

## COMPREHENSIVE SERVICES

We offer competitive repair and calibration services, as well as easily accessible documentation and free downloadable resources.

## SELL YOUR SURPLUS

We buy new, used, decommissioned, and surplus parts from every NI series. We work out the best solution to suit your individual needs.

 Sell For Cash    Get Credit    Receive a Trade-In Deal

## OBSOLETE NI HARDWARE IN STOCK & READY TO SHIP

We stock **New**, **New Surplus**, **Refurbished**, and **Reconditioned** NI Hardware.



*Bridging the gap between the manufacturer and your legacy test system.*

 1-800-915-6216

 [www.apexwaves.com](http://www.apexwaves.com)

 [sales@apexwaves.com](mailto:sales@apexwaves.com)

All trademarks, brands, and brand names are the property of their respective owners.

**Request a Quote**

 **CLICK HERE**

**SCXI-1303**

## Object Class Addendum

This addendum contains new object classes that are not documented in the *Lookout Reference Manual*. Use the following documentation for these Lookout Object Classes

### Contents

New Object Classes .....	3
Aquatrol Object Class.....	3
RTU Configuration Dialog Box .....	5
Aquatrol Data Members .....	5
Status Messages .....	7
ASCII .....	8
Status Messages .....	15
Mitsubishi, MitsubishiFX .....	17
Status Messages .....	20
NIDAQDevice.....	22
NIDAQ.INI .....	23
Error Messages.....	24
NISCXI.....	26
NIDAQ.INI .....	28
Error Messages.....	29
Devices.....	30

---

Lookout™ is a trademark of National Instruments Corporation. Product and company names are trademarks or trade names of their respective companies.

Omron.....	31
Status Messages .....	33
Philips.....	35
Status Messages .....	38
Profibus DP Object Class .....	38
Configuring the Profibus DP network.....	38
Requirements .....	39
Profibus DP Data Members.....	40
Status Messages .....	41
ProfibusL2 Object Class.....	42
Lookout Messaging System .....	42
Sample Program.....	43
Detailed Explanation of the Profibus Example Program .....	44
Requirements .....	45
Profibus Data Members.....	48
Status Messages .....	49
Reliance Object Class.....	50
PC-Link Card Settings .....	51
Destination Settings .....	52
Reliance Data Members .....	52
Status Messages .....	53
S5_3964 Object Class.....	55
S5_3964 Parameters.....	55
S5_3964 Data Members .....	56
Alarms .....	59
SqlExec.....	60
Comments .....	62
Buffering.....	64
Status Messages .....	65
Toshiba Mseries/Toshiba Tseries .....	67
Status Messages .....	72

# New Object Classes

## Aquatrol Object Class

Aquatrol is a protocol driver class Lookout uses to communicate with Aquatrol W1500 controllers.

**Create Aquatrol Secondary**

Tagname:

Address:  Model:

Communication Settings

Serial port:  Data bits:

Baud Rate:  Stop bits:

Parity:  Phone:

PollRate =

Poll =

Communication alarm priority:

Retry attempts:  Receive timeout:  msecs

Skip every  poll requests after comm failure

**Address** is the address of the RTU as configured on the device. The address can be in the range of 100 to 999 inclusive.

**Model** is W1500 only at this time.

**Serial port** specifies which port the object uses for communication to the external device. This does not specify the communication type. Communication type is determined by the Options→Serial Ports... command.

**Data rate**, **Parity**, **Data bits**, and **Stop bits** reference the settings on the hardware device.

**Phone** specifies the number to be dialed if the serial port setting is configured for dial-up. This number only applies to the individual protocol object.

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. Aquatrol then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). *See Numeric Data Members in Chapter 5 for information on entering time constants.*

**Poll** is a logical expression that, when transitioned from false to true, causes Lookout to poll the device. This could be a simple expression like the signal from a pushbutton, or a complex algorithm.

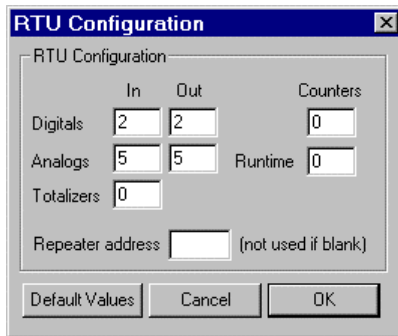
**Communication alarm priority** determines the priority level of object-generated alarms (0-10).

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device if it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Aquatrol object generates an alarm and releases the communication port back to the communications subsystem (COMSUB) which then moves on to the next device in the polling queue (if any).

**Receive timeout** is the time delay Lookout uses in waiting for a response from a device before retrying the request.

The **Skip every...** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout skips the device in the polling cycle accordingly. Once communications have been reestablished, the device will be polled on its regular cycle.

## RTU Configuration Dialog Box



The image shows a Windows-style dialog box titled "RTU Configuration". It contains a table for configuring I/O and counters. The table has columns for "In", "Out", and "Counters". The rows are "Digitals", "Analog", "Totalizers", and "Repeater address". The "Repeater address" row has a text input field and a note "(not used if blank)". At the bottom are three buttons: "Default Values", "Cancel", and "OK".

	In	Out	Counters
Digitals	2	2	0
Analog	5	5	Runtime 0
Totalizers	0		
Repeater address	<input type="text"/> (not used if blank)		

Default Values Cancel OK

You must configure Lookout to match the configuration in your Aquatrol device.

**Digital** (In/Out) specifies configuration for the number of discrete I/O.

**Analog** (In/Out) specifies configuration for the number of analog I/O.

**Totalizers** specifies configuration for the number of totalizers.

**Start** specifies configuration for the number of start counters.

**Runtime** specifies configuration for the number of runtime counters.

**Repeater Address** is the final destination address if a repeater is being used.

**Note**     **The Aquatrol must be configured with all I/O grouped together by kind. Digital inputs must be the first physical inputs on the device, followed by digital outputs. Analog inputs are next, followed by analog outputs. If you do not configure the I/O this way, you can receive invalid data.**

## Aquatrol Data Members

All readable and writable members (inputs/outputs), polling instructions, read/write blocking, serial port usage, and so on, are bundled with the object. Therefore, as soon as you create an Aquatrol object you immediately have access to all the object data members (see data member list below).

**Note** Lookout protocol driver objects automatically generate an efficient read/write blocking scheme based on the inputs and outputs being used in your process file. You are not required to build your own I/O blocking table.

*Aquatrol data members*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
AI1-AI48	numeric	yes	no	Analog Input, 16 bit.
AO1-AO48	numeric	no	yes	Analog Output, 16 bit.
ColdRestart	logical	yes	no	True if the device power has cycled on/off or the reset button has been pushed.
CommFail	logical	yes	no	True if serial communications have failed.
ConfigError	logical	yes	no	True if the configuration of the device does not match the configuration set in the configuration dialog of the Aquatrol device.
DataFail	logical	yes	no	True if the device has had a data failure.
DI1-DI48	logical	yes	no	Discrete Input, 1 bit.
DO1-DO48	logical	no	yes	Discrete Output, 1 bit.
EEPROMerror	logical	yes	no	True if the device has an EEPROM error.
LowBattery	logical	yes	no	True if the device has a low battery.
PowerFail	logical	yes	no	True if there is no power to the device.
R1-R48	numeric	yes	no	Runtimes, 16 bit.
Poll	logical	no	yes	True initiates a Poll sequence.
PollRate	numeric	no	yes	Time interval of Polling sequence.
S1-S48	numeric	yes	no	Starts, 16 bit.
T1-T48	numeric	yes	no	Totalizers, 16 bit.
Update	logical	yes	no	False during Polling sequence.
WarmRestart	logical	yes	no	True if the device has been reset locally.

## Status Messages

### No response within timeout period

No response within timeout period for repeated message. Lookout did not received the expected response within the **Receive timeout** period. The object sent an inquiry and received an acknowledgment, but the device did not send an expected response to the request. This might happen if the response was interrupted. You may have to increase **Receive timeout**.

### Frame Error (garbled): Invalid destination address

Lookout has received a frame with an invalid return address. Make sure that no duplicate addresses exist on the Aquatrol network.

### Frame Error (garbled): Invalid source address

Lookout has received a frame from a device that you are not actively polling. Make sure that there is only one outstanding master request at a time.

### Frame Error (garbled): Invalid message length for configuration

Lookout has received a frame whose length conflicts with the length expected based on the configuration settings in the Aquatrol device configuration dialog box.

### Frame Error (garbled): Invalid CRC

Lookout has received a frame with an invalid CRC (cyclic redundancy check). Check for signal noise on Aquatrol network.



**Invalid discrete address #: Check configuration settings**

**Invalid analog address #: Check configuration settings**

**Invalid runtime address #: Check configuration settings**

**Invalid start address #: Check configuration settings**

**Invalid totalizer address #: Check configuration settings**

All the above errors mean that a I/O point being either read or written is out of range with respect to the configuration set in the Aquatrol device configuration dialog box.

## ASCII

ASCII is a protocol driver class Lookout uses to communicate with any serial device that accepts ASCII characters.

An ASCII object contains no predefined data points. When you create an ASCII object, you must define your data request strings as well as the template Lookout uses to parse the response frame.

**Create ASCII Secondary**

Tagname: ASCII1

Communication Settings:

Serial port: COM1 Data bits: 8

Baud Rate: 9600 Stop bits: 1

Parity: None

Phone number:

Monitor serial port ☐

Communication alarm priority: 8

Retry attempts: 4 Receive timeout: 250 msec

Skip every 5 poll requests after comm failure

Ok Cancel

**Serial port** specifies which COM port the object uses for communicating to the external device. This does not specify the communication type. Communication type is determined by the Options→Serial Ports... command.

**Baud rate** indicates the rate that Lookout uses to communicate with the hardware device.

**Data bits** indicates the number of data bits that Lookout uses to communicate with the hardware setting should match the selection made on the physical device. This setting should match the selection made on the physical device.

**Stop bits** indicates the number of stop bits that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Parity** indicates the parity that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Phone number** specifies the number to be dialed if the selected serial port is configured for dial-up. This number only applies to the individual protocol object.

**Monitor Serial Port** specifies whether you will be receiving unsolicited frames.

**Communication alarm priority** determines the priority level of alarms generated by the ASCII object. Such alarms are typically related to communications with the physical device.

**Retry attempts** specifies the number of times Lookout attempts to establish communications with a device if it is not getting a valid response. After it tries the number of **Retry attempts** specified, the ASCII object generates an alarm and releases the communication port back to the communications subsystem (COMSUB) which then moves on to the next device in the polling queue (if any). Refer to Chapter 6, *Serial Communications*, in the *Lookout Reference Manual* for more information.

**Receive timeout** is the amount of time Lookout waits for a response from a device before retrying the request.

The **Skip every \_\_\_\_ poll requests after comm failure** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout polls the device only once in the specified number of poll cycles. Once communication has been reestablished, the device is polled on its regular cycle.

### ASCII data members

Data member	Type	Read	Write	Description
RequestFormat	text	no	yes	Format used to create request frame.
ResponseFormat	text	no	yes	Format used to parse response frame.
Send	logical	no	yes	Sends request frame.
RQV1, RQV100	numeric	no	yes	Variable list used to populate request frame with numeric values.
RQV1.txt, RQV100.txt	text	no	yes	Variable list used to populate request frame with text values.
RQV1.logical, RQV100.logical	logical	no	yes	Variable list used to populate request frame with logical values.
RSV1, RSV100	numeric	yes	no	Variable list used to store values retrieved from response frame.
RSV1.txt, RSV100.txt	text	yes	no	Variable list used to store values retrieved from response frame.
RSV1.logical, RSV100.logical	logical	yes	no	Variable list used to store values retrieved from response frame.
OffHook	logical	no	yes	Keeps the driver from releasing the serial port.
Request	text	yes	no	Exact request frame sent.
Response	text	yes	no	Exact response frame received.
CommFail	logical	yes	no	Object-generated signal that is on if Lookout cannot communicate with the device(s).
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.

