 11-8

text expressions, for displaying dynamic graphics (table), 2-6 to 2-7
TEXT function, 1-28
text functions (table), 1-26 to 1-29
text operators, 1-15
text signals, displaying in graphics, 2-12 to 2-13
third party reports, 7-13
TIF function, 1-20
time
 date/time functions (table), 1-31
title bars
 viewing security, 6-11
TODAY function, 1-31
traces table, for Citadel data, 8-5
Transparent pixel data fields, 2-5
trigonometric functions (table), 1-29 to 1-30
TRIM function, 1-28
TRUE function, 1-20
TRUNC function, 1-24
trusted DDE share, A-13 to A-14
U
UPPER function, 1-29
V
VAR function, 1-25
VARP function, 1-25
viewing security, 6-10 to 6-12
 control panels, 6-10
 controllable objects, 6-10 to 6-11
 system settings, 6-11 to 6-12
Visual Basic
 accessing Citadel data, 8-17
 compliance with ODBC (note), 8-4
W
Wait for connection (modem) setting, 3-6 to 3-7
white space, in expressions, 1-14
Windows Metafile (.WMF) graphics
 compared with bitmap files, 2-18 to 2-19
 displaying, 2-4 to 2-6