\* RQVn, RQVn.txt and RQVn.logical all represent the same value in different forms.

\* RSVn, RSVn.txt and RSVn.logical all represent the same value in different forms.

## Request and Response Format Strings

The request and response format strings consist of static characters and markers that control how the request and response frames respectively are formatted or decoded. The request format string is used to **create** the request frame, which is sent to the device, while the response format string is used to **decode** the response frame, which comes from the device.

Static characters in the format strings are reproduced exactly in the request or response frame. Markers specify the location within the frame and type of data which should be found there, such as five characters read as an unsigned integer, for example. The ASCII object constructs a request frame by processing the sequence of static characters and markers in the request format string, and including data from RQV data members.

The response format string decodes a response frame using an analogous process, storing the results in RSV data members.

To construct a request frame, the ASCII object parses the request format string character by character. Static characters are copied directly to the request frame. When a marker is encountered the ASCII object reads a value from the appropriate RQV variable and places it into the request frame.

There are 100 RQV and RSV values provided for in the ASCII object data member collection. The first marker in a format string uses the value from RQV1 (or RQV1.txt or RQV1.logical), the next marker uses the value RQV2, and so on. Values taken from Response strings are stored in RSV data members in the same way.

Keep in mind that writing into RQV1 changes the value both for RQV1.text and RQV1.logical. Their only difference is the format in which they are represented. The same principle applies to the RSV data members.

**Note**     **There is no precedence to the order in which multiple objects connected to the same variable number will initialize upon opening the process file, such as the case in which a Pot object is connected to RQV1 while a TextEntry object is connected to RQV1.txt. You should take care to initialize such variables to the proper value after opening a process file.**

To decode a response frame, the ASCII object compares the response frame to the response format string character by character. The static characters in the response frame must match those in the response format string or the decoding process terminates. Static characters are, in effect,

discarded by the ASCII object as they are matched between the response format string and the response frame.

When the ASCII object encounters a marker, it places the data indicated by the marker into the appropriate RSV data member.

The conversion of a portion of the response frame to a data type specified by a marker in the response format string must be valid, or the process will terminate.

If nothing halts the process, decoding terminates when the end of the response frame string is reached.

There are examples of both request frames and response frames at the end of this section, but for the examples to make sense, you must first understand the ASCII object markers.

### Markers

The general format for a marker is:

%[width] [type]

Each field in the marker format is a single character or a number signifying a particular format option.

The **%** sign denotes the beginning of the marker. If the percent sign is followed by a character that has no meaning as a format-control character, that character and the following characters (up to the next percent sign) are treated as static characters, that is, a sequence of characters that must match the frame exactly. For example, to specify that a percent-sign character is a static character part of the frame, use **%%**.

**Width** is a positive decimal integer specifying the number of characters that particular value occupies in the frame. By default ASCII will pad the value with blank spaces if the value takes up fewer characters than the value specified by width. Including a 0 before the width value forces the ASCII object to pad with zeroes instead of blank spaces.

**Type** determines whether the field is interpreted as a character, a string, or a number.

### *Data types allowed by ASCII*

Character	Data Type
d	Decimal integer
x, X	Hexadecimal integer
u	Unsigned decimal integer
f	Floating-point
s	String
b	Byte (binary)

The simplest format specification contains only the percent sign and a type character (for example, %s). That would place the value in the response frame in the RSV1.txt data member.

Request format string	RQV1	Request frame
>%5d	34	> 34
>%05d	34	>00034

The request format string also has a precision value in the form **%[width].[precision][type]**. This specifies the number of digits to the right of the decimal point, if any, in the request frame. If you use a float (%f) and do not specify a precision value, the ASCII object assumes a default of 6.

Characters are converted and stored in RSV data members from response frames in the order they are encountered in the response format. However, fewer than **[width]** characters may be read if a white-space character (space, tab, or newline) or a character that cannot be converted according to the given format occurs before **[width]** is reached.

Values needed for request frames come from the RQV data members, and are also used in the order in which they occur in the request format.

To read strings not delimited by space characters, or that contain spaces, you can substitute a set of characters in brackets (**[ ]**) s (string) type character. The corresponding input field is read up to the first character that does not appear in the bracketed character set. Using a caret (^) as the first

character in the set reverses this effect: the ASCII object reads input field up to the first character that does appear in the rest of the character set.

<b>Response format string</b>	<b>RSV1.txt</b>	<b>Response frame</b>
-------------------------------	-----------------	-----------------------

\$[A-Z, a-z, ]\$	Natl Inst	\$Natl Inst\$
>[^, s]	days	>day

Notice that %[a-z] and %[z-a] are interpreted as equivalent to %**[abcde...z]**, and that the character set is case sensitive.

**Note The brackets only work in response format strings. They have no effect in the request format string.**

The ASCII object scans each field in the response frame character by character. It may stop reading a particular field before it reaches a character for a variety of reasons:

- The specified width has been reached.
- The next character cannot be converted as specified.
- The next character conflicts with a character in the response format string that it is supposed to match.
- The next character fails to appear in a given character set.

No matter what the reason, when the ASCII object stops reading an field, the next field is considered to begin at the first unread character. The conflicting character, if there is one, is considered unread and is the first character of the next field.

### Entering Format String

For a static connection to one of the format data members, enter your format string in the yellow field box in the Edit Connections dialog box. Remember to begin and end the format strings with quotation marks so that Lookout will accept the string input.

You can also connect any valid text data member, such as a text entry object, to the format data members.

# Status Messages

The following alarm messages only apply to a polling cycle initiated by the ASCII object with a request frame where a response format is non-null. When monitoring a serial port, no alarm messages are generated.

## No response within timeout period

Lookout received no response from a device within the **Receive timeout** period. The ASCII object can use the COM port, but when it polls the device, it doesn't respond—as if it is not even there. Verify your baud rate settings, cable connections, power, configuration settings, COM port settings, and polling request frame.

## Data type or length does not match format string

This means that the conversion directive for a portion of the response frame could not produce the desired data type.

## Frame Error (garbled): Response frame does not match response format

This means that one or more of the static characters in the response format did not match the response frame.

### Request Frame Construction Examples:

Request format string	RQV	Request frame
<01%4u%s	RQV1=1234	<011234Steph
	RQV2.txt=Steph	
<01%04u%s	RQV1=34	<010034Steph
	RQV2.txt=Steph	
<01% 4u%s	RQV1=34	<01 34Steph
	RQV2.txt=Steph	

A zero in front of the four pads with zeroes; a space pads with spaces.



### Response format Examples:

Response frame	Response format string	RSV
* ([16.38:	* (%5.2f:	RSV1=16.38

**Note** The decimal point counts as a character when decoding floats (%f). Also, decimal points denoting precision are not allowed when decoding a float in the response frame.

>>Test Text<<	>>%s<<	RSV1.txt=Test
---------------	--------	---------------

The space between the words terminates the conversion. See the bracketed character example above in order to span a space or other special characters .

>>Test Text<<	>>%s%s<<	RSV1.txt=Test
		RSV2.txt=Text

>>DogCat<<	>>%3s%3s<<	RSV1.txt=Dog
		RSV2.txt=Cat

The response format uses a space as a delimiter.

## Mitsubishi, MitsubishiFX

Mitsubishi is a protocol driver class Lookout uses to communicate with Mitsubishi A0J2, A0J2H, A1, A1S, A1N, A1E, A2, A2n, A2C, A2E, A3, A3N, A3E, A3H, and A3M devices using the serial communication protocol.

A Mitsubishi object contains a great deal of data. It supports reading and writing of all predefined data points. When you create a Mitsubishi object, you have immediate access to all the data members for that object (*see data member list below*).

**Note** This object needs Lookout version 3.6 build 15 or greater.

The screenshot shows a Windows-style dialog box titled "Create A Series Secondary". It contains the following fields and controls:

- Tag:** Text box containing "Mitsubishi1".
- Address:** Text box containing "255".
- PLC Model:** Dropdown menu showing "A0J2".
- Communication Settings:** A group box containing:
  - Serial port:** Dropdown menu showing "COM1".
  - Data bits:** Dropdown menu showing "8".
  - Baud Rate:** Dropdown menu showing "9600".
  - Stop bits:** Dropdown menu showing "1".
  - Parity:** Dropdown menu showing "Odd".
  - Phone number:** Empty text box.
- PollRate =** Text box containing "0.01".
- Poll =** Empty text box.
- Communication alarm priority:** Text box containing "8".
- Retry attempts:** Text box containing "4".
- Receive timeout:** Text box containing "250" followed by "msecs".
- Skip every** Text box containing "5" followed by "poll requests after comm failure".
- Buttons:** "Ok" and "Cancel" buttons.

**Serial port** specifies which comm port the object uses for communicating to the external device. This does not specify the communication type. Communication type is determined by the Options→Serial Ports... command.

**Data rate** indicates the baud rate that Lookout uses to communicate with the hardware device. This **Data rate** setting should match the selection made on the physical device.

**Data bits** indicates the number of data bits that Lookout uses to communicate with the hardware device. This **Data bits** setting should match the selection made on the physical device.

**Stop bits** indicates the number of stop bits that Lookout uses to communicate with the hardware device. This **Stop bits** setting should match the selection made on the physical device.

**Parity** indicates the parity that Lookout uses to communicate with the hardware device. This **Parity** setting should match the selection made on the physical device.

**Phone number** specifies the number to be dialed if the selected serial port is configured for dial-up. This **Phone number** only applies to the individual protocol object.

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. The object then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). See the *Numeric Data Members* in Chapter 5, *Using Objects*, of your Lookout Reference Manual for more information on entering time constants.

**Poll** is a logical expression. When transitioned from FALSE to TRUE, it causes Lookout to poll the device. A poll could be a simple expression, such as the signal from a pushbutton, or it could be a complex algorithm.

**Communication alarm priority** determines the priority level of alarms generated by the Mitsubishi object. Such alarms are typically related to communications with the physical device.

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device when it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Mitsubishi object generates an alarm and releases the communication port back to the communications subsystem (COMSUB). The subsystem then moves on to the next device in the polling queue (if any). See Chapter 6, *Serial Communication*, in your Lookout Reference Manual for more information.

**Receive timeout** is the time delay Lookout uses in waiting for a response from a device before retrying the request.

The **Skip every...** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout skips the device in the polling cycle. Once communications have been reestablished, the device is polled on its regular cycle.

As with all Lookout drivers, you can access I/O points and other data through data members. The following is a table of data members currently supported by the Mitsubishi object class.

*Mitsubishi data members (A series)*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
AR0, AR27	numeric	yes	yes	Auxiliary relay area, read as 16-bit word.
AR0.0, AR27.15	logical	yes	yes	Auxiliary relay area, read as 1-bit discrete.
C0, C1023	numeric	yes	yes	Counter, 16-bit word.
CommFail	logical	yes	no	Object-generated signal, on if Lookout cannot communicate with the device(s).
D0, D1023	numeric	yes	yes	Data register, 16-bit word.
DM0, DM9999	numeric	yes	yes	Data memory area, 16-bit word.
HR0, HR99	numeric	yes	yes	Holding relay area, read as 16-bit word.
HR0.0, HR99.15	logical	yes	yes	Holding relay area, read as 1-bit discrete.
IR0, IR511	numeric	yes	yes	I/O area, read as 16-bit word.
IR0.0, IR511.15	logical	yes	yes	I/O area, read as 1-bit discrete.
LR0, LR63	numeric	yes	yes	Link relay area, read as 16-bit word of information.
LR0.0, LR63.15	logical	yes	yes	Link relay area, read as 1-bit discrete.
M0, M2047	logical	yes	yes	Discrete coil, 1-bit.
Poll	logical	no	yes	Lookout expression that when transitioned from false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device polling frequency.
T0, T1023	numeric	yes	yes	Timer, 16-bit word.
TC0, TC999	numeric	yes	no	Timer/Counter, read as 16-bit word.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.
W0-W1023	numeric	yes	yes	Data register, 16-bit word.
X0, X2047	logical	yes	no	Discrete input, 1-bit.

### *MitsubishiFX data members (FX series)*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
Y0, Y2047	logical	yes	yes	Discrete output, 1-bit.
D0, D1023	numeric	yes	yes	Data register, 16-bit word.
T0, T1023	numeric	yes	yes	Timer, 16-bit word.
X0, X1023	logical	yes	no	Discrete input, 1-bit.
CommFail	logical	yes	no	Object-generated signal that is on if Lookout cannot communicate with the device(s).
Y0, Y1023	logical	yes	yes	Discrete output, 1-bit.
M0, M1023	logical	yes	yes	Discrete coil, 1-bit.
sM0, sM1023	logical	yes	yes	Special discrete coil, 1-bit.
S0, S1023	logical	yes	yes	State, 1-bit.
C0, C1023	numeric	yes	yes	Counter, 16-bit word.
Poll	logical	no	yes	Lookout expression that when transitioned from false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device polling frequency.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.

## **Status Messages**

### **No response within timeout period**

Lookout received no response from a device within the **Receive timeout** period. The Mitsubishi object is able to use the comm port, but when it polls the device, it does not respond—as if it is not even there. If you have daisy-chained several devices, you have introduced an inherent delay. You may need to significantly increase **Receive timeout** (and **PollRate**) to ensure Lookout is allowing enough time to receive the expected response. This increase has nothing to do with the processing capabilities of Lookout, but is based solely on **Data rate** and the number of devices on

the chain. Also, verify your Baud rate settings, cable connections, power, configuration settings, comm port settings, and polling addresses.

### **Incorrect frame check sum (FCS)**

The frame was received with an invalid frame check sum. Check cabling for two or more devices with the same address.

### **Incorrect PC number in response**

The frame received had an incorrect source address. Check for two or more devices with the same address.

### **Incorrect command in response**

The frame received had an incorrect command. Check for two or more devices with the same address.

### **Garbled or Invalid frame**

The frame was received without proper termination character (ETX). Check the Lookout receive gap setting.

### **No acknowledgment for write frame**

The write frame was not acknowledged by the PC. Check the address of Mitsubishi object or the address range of the PC.

### **Mitsubishi errors reported in the response**

These errors are reported by the Mitsubishi device, and are in turn reported to the user in text form.

### **Mitsubishi models supported**

A series:        A0J2, A0J2H, A1, A1S, A1N, A1E, A2, A2n, A2C, A2E, A3, A3N, A3E, A3H, A3M, Fxon

FX series:       All models through March, 1997.

## NIDAQDevice

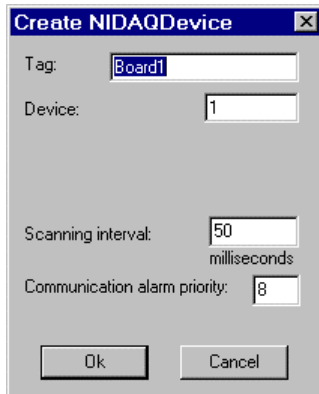
Lookout uses the NIDAQDevice object class to communicate with National Instruments data acquisition devices including data acquisition devices connected in parallel mode to SCXI hardware. Refer to the NISCXI object class for more information concerning configurations using SCXI hardware connected in multiplexed mode.

To use this object, you should have NI-DAQ 5.0 or better software installed. While the NIDAQDevice object will work with some earlier versions of NI-DAQ software, it does not perform well with these earlier versions, and has not been extensively tested.

Consult your NI-DAQ hardware and software manuals for information on installing and configuring National Instruments data acquisition software and hardware.

When you create an NIDAQDevice object for the first time, or when you launch Lookout and run an application using the NIDAQDevice object, Lookout automatically loads the NI-DAQ software. This can take a few moments, during which the screen is frozen. Once the NI-DAQ software has loaded, Lookout returns to its former speed.

**Note** This object class is available on 32-bit versions of Lookout 3.7 and later. It is not backward compatible with earlier versions of Lookout, and does not run on the 16-bit version of Lookout.



**Device** is the NI-DAQ device number. The DAQ configuration utility (WDAQConf.exe) assigns this number to an installed device. Valid device numbers range is from 1 to 16.

**Scanning interval** is the time period between analog and digital input polls. The valid range is 20 msec to 1 day(expressed in msec). You should also keep in mind that some National Instruments DAQ cards need time to stabilize, and so that a scanning rate that is faster than the stabilization rate of the card will return suspect values.

**Communication alarm priority** determines the priority level of the alarms generated by the NIDAQDevice object.

#### *NIDAQDevice Object Class Data Members*

Data member	Type	Read	Write	Description
AI0, AI63	numeric	yes	no	Analog input channel in volts.
AO0, AO63	numeric	no	yes	Analog output channel in volts.
CommFail	logical	yes	no	Object-generated signal that is on if, for whatever reason, Lookout cannot communicate with or control the device without error.
DI0, DI31	logical	yes	no	Digital input line.
DO0, DO31	logical	no	yes	Digital output line.
Update	logical	yes	no	Object-generated signal that pulses each time the object polls the device.

## **NIDAQ.INI**

You may configure individual channel attributes by editing the NIDAQ.INI file in your Lookout directory.

To specific a channel, use the following format: [DEV1.AI5]

This specifies analog input channel 5 on device number 1.

**UPPERLIMIT** specifies the upper input limit in volts (for example, UPPERLIMIT=2.5). Usually used in conjunction with **LOWERLIMIT**.

**LOWERLIMIT** specifies the lower input limit in volts (for example, LOWERLIMIT=-3.7). Usually used in conjunction with **UPPERLIMIT**.



**INPUTMODE** specifies the analog input connection mode to use for this channel. Valid choices are 1 for differential, 2 for referenced single-ended, and 3 for non-referenced single-ended.

## Error Messages

### Error loading NI-DAQ driver

Lookout was not able to communicate to the NI-DAQ driver. Be sure that NI-DAQ has been installed properly, and the NI-DAQ Configuration Utility has been used to configure the your hardware devices.

**NIDAQ: Analog Input: Invalid Data Member(s)**

**NIDAQ: Analog Output: Invalid Data Member(s)**

**NIDAQ: Digital Input: Invalid Data Member(s)**

**NIDAQ: Digital Output: Invalid Data Member(s)**

At least one channels you have specified is not valid for your current hardware configuration. Be sure your hardware is configured properly using the NI-DAQ Configuration Utility.

**NIDAQ: Error code: NNNNN**

**NIDAQ: Analog Input: Error code: NNNNN**

**NIDAQ: Analog Output: Error code: NNNNN**

**NIDAQ: Digital Input: Error code: NNNNN**

**NIDAQ: Digital Output: Error code: NNNNN**

NI-DAQ has detected an error condition. Please refer to your *NI-DAQ Function Reference Manual* to determine the meaning of the error code.

*National Instruments data acquisition devices supported by Lookout*

<b>Device</b>	<b>AI</b>	<b>AO</b>	<b>DIO</b>
AT-AO-6		yes	yes
AT-AO-10		yes	yes
AT-MIO-16E-1	yes	yes	yes
AT-MIO-16E-2	yes	yes	yes
AT-MIO-16E-3	yes	yes	yes
AT-MIO-16DE-10	yes	yes	yes
AT-MIO-16XE-50	yes	yes	yes
AT-MIO-16X	yes	yes	yes
AT-MIO-16F-5	yes	yes	yes
CS-2017	CJC		
DAQCard-1200	yes	yes	yes
DAQCard-700	yes		yes
DAQCard-5XX	yes		yes
DAQPad-MIO-16XE-50	yes	yes	yes
DAQPad-1200	yes	yes	yes
Lab-PC+	yes	yes	yes
PC-LP-16	yes		yes
SC-2070	CJC		
SCB-68	CJC		
SCB-100	CJC		

## NISCXI

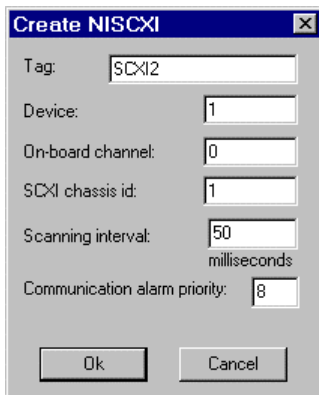
Lookout uses the NISCXI object class to communicate with National Instruments data acquisition devices connected in multiplex mode to SCXI hardware. Refer to the NIDAQDevice object class for more information concerning parallel SCXI configurations, and data acquisition configurations without SCXI.

To use this object, you should have NI-DAQ 5.0 or better software installed. While the NIDAQDevice object will work with some earlier versions of NI-DAQ software, it does not perform well with these earlier versions, and has not been extensively tested.

Consult your NI-DAQ and NI-SCXI hardware, and NI-DAQ software manuals for information on installing and configuring National Instruments data acquisition software and hardware.

When you create an NISCXI object for the first time, or when you launch Lookout and run an application using the NISCXI object, Lookout automatically loads the NI-DAQ software. This can take a few moments, during which the screen is frozen. Once the NI-DAQ software has loaded, Lookout returns to its former speed.

**Note** This object class is available on 32-bit versions of Lookout 3.7 and later. It is not backward compatible with earlier versions of Lookout, and does not run on the 16-bit version of Lookout.



The image shows a dialog box titled "Create NISCXI" with a close button (X) in the top right corner. The dialog contains several input fields with labels to their left: "Tag:" with a text box containing "SCXI2"; "Device:" with a text box containing "1"; "On-board channel:" with a text box containing "0"; "SCXI chassis id:" with a text box containing "1"; "Scanning interval:" with a text box containing "50" and the unit "milliseconds" below it; and "Communication alarm priority:" with a text box containing "8". At the bottom of the dialog are two buttons: "Ok" and "Cancel".

**Device** is the NI-DAQ device number of the controlling device. The DAQ configuration utility (WDAQConf.exe) assigns this number to an installed device. Valid device numbers range is from 1 to 16.

**On-board channel** specifies the analog input channel of the controlling device used to address this SCXI chassis. When only one chassis is being controlled by the controlling board, this value should be 0. When this chassis in a multichassis configuration, this value should reflect the order of this chassis. For instance, the first chassis should be 0, the next chassis should be 1, and so on. Valid range is 0 to 7. If there are no analog input channels in this chassis, use the value 0.

**SCXI chassis id** is the chassis Id number assigned to this chassis by the DAQ configuration utility.

**Scanning interval** is the time period between analog and digital input polls. Valid range is 20 msec to 1 day (expressed in msec).

**Communication alarm priority** determines the priority level of the alarms generated by the NISCXI object.

#### *NISCXI Device Object Class Data Members*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
CommFail	logical	yes	no	Object-generated signal that is on if, for whatever reason, Lookout cannot communicate with or control the device without error.
MD1.AI0, MD12.AI63	numeric	yes	no	Analog input channel in volts.
MD1.AO0, MD12.AO63	numeric	no	yes	Analog output channel in volts.
MD1.cjtemp, MD12.cjtemp	numeric	yes	no	Built-in cold-junction sensor on analog input terminal blocks.
MD1.DI0, MD12.DI31	logical	yes	no	Digital input line.
MD1.DO0, MD12.DO31	logical	no	yes	Digital output line.
Update	logical	yes	no	Object-generated signal that pulses each time the object polls the device.

*(continued)*

### *NISCXI data members*

MD1.Etc0- MD12Etc31	numeric	yes	no	Analog input channel with built-in polynomial conversion for an E type thermocouple.
MD1.Ktc0- MD12Ktc31	numeric	yes	no	Analog input channel with built-in polynomial conversion for a K type thermocouple.
MD1.Jtc0- MD12Jtc31	numeric	yes	no	Analog input channel with built-in polynomial conversion for a J type thermocouple.
MD1.Rtc0- MD12Rtc31	numeric	yes	no	Analog input channel with built-in polynomial conversion for an R type thermocouple.
MD1.Stc0- MD12Stc31	numeric	yes	no	Analog input channel with built-in polynomial conversion for an S type thermocouple.
MD1.Ttc0- MD12Ttc31	numeric	yes	no	Analog input channel with built-in polynomial conversion for a T type thermocouple.

## **NIDAQ.INI**

You may configure individual channel attributes by editing the `NIDAQ.INI` file in your Lookout directory.

### Channel Attributes

To specify a channel, use the following format: `[DEV3.SC1.MD2.AI5]`

This specifies analog input channel 5 on module 2 of SCXI chassis 1 controlled by device 3.

**UPPERLIMIT** specifies the upper input limit in volts (for example, `UPPERLIMIT=2.5`). Usually used in conjunction with **LOWERLIMIT**.

**LOWERLIMIT** specifies the lower input limit in volts (for example, `LOWERLIMIT=-3.7`). Usually used in conjunction with **UPPERLIMIT**.

### Cold-Junction Sensor Attributes

To specify a cold-junction sensor, use the following format: [DEV3.SC1.MD2.CJC]

This specifies the cold-junction sensor on module 2 of SCXI chassis 1 controlled by device 3.

**SENSORTYPE** specifies the type of temperature sensor on the terminal block. Valid choices are **IC** and **THERMISTOR**. Check your SCXI terminal block documentation to determine the correct sensor type.

**RESAMPLE** specifies the number of minutes between resampling the cold-junction sensor. Valid range is 1 to 10080, and 0. Use 0 to only check the cold-junction sensor once at the start, and do not resample the sensor later.

## **Error Messages**

### **Error loading NI-DAQ driver**

Lookout was not able to communicate to the NI-DAQ driver. Be sure that NI-DAQ has been installed properly, and that you used the NI-DAQ Configuration Utility to configure the your hardware devices.

**NIDAQ: Analog Input: Invalid Data Member(s)**

**NIDAQ: Analog Output: Invalid Data Member(s)**

**NIDAQ: Digital Input: Invalid Data Member(s)**

**NIDAQ: Digital Output: Invalid Data Member(s)**

At least one channels you have specified is not valid for your current hardware configuration. Be sure your hardware is configured properly using the NI-DAQ Configuration Utility.

**NIDAQ: Error code: NNNNN**

**NIDAQ: Analog Input: Error code: NNNNN**

**NIDAQ: Analog Output: Error code: NNNNN**

**NIDAQ: Digital Input: Error code: NNNNN**

## NIDAQ: Digital Output: Error code: NNNNN

NI-DAQ has detected an error condition. Please refer to your *NI-DAQ Function Reference Manual* to determine the meaning of the error code.

### Devices

You can use the following National Instruments data acquisition devices with Lookout.

*National Instruments SCXI devices supported by Lookout*

Device	AI	AO	DIO
SCXI-1000			
SCXI-1001			
SCXI-1100	yes		
SCXI-1102	yes		
SCXI-1120	yes		
SCXI-1121	yes		
SCXI-1124		yes	
SCXI-1160			yes
SCXI-1161			yes
SCXI-1162			yes
SCXI-1163			yes
SCXI-1163R			yes
SCXI-1200	yes	yes	yes
SCXI-1300	CJC		
SCXI-1303	CJC		
SCXI-1320	CJC		
SCXI-1328	CJC		
SCXI-2000	yes	yes	yes
SCXI-2400	yes	yes	yes

# Omron

Omron is a protocol driver class Lookout uses to communicate with Omron devices using the Host Link serial communication protocol.

An Omron object contains a great deal of data. It supports reading and writing of all predefined data points. When you create an Omron object, you have immediate access to all the data members for that object (*see data member list below*).

The screenshot shows the 'Create Omron Secondary' dialog box. The 'Tag' field contains 'Omron'. The 'Address' field contains '00' and the 'PLC Model' dropdown is set to 'C20'. Under 'Communication Settings', 'Serial port' is 'COM1', 'Parity' is 'Even', 'Baud Rate' is '9600', 'Stop bits' is '2', and 'Data bits' is '7'. The 'Phone number' field is empty. Below these, 'PollRate' is '0.01' and 'Poll' is an empty field, both highlighted in yellow. At the bottom, 'Communication alarm priority' is '0', 'Retry attempts' is '4', 'Receive timeout' is '250' ms, and 'Skip every' is '5' poll requests after connection fails. 'Ok' and 'Cancel' buttons are on the right.

**Serial port** specifies which comm port the object uses for communicating to the external device. This does not specify the communication type. Communication type is determined by the Options→Serial Ports... command.

**Data rate** indicates the baud rate that Lookout uses to communicate with the hardware device. This **Data rate** setting should match the selection made on the physical device.

**Data bits** indicates the number of data bits that Lookout uses to communicate with the hardware device. This **Data bits** setting should match the selection made on the physical device.



**Stop bits** indicates the number of stop bits that Lookout uses to communicate with the hardware device. This **Stop bits** setting should match the selection made on the physical device.

**Parity** indicates the parity that Lookout uses to communicate with the hardware device. This **Parity** setting should match the selection made on the physical device.

**Phone number** specifies the number to be dialed if the selected serial port is configured for dial-up. This **Phone number** only applies to the individual protocol object.

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. The object then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). See the *Numeric Data Members* section in Chapter 5, *Using Objects*, of your Lookout Reference Manual for more information on entering time constants.

**Poll** is a logical expression. When transitioned from FALSE to TRUE, it causes Lookout to poll the device. A **Poll** could be a simple expression, such as the signal from a pushbutton, or it could be a complex algorithm.

**Communication alarm priority** determines the priority level of alarms generated by the Omron object. Such alarms are typically related to communications with the physical device.

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device when it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Omron object generates an alarm and releases the communication port back to the communications subsystem (COMSUB). The subsystem then moves on to the next device in the polling queue (if any). See Chapter 6, *Serial Communications*, in your Lookout Reference Manual for more information.

**Receive timeout** is the time delay Lookout uses in waiting for a response from a device before retrying the request.

The **Skip every...** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout skips the device in the polling cycle. Once communications have been reestablished, the device is polled on its regular cycle.

As with all Lookout drivers, you can access I/O points and other data through data members. The following is a table of data members currently supported by the Omron object class.

### *Omron data members*

Data member	Type	Read	Write	Description
AR0, AR27	numeric	yes	yes	Auxiliary relay area, read as 16-bit word.
AR0.0, AR27.15	logical	yes	yes	Auxiliary relay area, read as 1-bit discrete.
DM0, DM9999	numeric	yes	yes	Data memory area, 16-bit word.
CommFail	logical	yes	no	Object-generated signal that is on if, for any reason, Lookout cannot communicate with the device(s).
HR0, HR99	numeric	yes	yes	Holding relay area, read as 16-bit word.
HR0.0, HR99.15	logical	yes	yes	Holding relay area, read as 1-bit discrete.
IR0, IR511	numeric	yes	yes	I/O area, read as 16-bit word.
IR0.0, IR511.15	logical	yes	yes	I/O area, read as 1-bit discrete.
LR0, LR63	numeric	yes	yes	Link relay area, read as 16-bit word of information.
LR0.0, LR63.15	logical	yes	yes	Link relay area, read as 1-bit discrete.
TC0, TC999	numeric	yes	no	Timer/Counter, read as 16-bit word.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.

**Note**     **The Omron requires a special cable configuration in order to work properly. See your Omron hardware documentation for the correct configuration.**

## **Status Messages**

### **No response within timeout period**

Lookout received no response from a device within the **Receive timeout** period. The Omron object is able to use the comm port, but when it polls the device, it does not respond—as if it is not even there.

### **Cannot set PLC to MONITOR mode**

The Omron object is trying to set the PLC in MONITOR mode in order to communicate with the PLC correctly, but cannot perform the operation.

### **Incorrect address in response**

The frame received had an incorrect source address. Check for two or more devices with the same address.

### **Incorrect command in response**

The frame received had an incorrect command. Check for two or more devices with the same address.

### **Incorrect data type in response**

The frame received had an incorrect data type marker.

### **Incorrect frame check sum (FCS)**

The frame received had an incorrect check sum.

### **Omron errors reported in the response**

These errors are reported by the Omron device, and are in turn reported to you in text form.

### **Models supported:**

C20, C200, C500, C1000, C2000, CQM, CPM1

# Philips

Philips is a protocol driver class Lookout uses to communicate with Philips devices using the PPCCOMM communication protocol.

A Philips object contains a great deal of data. It supports reading and writing of all predefined data points. When you create a Philips object, you have immediate access to all the object data members (*see data member list below*).

**Create Philips Secondary**

Tag:

Address:  PLC Model:

Communication Settings:

Serial port:  Parity:

Baud Rate:  Stop bits:

Data bits:

Phone number:

PollRate =

Poll =

Communication alarm priority:

Retry attempts:  Receive timeout:  msec

Skip every  poll requests after comm failure

**Serial port** specifies which communications port the object uses for communicating to the external device. This does not specify the communication type. Communication type is determined by the Options→Serial Ports... command.

**Data rate** indicates the baud rate that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Data bits** indicates the number of data bits that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Stop bits** indicates the number of stop bits that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Parity** indicates the parity that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Phone number** specifies the number to be dialed if the selected serial port is configured for dial-up. This number only applies to the individual protocol object.

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. The object then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). *See Numeric Data Members in Chapter 5 for information on entering time constants.*

**Poll** is a logical expression that, when transitioned from false to true, causes Lookout to poll the device. This could be a simple expression like the signal from a pushbutton, or it could be a complex algorithm.

**Communication alarm priority** determines the priority level of alarms generated by the Philips object. Such alarms are typically related to communications with the physical device.

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device if it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Philips object generates an alarm and releases the communication port back to the communications subsystem (COMSUB) which then moves on to the next device in the polling queue (if any). *Refer to the communications chapter for more information.*

**Receive timeout** is the time delay Lookout uses in waiting for a response from a device before retrying the request.

The **Skip every...** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout will skip the device in the polling cycle accordingly. Once communications have been reestablished, the device will be polled on its regular cycle.

As with all Lookout drivers, you can access I/O points and other data through data members. The following is a table of data members currently supported by the Philips object class.

*Philips data members*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
IX0.0 - IX123.15	logical	yes	no	Discrete input, 1 bit
IB0.0 - IB1023.1	numeric	yes	yes	Discrete inputs, 8 bits
IW0 - IW1023	numeric	yes	yes	Discrete inputs, 16 bits
ID0 - ID1023	numeric	yes	yes	Discrete inputs, 32 bits
MX0.0 - MX123.15	logical	yes	no	Data register, 1 bit
MB0.0 - MB1023.1	numeric	yes	yes	Data register, 8 bits
MW0 - MW1023	numeric	yes	yes	Data register, 16 bits
MD0 - MD1023	numeric	yes	yes	Data register, 32 bits
QX0.0 - QX123.15	logical	yes	no	Discrete output, 1 bit
QB0.0 - QB1023.1	numeric	yes	yes	Discrete outputs, 8 bits
QW0 - QW1023	numeric	yes	yes	Discrete outputs, 16 bits
QD0 - QD1023	numeric	yes	yes	Discrete outputs, 32 bits
Poll	Logical	no	yes	Lookout expression that when transitioned false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device's polling frequency.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.
CommFail	logical	yes	no	Object-generated signal that is on if Lookout cannot communicate with the device(s)

## Status Messages

### Invalid frame format

### Bad Checksum

### Invalid frame length

These messages indicate garbled communications. Check cabling for signal noise or multiple devices with the same address.

### Philips alarms

These are messages returned to Lookout in a response frame. See your Philips documentation for meanings and solutions

## Profibus DP Object Class

Profibus DP is a protocol driver class for communicating with PLCs and remote I/O units using the Decentralized Periphery protocol in the Profibus standard. The Lookout driver currently supports the S-S Technologies card 5136-PFB PC card only. Support for the Hilscher/Synergetic card CIF30-DPM will be available in the future, but is not included in this release.

## Configuring the Profibus DP network

In order to use Lookout's Profibus DP driver for the S-S card, you must first configure your DP network using the Siemens COM ET-200 software. This software is not included in the Lookout release and must be purchased separately. The S-S card is the master device on the Profibus network. Because the COM ET-200 software does not know about this device, you will have to select some other device as the master. You may leave all of its parameters as the defaults because the S-S card ignores the parameters of this master system.

You must, however, enter a master station number and configure the bus parameters. These parameters are used by the SS card to configure the DP network.

Next, you must configure the parameters for each of the slave devices. Refer to the online documentation for the COM ET200 software for instructions on how to do this. Once all the slave devices are configured, save the file to an .et2 format and export the file to a binary file .2bf

format. This last step is done using the Export selection in the File menu. Place the binary file in a location accessible to Lookout.

## Requirements

To run the 16-bit version of this object, you must be running Windows 3.1 or Windows 95, with an S-S Technologies 5136-PFB card installed in the computer. Pre-install the firmware module `pfbprofi.ssl` on the card (using the `pfbinst` utility). The usual way to do this is to insert a call to `pfbinst` in your `autoexec.bat` file.

To run the 32-bit version of this object, you must be running Windows NT 3.51 or greater or Windows 95, with an S-S Technologies 5136-PFB card installed in the computer. Lookout automatically loads the firmware module when an object is created.

Both versions require that the configuration file in COM ET 200 binary format be placed in a directory accessible to Lookout.

The screenshot shows a Windows-style dialog box titled "Create Profibus DP". It has a standard title bar with a close button (X). The dialog contains the following elements:

- Tag:** A text box containing "ProfiDP1".
- Interface:** A dropdown menu showing "SS Tech 5136-PFB".
- S-S Card Settings:** A section with a grey background containing:
  - ComET200 binary file:** A text box with "D:\COMWIN20\PROG" and a "Browse" button to its right.
  - Card I/O Port:** A text box containing "250".
  - Card Base Address:** A text box containing "0000".
  - Card Scan Rate (ms):** A text box containing "50".
- Slave Address:** An empty text box.
- Alarm Level:** A text box containing "8".
- Buttons:** "OK" and "Cancel" buttons are located at the bottom right of the dialog.

**Interface** is the type of card to be used to communicate with the DP network. At present, this driver only supports the S-S card interface.



**Slave Address** is the Profibus address for the particular slave device that this object is to communicate with.

**Alarm Level** determines the priority level of object-generated alarms (0-10).

### PFB Card Settings

**COM ET200 Binary File** is the path to the binary configuration file produced by the Siemens COM ET200 software. This file must match the network that the S-S card is actually connected to.

**Card I/O Port** specifies the base port address for the card. The jumpers on the card must be preset to the port address selected.

**Card Base Address** specifies the base address location of the card memory. At present, only one 5136-PFB card in a computer is supported. The default is D000. In 16-bit Lookout, this value is preset when loading the firmware onto the card. In 32-bit Lookout, the Profibus DP object itself sets this on the card when it is loading the firmware.

**Network Init Timeout** specifies the number of milliseconds the card should wait while attempting to initialize the network before generating an alarm. The default is 1000 ms or one second.

**Card Scan Rate** is the rate at which the 5136-PFB card is scanned to look for fresh data. The default is 50 milliseconds.

## **Profibus DP Data Members**

All readable and writable members—inputs/outputs—are bundled with the object. Therefore, as soon as you create a Profibus DP object you immediately have access to all the object data members.

Data can be addressed either as bytes or words within slots, or as bits within these bytes or words. However, you cannot address a slot configured for digital input or output (as bytes) using word numbers. Similarly you cannot address a slot configured for analog input or output (as words) using byte numbers.

### *Profibus DP data members*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
CommFail	logical	yes	no	Object-generated signal that is on if, for any reason, Lookout cannot communicate with the Profibus card.
Slot0B0 - Slot127B63	numeric	yes	yes	This addresses data in a slot as a sequence of bytes.
Slot0B0.0 - Slot127B63.7	logical	yes	yes	This addresses individual bits while looking at a slot as a sequence of bytes.
Slot0W0 - Slot127W63	numeric	yes	yes	This addresses data in a slot as a sequence of words.
Slot0W0.0 - Slot127W63.7	logical	yes	yes	This addresses individual bits while looking at a slot as a sequence of words.

## **Status Messages**

### **Profibus SS Card not found**

The SS card was not found at the specified port and card base address. Check that the jumper settings on your card match the port address that you specified in the object creation dialog box. Also, check that the card is properly seated in the slot. In the 16-bit version, make sure you are loading the `pfbrofi.ssl` module onto the card prior to starting Lookout.

### **Profibus init failed**

The initialization of the Profibus SS card failed. This alarm is usually followed by an explanation of the reason for this failure.

### **Error accessing COM ET 200 binary file**

A file error occurred while attempting to open the COM ET 200 binary file.

## **Unconfigured Read Error**

A read was attempted on a data member that is not valid in the current Profibus DP network configuration.

## **Unconfigured Write Error**

A write was attempted on a data member that is not valid in the current Profibus DP network configuration.

## **Slave Error**

There was a problem with the particular slave device you are trying to access. This alarm is followed by a description of the problem with that slave device.

# **ProfibusL2 Object Class**

ProfibusL2 is a protocol driver class for communicating with PLCs using Profibus Sinec-L2.

One of the problems with communicating with PLCs using Profibus Sinec-L2 is that there is no standard messaging system for transferring data. The Sinec L2 interface has a variety of message types available that you can use with applications as well as PLCs on a network. Lookout uses its own messaging system for transferring data and provides help for you to implement the PLC end of the messaging system.

## **Lookout Messaging System**

You should read the Siemens manual *Sinec-L2 Interface of the S5-95U Programmable Controller*, or some similar document that details a Sinec-L2 interface and how to write programs for your particular Sinec-L2 interface device.

Lookout uses two message types to receive and send all the data it needs. It uses the message type SRDh (Send and Request Data, high priority) to receive data from the PLC. To send data to the PLC, Lookout uses the message type SDAh (Send Data with Acknowledge, high priority).

All messages in Sinec-L2 are directed at specified SAPs (Service Access Points). You must program SAPs on each PLC to provide or accept the data you want to exchange with Lookout. You can use a given SAP only for reading or writing. Each poll request for a configured SAP

reads *all* the data in that SAP. In the same way, if you are writing to a particular SAP, each write transfers all the data for that SAP to the PLC.

## Sample Program

Included in your copy of Lookout is a sample Siemens STEP5 program file, LKTPFBST.S5D. This file is placed in the \sinec12 directory under your Lookout directory during installation. This program contains an example of the logic necessary to create the Service Access Points (SAPs) for communications from a Simatic S5-95U to Lookout software using the Profibus protocol.

The program consists of one program block, three function blocks, three data blocks, and the modifications needed in OB1 and DB1 to successfully configure the Profibus communication link. This sample establishes two SAPs to the Profibus Layer 2 services of the PLC. Lookout uses the first SAP, 34, to write values to the PLC. Lookout uses the second SAP, 35, when reading values from the PLC. You can read or write data in byte, word, or double word formats.

In order to install the example Profibus program, follow the steps below.

1. **TRANSFER ALL BLOCKS TO THE CPU MEMORY**

This must include all PBs, FBs, and DBs.

2. **PROFIBUS CONFIGURATION**

The example configures SAP 34 and SAP 35. However, you may modify the program, if necessary, to customize your application.

3. **PROFIBUS EXECUTION**

Jump to PB 223 is required from OB1. An example of this jump is included in OB1 in the program.

4. **PROFIBUS DATA HANDLING**

The sample program uses DB 234, DB 235, and DB 236 to move data from the PLC to Lookout and from Lookout to the PLC. All data must be mapped to and from data blocks for Profibus communications. These data blocks also contain 8-byte headers used as a part of the Profibus configuration parameters. Again, you may modify these blocks if necessary. In addition, flag bytes above 200 are used by the example program, so you should avoid using them.

## Detailed Explanation of the Profibus Example Program

The example program employs the Layer 2 services of the Siemens Simatic S5-95U for the Profibus communications to Lookout. In particular, the program uses SRD and RUP\_MULTIPLE. The configuration of these services is described in detail in Chapter 8, *Data Transmission by Accessing Layer 2 Services*, of the *Sinec-L2 Interface of the S5-95U Programmable Controller* manual.

### DB1 Configuration

The first step in configuring the S5-95U is to insert the correct Profibus parameters in DB1 of the PLC. You should employ the COM\_DB1 software, available from Siemens, when setting these parameters.

First, you must enter the SINEC L2 basic parameters. These parameters include the station address, baud rate, target rotation time, and more. The parameters must exactly match those used in configuring your SinecL2 object. You can find the defaults for these parameters in the example program where they are used.

Next you must configure the SINEC L2 layer 2 service. Here, you define the individual SAPs and designate a status byte for each SAP. SAPs used for writing data to the PLC must have a status byte allocated in the **Receive** area. SAPs used for reading data from the PLC must have a status byte allocated in both the **Send** and **Receive** areas.

You can define up to 23 SAPs, numbering from 33 to 54 and also including 64. The example program uses SAP 34 for writing data, and FW 204 as its status and length bytes. The program uses SAP 35 for reading data, with FW 200 and FW 202 used for the status and length bytes in the **Send** and **Receive** areas.

### Function Block Explanation

The example program contains three function blocks to handle the data transfer between the Profibus SAPs and the CPU internal memory. These blocks evaluate the status bits associated with each SAP, and send or receive the data when allowable.

Function Block FB 224 coordinates the data written from Lookout to the PLC. The CPU looks for an *indication* from the SAP that new data exists. If the receive is viable, the block performs a jump to FB 253, which is the integrated L2-REC function block. The block takes the data and writes it to the area specified by DBNR. The example uses DB 234, starting at DW 0 and continuing for an unknown wildcard length. The first 8 bytes of DB 234, DW 0 through DW 3,

contain the header for the message. You must include this header in the data block for proper operation.

Function Blocks FB 223 and FB 225 handle the data Lookout is attempting to read from the PLC. FB 225 acts in the same way as FB 224, looking for an indication that the SAP has received a new request for data. FB 223 actually writes the data to the SAP using a jump to FB 252, the integrated L2-SEND function block. The data is taken from the area designated by DBNR. In this case, DB 235 is used by FB 223 and DB 236 is used by FB 225. The first 8 bytes act as a header to the message to follow, and must be set with specific parameters. In DB 235, DW 1 through DW 4 are used for the Send, and DW 128 through DW 131 are used for the Indication. In DB 236, DW 1 through DW 4 are used for the Indication. The parameters required in the data blocks are discussed in the *Sinec-L2 Interface of the S5-95U Programmable Controller* manual.

Program Block PB 223 initiates the jumps to all the function blocks for the Profibus communications. You must add a jump to PB 223 in your OB 1 in order to start the communications.

Finally, all the program blocks and function blocks use bits of the **Send** and **Receive** status bytes in their logic. If you change these bytes in DB 1, the you must also change the associated logic accordingly.

## Requirements

To run the 16-bit version of this object, you must be running Windows 3.1 or Windows 95 and have an S-S Technologies 5136-PFB card installed in the computer. You must pre-install the firmware module `pfbprofi.ssl` on the card (using the `pfbinst` utility). The usual way to do this is to insert a call to `pfbinst` in your `autoexec.bat` file

To run the 32-bit version of this object, you must have Windows NT 3.51 or better, or Windows 95 loaded, and an S-S Technologies 5136-PFB card installed in the computer. Lookout automatically loads the firmware module when an object is created.

**Note**     **This object class is available on Version 3.6 and later. It is not backward compatible to earlier versions.**

**Create Profibus SinecL2**

Tag:

**PFB Card Settings**

Card Memory Address:  Base Port Address:

**Profibus Parameters**

Card Network Address:  Network Baud Rate:  Idle Time1:

PLC Network Address:  Network High Address:  Idle Time2:

Token Rotation Time:  Slot Time:  Ready Time:

PollRate =

Poll =

Communication alarm priority:

Retry Attempts:

PLC's network address must be a number less than 255

Ok Cancel

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. Lookout then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). See the *Numeric Data Members* section of Chapter 5, *Using Objects*, of your *Lookout Reference Manual* for further information on entering time constants.

**Poll** is a logical expression that, when transitioned from false to true, causes Lookout to poll the device. This could be a simple expression like the signal from a pushbutton, or a complex algorithm.

**Communication alarm priority** determines the priority level of object-generated alarms (0-10).

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device if it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Profibus L2 object generates an alarm.

**Card Memory Address** specifies the base address location of the card memory. At present, only one 5136-PFB card in a computer is supported. The default is D000. In 16-bit Lookout, this value is preset when loading the firmware onto the card while in 32-bit Lookout, the Profibus object itself sets this on the card when it is loading the firmware.

**Base Port Address** specifies the base port address for the card. The jumpers on the card must be preset to the port address selected.

### PFB Card Settings

**Card Network Address** specifies the desired Profibus address for the card on the bus. The allowed range is 0 to 126.

**PLC Network Address** specifies the Profibus address for the PLC on the bus. The allowed range is 0 to 126.

**Network High Address** specifies the highest possible Profibus address possible on the bus. The allowed range is 0 to 126.

**Token Rotation Time** specifies the target maximum token rotation time for the network in Tbits (bit times). The allowed range is 256 to 16,777,215.

**Network Baud Rate** specifies the baud rate to be used on the network. The allowed selections are shown in the selection box.

**Slot Time** specifies how long the card waits for a reply to a message, in Tbits. The allowed range is 37 to 16,383.

**Idle Time1** specifies the time in Tbits that the card waits after it receives a reply, an acknowledge or a token message before sending a message. Range: 35 to 1023

**Idle Time2** specifies the time in Tbits that the card waits after sending an SDN (Send Data with no acknowledge) message before it sends again. Range: 35 to 1023

**Ready Time** specifies the time in Tbits that the card, after it sends a command, is ready to receive the ACK or response. It is also the time the card waits after receiving a command. The allowed values range from 11 to 1023.



## Profibus Data Members

All readable and writable members (inputs/outputs), polling instructions, and so on, are bundled with the object. Therefore, as soon as you create a Profibus object you immediately have access to all the object data members (see data member list below).

Data is addressed using SAP numbers and, within a SAP, the byte, word or Dword number. For example, if SAP 33 is configured for reading, SAP33B200 refers to the 201st byte of that SAP. Similarly, SAP42DW10.20 refers to the 21st bit of the 11th Dword of SAP 42.

The legal SAPs allowed by the Lookout Profil2 driver range from 0 to 255. Each SAP may provide up to 242 bytes of data (0-241) which you can address either as bytes, 121 words, or 60 Dwords. The actual range of allowed SAPs varies according to the device being addressed. You can address bits by appending a period and bit number after the byte number, word number, or Dword number.

**Note** Lookout protocol driver objects automatically generate an efficient read/write blocking scheme based on the inputs and outputs being used in your process file. You are not required to build your own I/O blocking table.

### *ProfibusL2 data members*

Data member	Type	Read	Write	Description
CommFail	logical	yes	no	Object-generated signal that is on if, for any reason, Lookout cannot communicate with the Profibus card.
Poll	logical	no	yes	Lookout expression that when transitioned from false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device polling frequency.
SAP0B0-SAP255B241	numeric	yes	yes	This addresses all the data in the SAP as a sequence of bytes.
SAP0B0.0-SAP255B241.7	logical	yes	yes	This addresses individual bits while looking at an SAP as a sequence of bytes.
SAP0W0-SAP255W120	numeric	yes	yes	This addresses all the data in the SAP as a sequence of words.

### *ProfibusL2 data members (continued)*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
SAP0W0.0-SAP0255W120.15	logical	yes	yes	This addresses individual bits while looking at an SAP as a sequence of words.
SAP0DW0-SAP255DW60	numeric	yes	yes	This addresses all the data in the SAP as a sequence of double words.
SAP0DW0.0-SAP255DW60.31	logical	yes	yes	This addresses individual bits while looking at an SAP as a sequence of double words.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.

## **Status Messages**

### **Profibus SS Card not found**

The SS card was not found at the specified port and card base address. Check that the jumper settings on your card match the port address that you specified in the object creation dialog box. Also, check that the card is properly seated in the slot. In the 16-bit version, make sure you are loading the `pfbrofi.ssl` module onto the card prior to starting Lookout.

### **Profibus SS Card timed out on message**

The SS card did not respond within a reasonable time to the message sent by Lookout. This may mean that you have lost communication with the card. Try restarting Lookout to see if this fixes the problem (reloading the firmware if you are using 16-bit Lookout). If the problem persists, call National Instruments for assistance.

## Reliance Object Class

Reliance is a protocol driver class for communicating with Reliance AutoMate controllers. This object does not run under Windows NT at this time.

**Note** This object class is available on Version 3.6 and later. It is not backward compatible with earlier versions.

The screenshot shows the 'Create Reliance' dialog box with the following settings:

- Tag:** Reliance1
- Interface:** PC-Link
- PC-Link Card Settings:**
  - Memory Address: Board not found
  - Port Address: (empty)
  - Interrupt: None
  - Node ID: (empty)
  - Max Nodes: (empty)
- Destination Settings:**
  - Node ID: 1
  - Slot ID: 1
  - PollRate: 0:01 (highlighted in yellow)
  - Poll: (empty, highlighted in yellow)
  - Communication alarm priority: 0
  - Retry attempts: 4
  - Receive timeout: 500 ms
  - Skip every 5 poll requests after comm failure

**Note** If you are developing your application without a PC-Link card installed in your computer, and do not know the settings that in the target system, you can enter 250 for Port Address, 0 for Node ID, and 3 for Max Nodes. You can then create the Reliance object and correct the parameter values for the actual system later, if necessary.

**Interface** is the communications method the object uses to communicate with the Reliance PLCs. Currently, only the Reliance R-Net is supported, using the Reliance PC-Link card.

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. Reliance then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). *See Numeric Data Members in Chapter 5 for information on entering time constants.*

**Poll** is a logical expression that, when transitioned from false to true, causes Lookout to poll the device. This could be a simple expression like the signal from a pushbutton, or a complex algorithm.

**Communication alarm priority** determines the priority level of object-generated alarms (0-10).

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device if it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Reliance object generates an alarm and releases the communication port back to the communications subsystem (COMSUB) which then moves on to the next device in the polling queue (if any).

**Receive timeout** is the time delay Lookout uses in waiting for a response from a device before retrying the request.

The **Skip every...** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout skips the device in the polling cycle accordingly. Once communications have been reestablished, the device will be polled on its regular cycle.

## PC-Link Card Settings

**Port Address** is the I/O port address used to access the software module on the card. See your PC-Link card documentation for specific settings.

**Interrupt** identifies the interrupt (IRQ) setting of your PC-Link card. Any time the card receives an response, it generates an interrupt recognized by Lookout.

**Node ID** is the node ID setting of the PC-Link card on the R-Net.

**Max Nodes** is the maximum number of nodes on the R-Net.

## Destination Settings

**Node ID** is the node ID on the R-Net setting of the AutoMate processor that controls the target AutoMate PLC.

**Slot ID** is the ID of the PLC rack slot in which the target AutoMate resides.

## Reliance Data Members

All readable and writable members (inputs/outputs), polling instructions, read/write blocking, serial port usage, and so on, are bundled with the object. Therefore, as soon as you create an Reliance object you immediately have access to all the object data members (see data member list).

**Note** Lookout protocol driver objects automatically generate an efficient read/write blocking scheme based on the inputs and outputs being used in your process file. You are not required to build your own I/O blocking table.

*Reliance data members (all addresses are in octal)*

Data member	Type	Read	Write	Description
0-157775	numeric	yes	yes	16-bit input registers encoded as unsigned binary integers ranging from 0 to 65,535.
0.0-15775.17	logical	yes	yes	Access individual bits in holding registers and read them as logical on/off values. The least significant bit is 1; the most significant bit is 17.
CommFail	logical	yes	no	Driver-generated signal that is on if Lookout cannot communicate with the device.
D0-D157774	numeric	yes	yes	Wide data format ranges from -99,999,999 to +99,999,999.
Poll	logical	no	yes	Lookout expression that, when transitioned from false to true, causes Lookout to poll the device.
PollRate	numeric	no	yes	Lookout expression that determines the polling frequency of the device.
Update	logical	yes	no	Driver-generated signal that pulses each time the driver polls the device.

## Status Messages

See your Reliance or PC-Link documentation for details of Reliance device generated messages.

### No response within timeout period

Lookout did not received the expected response within the **Receive timeout** period. The object sent an inquiry and received an acknowledgment, but the device did not send an expected response to the request. This might happen if the response was interrupted. You may have to increase **Receive timeout**.

### No return inquiry response from secondary unit

Lookout received no response whatsoever from the device within the **Receive timeout** period. The driver object is able to use the COM port, but when it polls the device, it doesn't respond—as if it is not even there. You may have to increase **Receive timeout** to ensure Lookout is allowing enough time to receive the expected response. Also, verify your baud rate settings, cable connections, power, configuration settings, COM port settings, and polling addresses.

### Bad LRC or BCC

The object is receiving a poll response from the device, but it could not decipher the response because it is garbled. Verify that all devices connected to the COM port have unique addresses. The last part of the message may actually be getting clipped off before it is completed. Consider increasing the number of **Retry attempts**. You may have to increase the **Receive gap** Serial Port setting to ensure Lookout is receiving the entire message. If your Serial Port is configured for radio, this could be caused by an audible squelch tail occurring at the end of a radio transmission. Try adjusting **RTS delay off** and **CTS timeout**.

### No attach response within timeout period

An attempt was made to establish communications with the PLC without any response. Check your cabling and COM port selections, power, configuration settings, and polling addresses.

### Invalid response [x]

An error in the structure of a response frame was detected. You may have two PLCs with the same address.

### **Incorrect response length [x]**

A response was received with an unexpected length. You may have to increase the **Receive gap** Serial Port setting to ensure Lookout is receiving the entire message.

### **Incorrect response Address**

A response was received with an address not matching the objects address. You may have two master devices on the network.

### **Card not found**

The reliance object cannot find the card at the **Port Address** specified.

### **Invalid card parameters**

The card parameters in the `reliance.ini` file are not valid. The `reliance.ini` file may be corrupted.

### **Timed out waiting for card ready**

The card timed out during initialization.

### **Timed out waiting for a response**

The card timed out waiting for a response from a PLC. Either the PLC is not set up properly, or the **Timeout** value for the object is set too low.

### **Received an unexpected message**

The Reliance object received a message it did not expect from the PLC. This can be caused by mechanical interference by the PLC. This should be a transitory condition.

### **Card not ready to send**

The card was not ready to send a message from the Reliance object. This indicates a severe problem with the card. The Reliance object will automatically reset the card and continue communicating when the card is initialized.

## S5\_3964 Object Class

S5\_3964 is a protocol driver class Lookout uses to communicate with Siemens Simatic S5 PLCs using the 3964 or 3964R protocols.

**Note** This object class is available on Version 3.7 and later. It is not backward compatible with earlier versions.

The screenshot shows a Windows-style dialog box titled "Create SimaticS5 3964". It contains the following fields and controls:

- Tag:** A text box containing "S5\_3964\_1".
- PLC Model:** A dropdown menu showing "100U".
- Protocol:** A dropdown menu showing "3964".
- PollRate =** A text box containing "0:01".
- Poll =** An empty text box.
- Communication alarm priority:** A text box containing "8".
- Receive timeout:** A text box containing "550" followed by "msecs".
- Retry attempts:** A text box containing "8".
- Skip every**  **poll requests after comm failure**
- Communication Settings:** A section containing:
  - Serial port:** A dropdown menu showing "COM1".
  - Baud Rate:** A dropdown menu showing "9600".
  - Phone number:** An empty text box.
- Buttons:** "Cancel" and "Ok" buttons at the bottom right.

### S5\_3964 Parameters

**PLC Model** specifies the model of Simatic S5 CPU the object will communicate with.

**Protocol** specifies the protocol used to communicate with the PLC.

**Serial port** specifies which port the object uses for communication to the PLC.

**Baud Rate** specifies the speed at which the object communicates with the PLC.

**Phone number** specifies the number to be dialed if the serial port setting is configured for dial-up. This number only applies to the individual protocol object.



**PollRate** is a numeric expression that determines how often to poll the device. Normally, this is a simple time constant such as 0:01 (one second). See the *Numeric Data Members* section in Chapter 5, *Using Objects*, of the *Lookout Reference Manual* for information on entering time constants.

**Poll** is a logical expression that, when transitioned from false to true, causes Lookout to poll the device. This could be a simple expression like the signal from a pushbutton, or a complex algorithm.

**Communication alarm priority** determines the priority level of object-generated alarms (0-10).

**Receive timeout** is the time the object waits for a response from a device before retrying a request.

**Retry attempts** specifies the consecutive number of times the object attempts to establish communications with a device if it is not getting a valid response. After it tries the number of **Retry attempts**, the object generates an alarm and releases the communication port.

The **Skip every...** setting instructs the object that, in the event of a communications failure, to skip the next specified number of polls to the PLC before attempting to re-establish communications. Once communications have been re-established, the device will be polled on its regular cycle.

## S5\_3964 Data Members

This protocol driver object contains a great deal of data. All readable and writable members (inputs/outputs), polling instructions, read/write blocking, serial port usage, and so on, are bundled with the object. As soon as you create a S5\_3964 object, you immediately have access to all the data members of that object.

**Note**     **Lookout's protocol driver objects automatically generate an efficient read/write blocking scheme based on the inputs and outputs being used in your process file. You are not required to build your own I/O blocking table.**

The suffixes for the PLC data members below (KC, KF, and so on) follow the Siemens format for data suffixes. That is, KC for Counter format, KT for Timer format, KB for byte format, KF for fixed-point format, KG for floating-point format, and no suffix for those data members for which only one format is possible, such as bit fields, counters, and timers.

## S5\_3964 data members

CommFail	logical	yes	no	Object-generated signal that is on if the object cannot communicate with the PLC.
Poll	logical	no	yes	Lookout expression that when transitioned from false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device polling frequency.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device
A0KC - A65535KC	numeric	yes	no	Absolute address.
A0KF - A65535KF	numeric	yes	no	Absolute address.
A0KF - A65535KT	numeric	yes	no	Absolute address.
C0 - C255	numeric	yes	no	Counter.
DB1D0.0 - DB255D255.15	logical	yes	no	A bit in a Data Block word.
DB1DL0KB - DB255DL255KB	numeric	yes	no	The left byte in a Data Block word.
DB1DR0KB - DB255DR255KB	numeric	yes	no	The right byte in a Data Block word.
DB1DW0KC - DB255DW255KC	numeric	yes	yes	Word in aData Block.
DB1DW0KF - DB255DW255KF	numeric	yes	yes	Word in aData Block.
DB1DW0KT - DB255DW255KT	numeric	yes	yes	Word in aData Block.
DB1DD0KG - DB255DD255KG	numeric	yes	yes	A double-word in a Data Block.

*S5\_3964 data members (continued)*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
DX1D0.0 – DX255D255.15	logical	yes	no	A bit in an Extended Data Block word.
DX1DL0KB – DX255DL255KB	numeric	yes	yes	The left byte in an Extended Data Block word.
DX1DR0KB – DX255DR255KB	numeric	yes	no	The right byte in an Extended Data Block word.
DX1DW0KC – DX255DW255KC	numeric	yes	yes	Word in in an Extended Data Block.
DX1DW0KF – DX255DW255KF	numeric	yes	yes	Word in in an Extended Data Block.
DX1DW0KT – DX255DW255KT	numeric	yes	yes	Word in in an Extended Data Block.
DX1DD0KG – DX255DD255KG	numeric	yes	yes	A double-word in an Extended Data Block.
F0.0 – F255.7	logical	yes	no	A bit in Flag byte.
FY0KB – FY255KB	numeric	yes	no	A Flag byte.
FW0KF – FW254KF	numeric	yes	no	A Flag word.
FD0KG – FD252KG	numeric	yes	no	A Flag double-word.
I0.0 – I127.7	logical	yes	no	A bit in a byte of the Input (PII) data area.
IB0KB – IB127KB	numeric	yes	no	A byte of the Input (PII) data area.
IW0KF – IW126KF	numeric	yes	no	Word in of the Input (PII) data area.
ID0KG – ID124KG	numeric	yes	no	A double-word of the Input (PII) data area.

### *S5\_3964 data members (continued)*

Q0.0 - Q127.7	logical	yes	no	A bit in a byte of the Output (PIO) data area.
QB0KB - QB127KB	numeric	yes	no	A byte of the Output (PIO) data area.
QW0KF - QW126KF	numeric	yes	no	Word in of the Output (PIO) data area.
QD0KG - QD124KG	numeric	yes	no	A double-word of the Output (PIO) data area.
RS0KC - RS511KC	numeric	yes	no	Word in in the System data area.
RS0KF - RS511KF	numeric	yes	no	Word in in the System data area.
RS0KT - RS511KT	numeric	yes	no	Word in in the System data area.
T0 - T255	numeric	yes	no	Timer.

## Alarms

### Object Alarms

The following alarms originate in the object, and are only generated after the object has retried the request the number of times specified by the **Retry attempts** parameter.

#### **No response from PLC**

Lookout did not receive a response from the device within the **Receive timeout** period. The driver object is able to use the COM port, but when it polls the device, the device doesn't respond. You may have to increase **Receive timeout** to ensure Lookout is allowing enough time to receive the expected response. Also, verify your baud rate settings, cable connections, power, configuration settings, COM port settings, and polling addresses.

## **Unexpected response from PLC**

A response was received from the PLC, but not the response expected according to the protocol.

## **Bad frame**

A response frame was received from the PLC, but the frame is not valid according to the protocol. This is usually caused by a truncated frame. You may need to increase the **Receive Gap** setting in the Options→Serial Ports... dialog.

## **Bad BCC**

The BCC computed by the object for a received frame did not match the BCC in the frame.

## PLC Alarms

The following alarms originate in the PLC, and are generated immediately by the object. There are no retry attempts.

## **Illegal DB/DX number**

The Data Block (DB) or Extended Data Block (DX) number in a read/write request to a PLC was not valid for the PLC.

## **Synchronization error**

The PLC and S5\_3964 object are not synchronized within the protocol. This usually happens after the object is modified and a read/write request was interrupted. After generating the alarm, the S5\_3964 object will attempt to resynchronize the protocol in the PLC.

# **SqlExec**

SqlExec is an object Lookout uses to communicate with the ODBC driver. ODBC allows Lookout to connect to any database that supports ODBC.

SqlExec connects to ODBC using the standard ODBC calls, and queries the database using SQL statements. The data, if any, is returned in the object data members.

The image shows a 'Create Sql' dialog box with the following fields and controls:

- Tag:** A text box containing 'Sql1'.
- Data Source=**: An empty text box.
- SQL=**: A large empty text box for the SQL statement.
- Alarm priority:** A spin box set to '8'.
- Buffer SQL**: An unchecked checkbox.
- Buttons:** 'Ok' and 'Cancel' buttons at the bottom right.

**Data Source** specifies the Data Source Name (DSN) as well as any other parameters needed by the ODBC driver to make the connection to the data source. For example if you want to connect to Excel 4.0, use

DSN=Excel Files; DBQ=C:\pathname;

**SQL** is the SQL string you want to pass to the ODBC driver.

**Alarm Priority** determines the priority level of object-generated alarms.

**Buffer SQL** tells the SqlExec object to buffer any entries that fail for that object. An entry consists of the Data Source string and the SQL string.

#### *SqlExec data members*

Data member	Type	Read	Write	Description
Execute	logical	no	yes	Executes the submitted SQL statement when it receives a change from low to high.
ReadOnly	logical	yes	yes	If true, opens the file in read only mode. If false, opens the file in read write mode.
Status	logical	yes	no	High when the object is processing a query.

### *SqlExec data members (continued)*

Failure	logical	yes	no	High if the last submitted query failed.
c1-c65535	numeric	yes	no	Value of the <i>n</i> th column in the row returned by ODBC.
c1-c65535.txt	text	yes	no	Value of the <i>n</i> th column in the row returned by ODBC.
c1-c65535.logical	logical	yes	no	Value of the <i>n</i> th column in the row returned by ODBC.

## Comments

The placement of the data in the data members is determined by the SQL string. If you submit the SQL string

```
"select Album, Artist, NumTracks from cdTable where NumTracks>5"
```

the return value for Album will be contained in the data member `c1.txt`, Artist in `c2.txt`, and NumTracks in `col3`. This value is displayed according to the data member you choose—Text, Logical, or Numeric. If you connect both to `c1` and `c1.txt`, you are connecting to the same value twice. In these circumstances, the value is represented in a different form.

A select statement only returns the first row that it encounters with the matching criteria. An update statement updates all the rows it encounters with the matching criteria.

If you connect your SqlExec object to a timer that pulses faster than several times per second, Lookout may become so busy handling SQL queries that it becomes unresponsive to user input from the mouse or keyboard.

Comma Separated Value (CSV) files are supported in ODBC. You can append data to them using the SQL insert command, but you cannot update records using the SQL update command. This is a limitation of the CSV file format.

- Excel files are accessible through ODBC, but they do have some unusual properties.
- If you use SqlExec to update an Excel 5.0 or greater workbook, the workbook must be closed and have multi-user editing turned off.

- If you are just trying to read from a workbook using a select query, then multi-user editing can be on, and the file can be opened or closed.
- If you try to access an Excel 5.0 or greater workbook while it is open and multi-user editing is off, Lookout will lock up until the Excel file is closed. This is caused by a problem in the Excel ODBC driver.
- If you are using Excel 4.0 you can update or read from the worksheet only when it is closed. If you try to access an Excel 4.0 worksheet while it is open, you will receive an alarm in Lookout.

Citadel is also accessible through ODBC, but is opened exclusively in read-only mode to protect the integrity of the data.

Listed below are a few sample DSN and SQL strings for connecting to different types of databases through ODBC. Notice that they differ slightly from database to database. Remember that **Data Source** and **SQL** are expressions. If you enter a string into the expression box, the string needs to be properly set off by quotes.

Database Type	Data Source String	SQL String
CSV	DSN=Text Files; DBQ= c:\ldev\ald2csv;	select Priority, "Date & Time", Description from alarm.csv where Priority = 4
Excel 4.0	DSN=Excel Files; DBQ= c:\ldev\ald2csv;	select Priority, "Date & Time", "Alarm Codes (Set:Reset:Acknowledge)", Description from alarm.xls alarm where Priority = 4
Excel 5.0	DSN=Excel Files;	select Priority, "Date & Time", "Alarm Code (Set:Reset:Acknowledged)", Description from Table1 where Priority = 4
MS Access	DSN=MS Access 7.0 Database; DBQ=c:\My Documents\compact.mdb;	select Title, AlbumID, Length from tracks where TrackID = 7
Citadel	DSN=Citadel 32-bit;	select Interval, LocalTime, "Waves.Square" from Threads where Interval = 0:1 and LocalTime > "2/14/97 14:34:30"



## Buffering

If you check the **Buffer SQL** checkbox in the Create SQL dialog box, the object will buffer entries that fail for the following reasons.

- Unable to connect to data source: error code 08001.
- Connection in use: error code 08002.
- Communication link failure: error code 08S01.
- Drivers SQLAllocEnv failed: error code IM004.
- Drivers SQLAllocConnect Failed: error code IM005.
- Unable to load translation DLL: error code IM009.

Entries are not buffered for any other error codes. This is to prevent entries that will always cause an error from being buffered, such as a syntax error in an SQL string.

You should only enable buffering for SqlExec objects that are inserting new data into a database, such as an SQL insert command.

When entries are being buffered, they are buffered according to the data source with which they are trying to connect. This prevents an SqlExec object being buffered for an Excel data source from blocking an SqlExec object connected to an MS Access data source.

Separating the buffers by data source also keeps entries from different SqlExec objects trying to connect to the same data source in their proper chronological order.

If an SqlExec object has to buffer data, it keeps the data in a directory called `dsndata` in your Lookout data directory. The data is kept in a CSV file, you can view in Excel. Do not try to view these files while Lookout is running.

Once an object starts buffering, it will buffer all subsequent entries to that data source for all objects that have buffering turned on. Because of this, an entry with a syntax or other error that should not be buffered, can be buffered. If this occurs, the buffering system discards that entry after the entry is passed to ODBC and ODBC returns an error. You are not notified when the entry is discarded.

If an object has buffering turned off, and that data source is currently being buffered, then that entry will bypass the buffer entirely. It will then, either succeed or fail according to the state of the connection to the data source.

If a data source has buffered data, the buffering system automatically tries to reconnect to the data source periodically. Once it connects, it clears the buffer by periodically de-queuing the first few entries and sending them to ODBC.

During buffering, the **Failure** data member has a slightly different meaning. If it is high, then the first entry in the buffer failed and was kept in the buffer. If it is low it means that the first entry in the buffer was successfully submitted to ODBC, and objects are still being buffered until the buffer can be cleared.

If you are creating and testing a process file, you may accumulate a large amount of buffered data in the `dsndata` directory. You can safely delete this directory when Lookout is *not* running. You will lose all the buffered data, but the `SqlExec` objects will load with no errors or alarms.

## Status Messages

### **ODBC Environment not allocated. No buffering will occur**

Indicates that memory could not be allocated for the connection to ODBC. This only occurs if your system is about to run out of memory. No buffering occurs if this alarm is set. Data that has already been buffered will not be lost.

### **ODBC(32).DLL Not loaded**

Indicates that the DLL could not be loaded. No buffering can occur if this alarm is set. Data that has already been buffered, will not be lost.

### ***Data Source Name: Objects are being buffered for this data source***

This indicates which data sources are being buffered. Once the buffered entries have been cleared, the alarm ceases.

### **Incorrect data source string, check syntax**

This means that the Data Source string is missing the `DSN=driver name;` pair.

SqlExec displays any error messages that it receives from ODBC as an alarm. If the error is an error that should cause buffering, the status message indicates whether the entry was buffered or not buffered. Each message returned by ODBC belongs to the ODBC group. Some of the more common ODBC error messages are:

**[Not Buffered] 37000/-3100: [Microsoft][*Driver Name*] Syntax error in query expression ‘ ...’**

The SQL string has a syntax error in it.

**[Not Buffered] S1000/-1811 [Microsoft][*Driver Name*]Couldn’t find file ‘(unknown)’**

This usually means that your data source expression is incorrect, or you have ODBC configured incorrectly for that data source expression, such as trying to connect to an Excel 5.0 workbook your ODBC Excel driver set up for Excel 4.0 worksheets. Check the data source expression and the configuration of the ODBC driver.

**[Not Buffered] 42000/-1809 [Microsoft][*Driver Name*]can’t update. Database or object is read-only.**

The file has been opened in read-only mode, but you are trying to write to it. If you have the data member **ReadOnly** set to true, set it to false. For some drivers, you can specify in the ODBC administration tool that all files opened by that driver should be opened in read-only mode. Check the ODBC configuration if you have **ReadOnly** set to false and are still getting this error. For some ODBC drivers (such as Citadel) the file is always opened in read-only mode to protect the integrity of the files.

## Toshiba Mseries/Toshiba Tseries

Toshiba is a protocol driver class Lookout uses to communicate with Toshiba M Series Ex100, M20, M49 and T Series T1, T2, and T3 devices using the Host Link serial communication protocol.

A Toshiba object contains a great deal of data. It supports reading and writing of all predefined data points. When you create a Toshiba object, you have immediate access to all the data members for that object (*see data member list below*).

**Create Toshiba Secondary**

Tagname:

Address:  PLC Model:

Communication Settings

Serial port:  Data bits:

Baud Rate:  Stop bits:

Parity:

Phone number:

PollRate =

Poll =

Communication alarm priority:

Retry attempts:  Receive timeout:  msec

Skip every  poll requests after comm failure

**Create Toshiba Secondary**

Tagname:

Address:  PLC Model:

Communication Settings

Serial port:  Data bits:

Baud Rate:  Stop bits:

Parity:

Phone number:

PollRate =

Poll =

Communication alarm priority:

Retry attempts:  Receive timeout:  msecs

Skip every  poll requests after comm failure

**Address** specifies the address of the PLC. The maximum address for a T Series PLC is 32 and for an M Series PLC, 15.

**Serial port** specifies which comm port the object uses for communicating to the external device. This does not specify the communication type. Communication type is determined by the Options→Serial Ports... command.

**Data rate** indicates the baud rate that Lookout uses to communicate with the hardware device. The **Data rate** setting should match the selection made on the physical device.

**Data bits** indicates the number of data bits that Lookout uses to communicate with the hardware device. The **Data bits** setting should match the selection made on the physical device.

**Stop bits** indicates the number of stop bits that Lookout uses to communicate with the hardware device. This setting should match the selection made on the physical device.

**Parity** indicates the parity that Lookout uses to communicate with the hardware device. This **Parity** setting should match the selection made on the physical device.

**Phone number** specifies the number to be dialed if the selected serial port is configured for dial-up. This **Phone number** only applies to the individual protocol object.

**PollRate** is a numeric expression that determines how often to poll the device. Lookout converts the numeric value of **PollRate** into a time signal that represents days and fractions of a day. The object then polls the device at the specified time interval. Normally, this is a simple time constant such as 0:01 (one second). *See the Numeric Data Members section in Chapter 5, Using Objects, of your Lookout Reference Manual for more information on entering time constants.*

**Poll** is a logical expression. When transitioned from FALSE to TRUE, it causes Lookout to poll the device. This could be a simple expression, such as the signal from a pushbutton, or it could be a complex algorithm.

**Communication alarm priority** determines the priority level of alarms generated by the Toshiba object. Such alarms are typically related to communications with the physical device.

**Retry attempts** specifies the consecutive number of times Lookout attempts to establish communications with a device when it is not getting a valid response. After it tries the number of **Retry attempts** specified, the Toshiba object generates an alarm and releases the communication port back to the communications subsystem (COMSUB). The subsystem then moves on to the next device in the polling queue (if any). *See Chapter 6, Serial Communication, in your Lookout Reference Manual for more information.*

**Receive timeout** is the time delay Lookout uses in waiting for a response from a device before retrying the request.

The **Skip every...** setting instructs Lookout not to poll a device it has lost communication with on every scheduled poll. Instead, Lookout skips the device in the polling cycle. Once communications have been reestablished, the device is polled on its regular cycle.

As with all Lookout drivers, you can access I/O points and other data through data members. The following is a table of data members currently supported by the Toshiba object class.

*Toshiba M Series data members*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
CommFail	logical	yes	no	Object-generated signal that is on if, for any reason, Lookout cannot communicate with the device(s).
D:0, D:1536	numeric	yes	yes	Data register.
R:0, R:1024	logical	yes	yes	Auxiliary relay device.
RW:0, RW:64	numeric	yes	yes	Auxiliary relay register.
T:0, T:128	numeric	yes	yes	Timer register.
Y:0, Y:512	logical	yes	yes	External output device.
Poll	logical	no	yes	Lookout expression that when transitioned from false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device polling frequency.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.
YW:0, YW:64	numeric	yes	yes	External output register.
Z:0, Z:512	logical	yes	yes	Link device.
ZW:0, ZW:64	numeric	yes	yes	Link register.
C:0-C:1536	numeric	yes	no	Counter.
CommFail	logical	yes	no	Object-generated signal that is on if, for any reason, Lookout cannot communicate with the device(s).
D:0, D:1536	numeric	yes	yes	Data register.

*Toshiba T Series data members*

<b>Data member</b>	<b>Type</b>	<b>Read</b>	<b>Write</b>	<b>Description</b>
Poll	logical	no	yes	Lookout expression that when transitioned from false to true causes the device to be polled.
PollRate	numeric	no	yes	Lookout expression that determines the device polling frequency.
R:0, R:1024	logical	yes	yes	Auxiliary relay device.
RW:0, RW:64	numeric	yes	yes	Auxiliary relay register.
S:0-S:1024	logical	yes	yes	Binary timer register.
SW:0-SW:62	numeric	yes	yes	Data register.
T:0, T:128	numeric	yes	no	Timer register.
Update	logical	yes	no	Object-generated signal that pulses low each time it polls the device.
X:0, X:512	logical	yes	no	External input device.
XW:0, XW:64	numeric	yes	no	External input register.
Y:0, Y:512	logical	yes	yes	External output device.
YW:0, YW:64	numeric	yes	yes	External output register.



## Status Messages

### No response within timeout period

Lookout received no response from a device within the **Receive timeout** period. The Toshiba object is able to use the comm port, but when it polls the device, it does not respond—as if it is not even there.

### Toshiba errors reported in the response

These errors are reported by the Toshiba device and are in turn reported to the user in text form.

**Missing address marker in frame.**

**Invalid address in response.**

**Invalid command in response.**

**Missing BCC marker in frame.**

**Invalid BCC.**

**Missing end of frame marker.**

All these alarms indicate a garbled response frame. Check the receive gap or the retry setting in Lookout.

### Toshiba models supported:

M Series: EX100, M20, M49

T Series: T1, T2, T3